[Migración de un proceso ETL a un ecosistema Hadoop]

Título: Migración de un proceso ETL a un ecosistema Hadoop

Autor: Manel Bonilla Rodriguez

Fecha defensa: 1 de Julio del 2015

Director: Carlos Villanova Arboledas

Business Intelligence

Empresa: everis BPO, S.L.U.

Ponente: Oscar Romero Moral

ESSI - FIB

Grado en Ingeniería Informática

Ingeniería del Software

Facultad de Informática de Barcelona (FIB)

Universidad Politécnica de Cataluña (UPC) – BarcelonaTech

Agradecimientos

En primer lugar me gustaría dar las gracias **a mi familia**, aquellas personas que han hecho de mi lo que soy y que han vivido conmigo a lo largo de los años, sin ellos nada de esto hubiera ni siquiera comenzado.

En segundo lugar me gustaría agradecer a mi pareja, **Julia Beltrán;** las grandes dosis de paciencia, cariño y compresión que invierte en mí constantemente.

A mis amigos **Jordi Bueno**, **Diego Perez y Meherana Hoq**, los cuáles durante las diferentes etapas de la vida me han ayudado a desconectar y a tomarme las cosas con algo más de calma y perspectiva.

En cuarto lugar agradezco a **mis compañeros de universidad,** especialmente a **Jazzey Semira** y **David Sánchez,** el tiempo que pasamos juntos. Sin ellos habría caído muchas más veces durante el transcurso de la carrera, con ellos he aprendido y superado las diferentes adversidades que han surgido durante la carrera.

A **Carlos Villanova** y a **Oscar Romero**, por guiarme en la realización de este TFG, y sus comentarios sobre el mismo, sin su ayuda este TFG sería bastante distinto.

Finalmente me gustaría agradecer a todos mis compañeros de everis, con especial mención a:

Cristian Gumbau, Maria Ramon, Ferran Gay , Alba Martin y Victor Castellar

Sin los cuáles mi estancia allí habría sido muy distinta a como ha sido.

Muchas gracias a todos

Resumen

Castellano – Migración de un proceso *ETL* a un ecosistema Hadoop

Evaluación de las horas dedicadas a las diferentes partes del proyecto, juntamente con los resultados obtenidos. Se analizan las diferentes dificultades tecnológicas encontradas para finalmente extraer una serie de conclusiones tanto del proyecto como de los contenidos aprendidos.

Català – Migració de un procés *ETL* a un ecosistema Hadoop

Avaluació de les hores dedicades a les diferents parts del projecte, juntament amb els resultats obtinguts. S'analitzen les diferents dificultats tecnològiques trobades per finalment extreure un seguit de conclusions tant del projecte com dels continguts apresos.

English - Migration of an ETL process to a Hadoop ecosystem

Evaluation of the hours spent on the different parts of the project, together with the results. The various technological difficulties encountered are analyzed to finally extract a series of conclusions for both, the project and for the learned content.

Índice

1 Introducción y estado del arte

Este trabajo de fin grado ha sido realizado gracias a la colaboración de la empresa everis y de la Facultad de informática de Barcelona, mediante la realización de un convenio de cooperación educativa por el autor del presente documento. Este trabajo de fin de grado trata sobre la realización de una migración de un proceso ya existente de *ETL* (*extract, transform and load*; extracción, transformación y carga) a un ecosistema Hadoop.

1.1 Formulación del problema

En la actualidad las empresas tienen la necesidad de poder analizar los datos que almacenan con la mayor brevedad disponible. Hoy en día las empresas compiten en un mercado global y cambiante, por ello las empresas deben ser capaces de obtener una ventaja competitiva frente a sus rivales en el mercado, de manera que puedan tomar las mejores decisiones posibles; y tomar ventaja frente a sus principales competidores diferenciándose de ellos.

Para poder tomar las mejores decisiones posible se vuelve imprescindible un análisis de la información a la que tienen acceso las empresas. Este análisis debe ser rápido, y debe asegurarse que los datos que se utilicen para ello dispongan de la calidad suficiente.

Dada la gran cantidad de datos de que disponen las organizaciones, estas en muchas ocasiones deben integrar la información de diferentes fuentes en un único lugar asegurándose que los datos son compatibles entre sí. Gestionar tal volumen de datos puede llegar a ser un procedimiento complejo, normalmente dividido en procesos distintos y de relativamente larga duración. Para esta integración de datos diversos se suelen utilizar procesos *ETL*.

Debido al aumento del cada vez más grande volumen de datos que deben gestionar los procesos *ETL*, se desea comprobar si las herramientas disponibles en *Big Data* se pueden utilizar para implementar procesos *ETL* y de qué forma.

Las herramientas típicas del *Big Data* son las del ecosistema Hadoop, estas herramientas se encuentran actualmente en expansión y desarrollo, por ello el resultado que se obtendrá será el de una de las posibles soluciones en *Big Data*.

1.2 Objetivos

1. Seleccionar el proceso *ETL* a migrar hacia Hadoop
   1.1. Observar el conjunto de procesos *ETL* disponibles con preferencia por los de mayor tiempo de proceso de datos
2. Tomar constancia de la situación inicial del proceso *ETL* a migrar
   2.1. Anotar el tiempo que dura el proceso para su posterior comparación
   2.2. Analizar qué tipos de tablas y de datos requiere el proceso para llevarse a cabo.
3. Realizar una migración de los datos necesarios para la representación del proceso *ETL* en Hadoop
   3.1. Cargar los diferentes datos en el ecosistema Hadoop desde ficheros externos con los datos volcados
   3.2. Cargar los diferentes datos en el ecosistema Hadoop utilizando las herramientas de acceso a bases de datos relacionales
4. Representar el proceso *ETL* en Hadoop
   4.1. Representar con las diferentes herramientas de las que se dispone en el ecosistema Hadoop el proceso *ETL* utilizando los datos de volcado
   4.2. Representar con las diferentes herramientas de las que se dispone en el ecosistema Hadoop el proceso *ETL* utilizando los datos de volcado y los de base de datos relacional
5. Comparar los tiempos de ejecución de ambos procesos y extraer conclusiones
   5.1. Analizar la mejora obtenible por el sistema respecto al proceso *ETL* actual
   5.2. Analizar posibles mejoras en el proceso *ETL* migrado al ecosistema Hadoop
   5.3. Analizar la flexibilidad y la transparencia del resultado final

1.3 Contexto

1.3.1 Extracción, transformación y carga

Los procesos *ETL* o de extracción, transformación y carga, permiten a las organizaciones recopilar en una única base de datos todos los datos de los que pueden disponer. Usualmente todos estos datos provienen de diversas fuentes, por lo que es necesario acceder a ellos, y formatearlos para poder ser capaces de integrarlos. Además, es muy recomendable asegurar la calidad de los datos y su veracidad, para así evitar la creación de errores en los datos.

Los procesos *ETL* tienen la finalidad de extraer los datos para ser almacenados en una base de datos, un *data mart* o un D*ata Warehouse*, donde en el futuro estos datos serán utilizados para su análisis o en un sistema operacional/transaccional para apoyar un proceso de negocio.

Otra posible utilidad de los procesos *ETL* es la posibilidad de utilizarlos para la integración de nuevos sistemas con otros sistemas heredados.[1]

Dada la gran variedad de posibilidades existentes para representar la realidad en un dato, junto con la gran cantidad de datos almacenados en las diferentes fuentes de origen, los procesos *ETL* consumen una gran cantidad de los recursos asignados a un proyecto.

Extracción

Esta fase de un proceso *ETL* es la encargada de recopilar los datos de los sistemas originales y transportarlos al sistema donde se almacenarán, de manera general suele tratarse de un entorno de *Data Warehouse* o almacén de datos. Los formatos de las fuentes de datos pueden encontrarse en diferentes formatos, desde ficheros planos, bases de datos relacionales entre otros formatos distintos.

Una parte de la extracción es la de analizar que los datos sean los que se esperaban, verificando que siguen el formato que se esperaba, para sino ha sido así ser rechazados.

Una de las características que se deben tener en cuenta al generar una fase de extracción es reducir al mínimo el impacto que se generase en el sistema origen de la información. No se puede poner en riesgo el sistema original, generalmente operacional; ya que si colapsase esto podría afectar el uso normal del sistema y generar pérdidas a nivel operacional.

Transformación

En esta fase se espera realizar los cambios necesarios en los datos de manera que estos tengan el formato y contenido esperado. De esta manera los datos se encontrarán listos para ser cargados en el sistema de destino.

Carga

Fase encargada de almacenar los datos en el destino, un *Data Warehouse* o en cualquier tipo de base de datos. Por tanto la fase de carga interactúa de manera directa con el sistema destino, y debe adaptarse al mismo con el fin de cargar los datos de manera satisfactoria.

1.3.2 Big Data

En los últimos años se ha reducido tanto el coste de mantener una información en disco que actualmente las organizaciones no se pueden a arriesgar a no almacenar una información que les podría ser de utilidad. Debido a este gran almacenaje de información por parte de las organizaciones nos encontramos con compañías con volúmenes muy grandes de información, que debe tratarse de manera muy rápida.

Para explicar los diferentes aspectos del *Big Data* se le suele asociar con las siguientes tres Vs: Volumen, Velocidad y Variedad.[2]

Volumen

Un sistema *Big Data* dispone de las características y herramientas necesarias para tratar un gran volumen de datos, esto es muy importante ya que existen ciertos problemas actuales de rendimiento por culpa de las magnitudes de los datos actuales a tratar. Este tipo de sistemas son capaces de trabajar con cantidades de datos de Terabytes de magnitud.

Velocidad

Los sistemas *Big Data* permiten el tratamiento de la información de manera muy rápida reduciendo los tiempos de espera, siendo lo más deseable análisis en tiempo real. Para ello, han de ser capaces de cargar información de manera muy rápida, y poder tratarla de manera eficiente.

Variedad

Están preparados para tratar diferentes formatos de datos de diferentes fuentes, tanto estructurados como no estructurados.

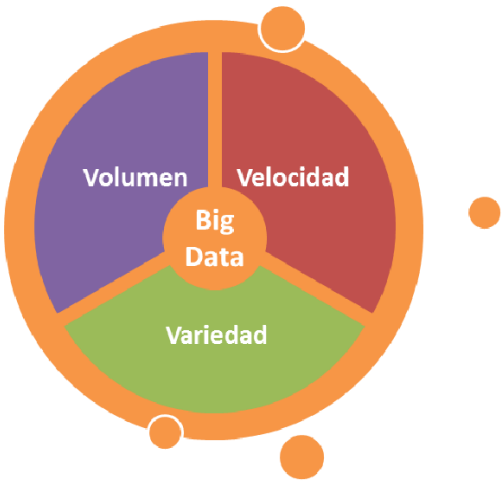


Figura 1: Esquema aspectos *Big Data*

Gartner describe *Big Data* de la siguiente forma:

"Big Data is high-volume, high-velocity and high-variety information assets that demand cost-effective, innovative forms of information processing for enhanced insight and decision making." [3]

En el transcurso de este proyecto nos centraremos en la velocidad de análisis de los datos, ya que los datos son estructurados y el volumen no es demasiado grande pero aun así el proceso consume un tiempo muy valioso.

Existen muchísimas herramientas que hacen uso del *Big Data*. Merece la pena destacar algunas de ellas.

Ecosistema Hadoop

Apache Hadoop es un proyecto de *software* de código abierto para la computación fiable, escalable y distribuida. Apache Hadoop aporta un *framework* que permite el procesado de grandes conjuntos de datos de manera distribuida a través de la utilización de sencillos modelos de programación. Preparado para escalar de un número reducido de máquinas, a miles, cada una de ellas ofreciendo computación local y almacenamiento. [4]

El *framework* base de Apache Hadoop contiene los siguientes módulos:[5], [6]

- Hadoop Common – Contiene librerías y herramientas para el resto de módulos.
- Hadoop Distributed File System (HDFS) – Sistema de ficheros distribuido y escalable, escrito en Java para Hadoop.
- Hadoop YARN – Plataforma de gestión y planificación de los recursos para las aplicaciones de los usuarios.
- Hadoop MapReduce – Modelo de programación para procesado de datos de gran escala.

Desde 2012, cuando se habla sobre Hadoop, se suele hablar sobre su ecosistema, por tanto del *software* base como de otras aplicaciones que pueden instalarse por encima o junto a él, tales como Apache Pig, Apache Hive, Apache Hbase, Apache Spark, Hbase, Cassandra entre otras.

1.4 Estado del Arte

En la actualidad existen principalmente dos vertientes a la hora de realizar un proceso *ETL*:

- Mediante *script*s

  En este tipo de soluciones se escriben *scripts* que acceden a la base de datos para cargar la información y lanzan las consultas necesarias para realizar todos los pasos que se deben generar dentro del proceso de *ETL*. Esta solución es de alta flexibilidad pero por el contrario es más difícil de crear y mantener. Este tipo de soluciones permiten corregir errores fácilmente ya que permiten realizar lo que se desee generalmente mediante lenguaje *SQL*, siendo la reutilización de código un factor importante a considerar en este tipo de soluciones. Necesitan de un tiempo de elaboración mayor, por lo tanto son más costosas; se ha de realizar la documentación manualmente al igual que el mantenimiento.

- Mediante herramientas

  Este tipo de soluciones intentan facilitar el proceso *ETL* realizando gráficamente un flujo de pasos a seguir tal y como se realizaría el proceso, y la herramienta traduciría este flujo automáticamente a *scripts* para realizarse el proceso *ETL*. En esta solución disminuye la flexibilidad, pero aporta una mayor facilidad para su creación y mantenimiento. Estas herramientas aportan facilidades para el usuario como visualizar el proceso como un flujo, generando documentación de manera automática. Al tratarse de una herramienta gráfica facilita realizar operaciones al usuario. Por el contrario existe cierta incertidumbre sobre que se realiza en concreto, y en ocasiones hay que adquirir licencias para poder ser utilizadas.

Hasta la fecha existen dos formas de ingerir los datos desde fuentes externas al ecosistema:

- Mediante ficheros de volcado

  Este método de ingerir los datos consiste en acceder a los datos existentes dentro de unos ficheros de volcado ya generados. Estos ficheros de datos tienen la ventaja de que pueden contener de manera parcial o total los datos necesarios. Además estos ficheros de volcado suelen ser generados mediante el sistema de datos original, por lo que no es necesario una interacción directa con los datos originales. Por el contrario es necesario conocer el formato de los datos a cargar, y como están los mismos estructurados dentro del fichero de volcado.

- Mediante herramientas

  El otro método de acceso a los datos consiste en acceder a los datos de manera directa. Este método utiliza una, o varias; herramientas las cuales accediendo al sistema original realizan una lectura de los datos almacenados para ser capaces de guardarlos en el sistema de destino. De forma general estas herramientas son compatibles con los principales productos de bases de datos relacionales y permiten automatizar el acceso a los datos y a su formato, facilitando la carga de los datos.

  No existe una solución perfecta o universal que sea la adecuada para aplicarse a este tipo de problemas. La solución que se deberá aplicar dependerá del propio proyecto, los datos a tratar, la disponibilidad, la complejidad, entre otros.

1.4.1 Estudios existentes

Extract, Transform, and Load Big Data with Apache Hadoop

Existe un estudio anterior sobre la creación de un proceso *ETL* dentro de Hadoop realizado por Intel en el año 2013. [7]

En la siguiente cita podemos ver uno de los motivos analizado por ellos para la realización de *ETL* con tecnologías de *Big Data*. Siendo este motivo un motivo de volumen, velocidad y variedad, por tanto idóneo para las tecnologías tratadas.

"None of these solutions is cheap or simple, and their cost and complexity are compounded by Big Data. Consider eBay, which in 2011 had over 200 million items for sale, separated into 50,000 categories, and bought and sold by 100 million registered users— all of which entailed roughly 9 petabytes of data. Google reportedly processes over 24 petabytes of data per day. AT&T processes 19 petabytes through their networks each day, and the video game World of Warcraft uses 1.3 petabytes of storage. All of these figures are already out-of-date as of this writing because online data is growing so fast." [7]

En resumen este libro blanco (o *White Paper*) trata algunas de las consideraciones tanto de *hardware* como de *software* que se han de tener en cuenta a la hora de utilizar Hadoop para realizar un *ETL*. Analiza los aspectos que motivan a llevar a cabo un proyecto de estas características, analiza también las diferentes opciones que ellos consideran a la hora de seleccionar que *software* del ecosistema de Hadoop van a utilizar para la implementación y las diferentes implicaciones que conlleva la realización de un proyecto de estas características.

ETLMR: A Highly Scalable Dimensional ETL Framework Based on

Artículo que hace referencia a una implementación que mejora ciertos aspectos del paralelismo de Hadoop, la cual no se encuentra disponible en la red. Añadimos algunas citaciones que se pueden encontrar interesantes de leer:

"However, lacks support for high-level ETL specific constructs, resulting in low ETL programmer productivity.

…In recent years, a novel "cloud computing" technology, , has been widely used for parallel computing in data-intensive areas. A program is written as map and reduce functions, which process key/value pairs and are executed in many parallel instances.

…There are some data warehousing tools available, including Hive, Pig and Pentaho Data Integration (PDI). Hive and Pig both offer data storage on the Hadoop distributed file system (HDFS) and scripting languages which have some limited ETL abilities. They are both more like a DBMS instead of a full-blown ETL tool.

… To implement an ETL program to function in a distributed environment is thus very costly, time-consuming, and error-prone. , on the other hand, provides programming flexibility, cost-effective scalability and capacity on commodity machines and a framework can provide inter-process communication, fault-tolerance, load balancing and task scheduling to a parallel ETL program out of the box.

…PDI is an ETL tool and provides Hadoop support in its 4.1 GA version. However, there are still many limitations with this version. For example, it only allows to set a limited number of parameters in the job executor, customized combiner and mapperonly jobs are not supported, and the transformation components are not fully supported in Hadoop." [8]

Es un artículo interesante ya que hace referencia a ciertos aspectos técnicos sobre los *ETL* a tener en cuenta en la realización de un proyecto como este, aun así se ha tenido en mayor consideración un artículo posterior de los mismos autores [9].

Performance Comparison of Hadoop Based Tools with Commercial ETL Tools

Este artículo es sobre un proyecto similar al que estoy realizando, en este proyecto se comparan 2 soluciones, una utilizando herramientas de *ETL* comerciales, y otra utilizando Hadoop. La comparación se realiza teniendo en cuenta tanto la velocidad de procesado, como el coste. Con todo y con eso no se detalla los conjuntos de datos utilizados, ni se ofrece la implementación que realizaron. Introduzco algunas citaciones que pueden ser de utilidad:

"The data needs to be collected, cleaned, curated and stored in a way that information retrieval and analysis for business intelligence becomes easy. This demand is accentuated due to requirement of collapsing processing window duration. ETL constitutes 60%-80% of business intelligence projects out of which ET is the major component.

The usual approaches to address this issue have been to add hardware, or adopt a faster ETL tool, or reduce refresh cycle of master data, and similar measures. However, now the data volume has reached to such a level where these steps are either increasing operational cost or reducing responsiveness of the business, or both, hence non-sufficient. This has lead us to evaluate options for moving from high performing ETL tools to new technology open source low cost option that uses Map Reduce (M/R) paradigm. We have implemented such solutions and the results are promising – as it can reduce cost and improve performance. Newer tools are emerging that uses the M/R paradigm which promises even faster processing and can take streaming inputs as well.

…Analyzing the past works and carrying out the experiments we conclude that Hadoop based solutions are better in comparison to commercial ETL tools.

…In future we would have to carry out the experiments with multiple data sets of bigger sizes to study the applicability of the technology on a wide variety of data sets." [10]

1.4.2 Soluciones existentes

Actualmente existen muy pocas soluciones para la realización de procesos *ETL* dentro del ecosistema Hadoop. Algunas de ellas únicamente muestran una serie de declaración de intenciones sin llegar a materializarse en algo. Además hace falta destacar la carencia de un estándar de trabajo a la hora de realizar un proyecto de *Big Data*, por tanto la decisión de que herramientas utilizar siempre es mediante intuición y se deben probar soluciones habitualmente a fin de determinar que herramienta es mejor para un determinado proyecto en concreto.

DMX-h

Una de las que se materializan es DMX-h un producto de Syncsort [11] , de la cual existen varios vídeos sobre su uso [12] , aunque tiene una gran problemática, es un producto empresarial y de pago, por tanto no es una solución a adaptar en el presente documento. Cabe destacar de sus características que se trata de una herramienta con interfaz visual, permite la realización de diferentes operaciones de manera gráfica generando una serie de operaciones por detrás de manera oculta al usuario.

CloudETL: Scalable Dimensional ETL for Hadoop and Hive

Este producto [9], y la empresa Queplix ya no existen, además no hemos sido capaces de encontrar una versión de CloudETL para su descarga y análisis, aun así se ha tomado su artículo como base para lo que se espera encontrar durante la realización del proyecto.

Los autores de este artículo son los mismos del artículo sobre ETLMR [8] .Realizamos una cita con lo más destacable del artículo:

"Many enterprises now collect and analyze tens or hundreds of gigabytes data each day. The data warehousing technologies are thus faced with the challenge of handling the growing data volumes in little time. However, many existing data warehousing systems already take hours, or even days to finish loading the daily data and will be to slow with larger data volumes.

…The paradigm provides cost-effective scalability for large-scale data sets and handles fault tolerance very well even on hundreds or thousands of machines.

…Writing HiveQL scripts for such processing is cumbersome and requires a lot of programming efforts.

…To tackle large-scale data, parallelization is the key technology to improve the scalability. The paradigm has become the de facto technology for large-scale data-intensive processing due to its ease of programming, scalability, fault-tolerance, etc.

…Other systems built only on top of but providing high-level interfaces also appear, including Pig and Hive. These systems provide scalability but with DBMS-like usability. They are generic for large-scale data analysis, but not specific to ETL. " [9]

Por tanto concluimos en que el producto detallado en el artículo ha desaparecido actualmente, aún dadas las características de facilidad que se mencionan en él estaba más cerca de ser una solución basada en *scripts* , no disponía de interfaz gráfica; que de ser una herramienta dedicada al *ETL*.

1.5 Actores implicados

- **Compañía everis**

  La compañía everis será una de los principales implicados y beneficiados por la realización de este proyecto. Al tratarse este proyecto de la realización de una solución para la realización de un *ETL* mediante herramientas actuales en el mercado, esto permitirá obtener una ventaja competitiva respecto sus soluciones anteriores y respecto a sus competidores. Gracias a esta ventaja competitiva podrá conocer una alternativa más para la realización de algunos de sus proyectos, y las implicaciones que esto tendría sin necesidad de arriesgarse en un proyecto para un cliente. Además, tendrá acceso a todos los materiales realizados durante este proyecto pudiéndolos reutilizar si fuese conveniente en proyectos posteriores.

- **Director del proyecto**

  El director del proyecto es Carlos Villanova Arboledas. Es quien se encargará de guiar al alumno durante la realización del proyecto, supervisará el trabajo del autor del proyecto y quien proporciona a través de everis los materiales utilizados para la realización del proyecto.

- **Equipo desarrollador**

  El equipo desarrollador de este proyecto consistirá de una sola persona, el alumno; durante su estancia de prácticas en everis. Dado que únicamente será desarrollado por el alumno este deberá asumir diferentes roles dentro del proyecto para así ser capaz de abarcar las diferentes partes que lo componen.

- **Tema de estudio**

  Debido a la escasa información sobre estudios anteriores referente a la realización de procesos *ETL* dentro de un ecosistema Hadoop, todo el conocimiento adquirido durante este proyecto y aquí expuesto será de valor tanto para el campo de estudio de los *ETL* como para el campo de estudio de las nuevas tecnologías de *Big Data*.

2 Alcance del proyecto

Dada la naturaleza de este proyecto y el tiempo disponible se ha optado por una simplificación respecto al conjunto original de procesos *ETL* de origen. Este proyecto abarca la migración de un proceso *ETL* a un ecosistema de Hadoop dado que se desconoce inicialmente la dificultad total de la migración y los resultados que se obtendrán de esta.

Este documento no expondrá la migración de la totalidad de los procesos *ETL* disponibles al ecosistema Hadoop, ya que se ha optado por realizar un primer acercamiento y analizarlo con el suficiente detalle para determinar la viabilidad y mejora que conllevaría.

Dado el hecho de que podemos disponer de los datos de origen del proceso *ETL*, se decidió que se realizarían dos versiones de carga de los datos, de manera que fuera posible compararlas entre ellas. Una primera versión, partiendo del volcado de datos del sistema original. Y una segunda, cargando estos volcados de datos en un base de datos relacional y accediendo a los datos de ella. De este modo será posible comparar el tiempo que tarda cada una, y el esfuerzo que ha sido necesario para adaptarla a los datos.

2.1 Limitaciones y riesgos

Existen una serie de limitaciones en este proyecto que se deben tener en cuenta y que pueden afectar de cierta manera el avance del proyecto.

La principal limitación es la cantidad de conocimientos que se deben adquirir. Por un lado se ha de tener en cuenta la gran variedad de herramientas que existen en el ecosistema Hadoop, sin menospreciar la cantidad de versiones que existen de estas. Otro factor a tener en cuenta es que el proceso final una vez debidamente creado y configurado sea algo automático y lo más transparente posible al usuario final. Por otro lado, existen ciertas particularidades en los procesos y en los propios datos que forzarán al uso de herramientas específicamente diseñadas para llevar a cabo el proceso *ETL*.

En una primera fase del proyecto se comenzará a trabajar con alguna de las herramientas ya instaladas de Hadoop que están disponibles en internet gratuitamente para la descarga. Esto será así para poder comenzar a desarrollar las primeras partes del proyecto en local, y después en un futuro se espera llevar esto a un conjunto de servidores con Hadoop conectados entre sí para evaluar el tiempo del sistema en un entorno distribuido. Por tanto existen unas limitaciones físicas, las de la máquina que se utilice en local. Y además existen unos riesgos sobre la disponibilidad de un servidor que cumpla las condiciones necesarias para utilizar un ecosistema Hadoop distribuido.

Para finalizar, mencionar que existe una limitación temporal en la realización de este proyecto, existiendo una fecha de entrega del documento final. Esta limitación temporal limita las opciones del proyecto y lo que se puede llevar a cabo en él.

2.2 Metodología y rigor

2.2.1 Método de trabajo

Para llevar a cabo la realización del proyecto se realizará un desarrollo iterativo e incremental, dividido en diferentes fases. Al finalizar algunas de las fases que conforman el proyecto, se espera una solución completa y funcional del mismo. Al finalizar una iteración existirá una tarea en la cual se evaluará lo realizado durante la fase. Fases posteriores incrementarán las soluciones de fases anteriores del proyecto. Además existirá una fase final dónde se evaluarán los resultados obtenidos en el conjunto de las fases anteriores del proyecto.

Cada una de las fases consta de un conjunto de tareas, entre las cuales existe una serie de precedencias, siendo imposible finalizar la fase si aún existe alguna de las tareas que lo conforman sin finalizar.

Al no disponerse de grandes conocimientos sobre las herramientas de las que se va a disponer es necesario que el proyecto mantenga cierta flexibilidad a la hora de asignar el tiempo para la realización de las tareas por si es necesario asignar más tiempo al desarrollo de alguna.

Cíclicamente se realizará una copia de seguridad cuando se haya conseguido un hecho relevante, o en caso contrario, cuando se ha dedicado un par de días a la implementación de funcionalidades del proyecto. De este modo podremos restaurar a una versión anterior del proyecto en cualquier momento posterior manteniéndonos a salvo de cambios que no aporten mejoras o dañen funcionalidades del sistema.

Se desarrollará en local, por lo que no existirá ningún riesgo de malmeter los datos originales, ya que se trabajará con ficheros locales (un volcado de los datos en ficheros) y sin acceso directo a los originales. Se irán incrementando las funcionalidades en las fases hasta obtener una versión completa y segura del proceso que hemos migrado.

2.2.2 Herramientas de Seguimiento

La gestión por parte del director consistirá en dar consejo al estudiante en la toma de decisiones sobre el proyecto. El director recibirá un email semanal por parte del estudiante con las tareas que se han realizado, que limitaciones se han enfrentado y como se han solucionado. Además, este email incluirá la previsión de tareas a realizar en la siguiente semana, de manera que si algo no es correcto se pueden tomar medidas para redirigir los esfuerzos en la dirección adecuada realizando los cambios oportunos.

El director del proyecto siempre podrá acceder tanto a las copias de seguridad del proyecto como a todos los materiales elaborados para el mismo. Esto es así ya que estos estarán disponibles en una carpeta en red a la cual se podrá acceder siempre que se tengan los datos de la misma y se esté conectado a la red interna de la compañía.

2.2.3 Método de validación

Una vez se hayan cargado los datos desde los ficheros externos al ecosistema Hadoop se procederá a comprobar que los datos han sido cargados de manera correcta y completa, asegurando por tanto, que los datos que se disponen para el proceso son correctos y seguros. Estas comprobaciones incluirán comparar el número de filas disponibles y comparar, en la medida que los formatos de los datos lo permitan; cada una de las columnas y de las filas de ambos ficheros para asegurar que son idénticos.

Para poder comprobar el rigor de la migración del proceso *ETL* original a Hadoop se compararán las tablas resultantes entre sí de ambos procesos para poder asegurar que ambos producen resultados idéntico. Esta comparación asegurará la rigurosidad de los resultados y de los métodos utilizados durante toda la migración del proceso *ETL*.

3 Planificación temporal

3.1 Características generales

Aquí se explicita la planificación que se ha realizado del proyecto. La importancia de esta planificación radica en poder controlar el desarrollo del proyecto con tal de ser capaces de tomar las precauciones necesarias, y realizar medidas correctivas en caso de ser necesario; para asegurar la finalización del proyecto.

3.1.1 Duración

La duración prevista de este proyecto es de un total de **6 meses**, ya que el proyecto inició la primera de las fases el 18 de diciembre del 2014 y se ha previsto la finalización del documento para el 15 de Junio del 2015.

3.1.2 Consideraciones globales

Dado que el proyecto se va a llevar a cabo por una única persona, no existe la posibilidad de realizar dos tareas en el mismo espacio de tiempo. Además el estudiante es el único desarrollador dedicado al proyecto, por lo tanto si dos tareas son realizadas en paralelo se deberá por tanto a una división de la dedicación del estudiante entre esas dos tareas.

Teniendo en consideración lo anterior las relaciones de precedencia limitarán la evolución del proyecto, ya que hasta que no se haya llevado la tarea anterior hasta la completitud no se podrá seguir con la siguiente tarea de manera adecuada. Por ello se considera innecesaria la realización de un diagrama de Pert, dado lo lineal del proyecto entre manos.

3.2 Recursos

Este proyecto consta de dos tipos básicos de recursos que se van a utilizar. Por un lado vamos a requerir de una serie de **recursos humanos**, y por el otro vamos a disponer de una serie de **recursos materiales**. El conjunto de los recursos aquí listados permitirán llevar a cabo el proyecto desde su inicio hasta su fin de manera correcta.

3.2.1 Recursos humanos

- Un estudiante en un puesto de becario, con una dedicación de 20 horas semanales.
- Un trabajador, quien realizará las funciones de director del TFG, con una dedicación de 1 hora semanal.

3.2.2 Recursos hardware

- Un ordenador corporativo (mínimo 8GB de memoria RAM) con conexión a internet.

3.2.3 Recursos software

- Suite Ofimática Microsoft Office
- Gestor de Máquinas virtuales , VMWare Workstation
- Imagen de Cloudera como entorno de ecosistema Hadoop

3.3 Plan de acción y valoración de alternativas

Se ha escogido como fecha de finalización el día 15 de Junio para ser capaces de disponer de un margen adicional de un mínimo de una semana e incluso según el caso de algo más para poder gestionar posibles eventualidades que surgiesen durante la realización del proyecto. La memoria del proyecto debe entregarse una semana antes de la lectura del TFG, siendo la primera fecha de lectura del turno de Junio el día 29, por tanto la fecha máxima de entrega para ese primer turno sería el día 22 de Junio, una semana después de nuestra previsión de finalización del proyecto.

Existen ciertos limitantes a la hora de la realización del proyecto. Un ejemplo de ello puede ser la necesidad de realizar las pruebas de evaluación de la solución mediante los recursos de los que se disponga.

Dado que semanalmente se revisa con el director del TFG y se le hace entrega de un email con el contenido realizado semanalmente, se puede prever con suficiente antelación las posibles desviaciones dentro del proyecto. Debido a este análisis semanal de las tareas realizadas es posible estar atento a posibles desviaciones en el proyecto y ser capaces de tomar constancia de ello, tomando medidas de ajuste en consecuencia sobre la planificación.

Hay que tener en consideración que ciertas actividades han sido dimensionadas según lo realizado en actividades anteriores, por tanto se encuentran parcialmente sobredimensionadas ya que estas actividades anteriores al tratarse de iniciales se espera que tengan un tiempo de dedicación mayor de aprendizaje que fases posteriores.

3.4 Fases y actividades

El proyecto se encuentra dividido en 4 fases diferentes y estas divididas a su vez en algunas actividades.

3.4.1 Fase 1 - Análisis Inicial

Esta actividad tiene el objetivo de introducir los conceptos y conocimientos necesarios para poder tomar decisiones de cómo se realizará la migración del proceso. Además se seleccionará uno de los procesos *ETL* teniendo preferencia por los de mayor tiempo de proceso de datos. Con esta actividad se pretende tomar consciencia del proyecto para ser capaz de tomar las mejores decisiones para abordar el proyecto.

3.4.2 Fase 1 - Contexto inicial

Durante el desarrollo de esta se tomará constancia del tiempo que tarda el proceso escogido a migrar, anotando el tiempo que dura el proceso para ser capaces de comparar la migración con este a posteriori. Además se observará el proceso en mayor profundidad para su mayor comprensión junto con las tablas y datos necesarios parar la migración. Esto permitirá valorar las diferentes fases por las que debe pasar el proyecto, y alcanzar un conocimiento suficiente como para prever el tiempo que llevará cada una de las tareas.

3.4.3 Fase 2 –Gestión de proyectos (GEP)

El objetivo de esta tarea es la realización de los documentos necesarios para la correcta evaluación de los 3 créditos ECTS correspondientes a la asignatura GEP de la Facultad de Informática de Barcelona. Esta asignatura se trata de una obligatoria cuya nota forma parte de la nota del TFG e implica la realización de una serie de documentos evaluables e incluibles en el TFG a posteriori. Se espera que GEP cubra el periodo de tiempo comprendido entre el 16 de Febrero y el 23 de Marzo, siendo la carga de horas trabajo insuficiente para cubrir la totalidad de horas de prácticas en empresa semanales, por ello es posible paralelizarla con otra de las actividades.

3.4.4 Fase 2 y Fase 3 - Carga de los datos

Esta actividad constará de dos partes diferenciadas. Una parte inicial dentro de la segunda fase del proyecto, dónde se cargarán los datos directamente de unos ficheros con el volcado de las tablas originales. Una segunda parte dentro de la tercera fase, dónde se cargarán los datos de un sistema de base de datos relacional. De este modo se simularán dos entornos, uno primero dónde accedemos a los datos provistos por un tercero en modo de volcado, y un segundo dónde es la propia migración quien accede a estos directamente desde base de datos relacionales.

3.4.5 Fase 2 y Fase 3 - Procesado de los datos

Similar a como se realizaba el proceso *ETL* en el sistema original, se llevará a cabo el proceso *ETL* dentro del sistema migrado, utilizando las herramientas del ecosistema Hadoop que se consideren más apropiadas para este fin. Imitando en todo momento el proceso original en medida de lo posible. Aunque esta actividad se encuentra dentro de la fase 2 y de la fase 3 del proyecto se considera que la mayor parte del trabajo será realizado durante la fase 2, siendo esta actividad de la fase 3 una mera repetición de los resultados de procesado anteriores, debido a que el proceso debería realizarse del mismo modo y los datos deberán ser los mismo.

3.4.6 Fase 2 y Fase 3 - Evaluación

Se compararán las tablas generadas por el proceso ya migrado para poder asegurar su rigor. Tomaremos nota del tiempo que dura el proceso que hemos migrado al ecosistema Hadoop y procederemos a compararlo respecto al tiempo que tardaba originalmente. Se tendrán en consideración los diferentes tiempos, tanto los de carga inicial del sistema donde se cargarán los datos estáticos necesarios para el proceso, como el tiempo que tarda en generar la tabla final del proceso. Se toma como presunción de las fases anteriores que los tiempos de procesado de datos serán muy similares sino los mismos en ambas versiones. Es por ello que el aspecto a considerar será la de la carga de los datos.

3.4.7 Fase 4 - Conclusiones

Finalmente, en esta fase se analizará todo lo realizado durante el proyecto así como la manera en la que se ha realizado y los resultados obtenidos. Se extraerán unas conclusiones y conocimientos sobre ello, a la par que se tratará si el resultado es satisfactorio para la posible implantación de otros procesos *ETL*. Todas estas decisiones serán basadas en los datos obtenidos durante la evolución del proyecto, y de los tiempos y resultados finales obtenidos de la ejecución del proyecto. Esta fase además incluye la finalización de la memoria, junto con la preparación de la presentación para la lectura del TFG.

3.5 Diagrama de Gantt

Aquí debajo podemos ver en dos imágenes el diagrama de Gantt correspondiente al proyecto de este TFG. Se puede visualizar la estimación de duración en días juntamente a la fecha de inicio y de fin de cada una de las tareas y fases. Además, se puede visualizar las precedencias entre tareas, tanto de manera numérica en la tabla, como de manera visual gracias a las flechas existentes dentro del diagrama de barras.

ID	Task Name	Duration	Start	Finish	Predecessors
1	**TFG [Migración de un proceso ETL a un ecosistema Hadoop]**	**128 days**	**Thu 18/12/14**	**Mon 15/06/15**	
2	**Fase 1**	**22 days**	**Thu 18/12/14**	**Fri 16/01/15**	
3	Análisis inicial	15 days	Thu 18/12/14	Wed 07/01/15	
4	Contexto inicial	7 days	Thu 08/01/15	Fri 16/01/15	3
5	**Fase 2**	**68 days**	**Mon 19/01/15**	**Wed 22/04/15**	**2**
6	Carga de datos volcados	40 days	Mon 19/01/15	Fri 13/03/15	
7	GEP	26 days	Mon 16/02/15	Mon 23/03/15	
8	Procesado de datos volcados	13 days	Tue 24/03/15	Thu 09/04/15	7
9	Evaluación con datos volcados	9 days	Fri 10/04/15	Wed 22/04/15	8
10	**Fase 3**	**28 days**	**Thu 23/04/15**	**Mon 01/06/15**	**5**
11	Carga de datos mediante Sqoop	12 days	Thu 23/04/15	Fri 08/05/15	
12	Procesado de datos de Sqoop	8 days	Mon 11/05/15	Wed 20/05/15	11
13	Evaluación de datos de Sqoop	8 days	Thu 21/05/15	Mon 01/06/15	12
14	**Fase 4**	**10 days**	**Tue 02/06/15**	**Mon 15/06/15**	**10**
15	Conclusiones	10 days	Tue 02/06/15	Mon 15/06/15	

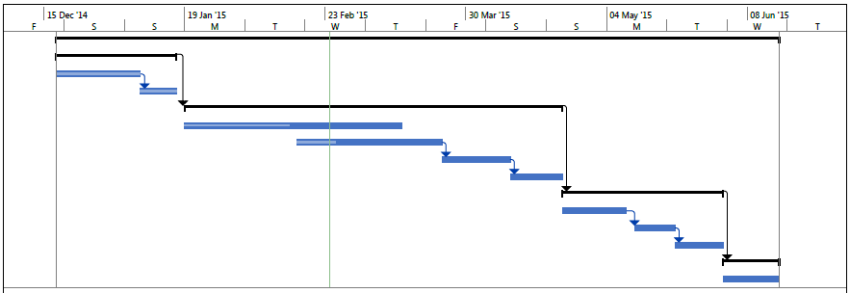


Figura 2: Diagrama de Gantt realizado en Microsoft Project.

Tal y como se indica en el diagrama de Gantt anteriormente mencionado las fases serán realizadas de manera secuencial siendo su forma final: Fase 1, Fase 2, Fase 3 y Fase 4. Este orden es realizado de esta manera ya que las actividades que conforman las fases están agrupadas según sus necesidades de actividades anteriores. La fase 2 tiene prioridad sobre la fase 3, ya que al gestionar un número menor de herramientas la dificultad debería ser menor y por tanto es más factible su implementación.

3.6 Fases añadidas

La realización de las tareas siguientes no se tuvo en cuenta durante la realización de la planificación inicial. Fueron tareas añadidas durante la realización del proyecto, y por tanto se ha de mencionar que cumplen.

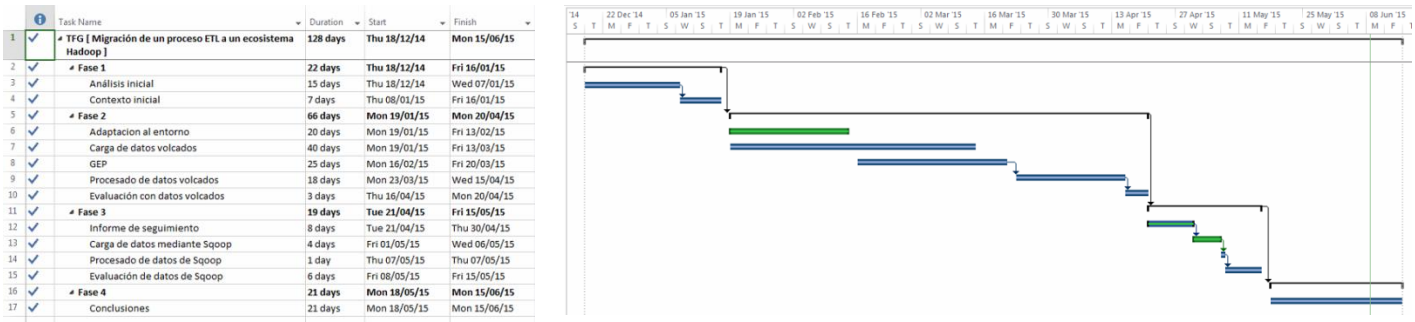
3.6.1 Fase 2 – Adaptación al entorno

Durante la tarea de adaptación al entorno se estuvo trabajando a la vez en la carga de los datos. Esta tarea ha sido creada debido a la carencia de conocimiento del ecosistema cuando se comenzó la realización del proyecto. Como únicamente se disponía de un conocimiento general de las aplicaciones, pero no de cómo acceder a ellas, o cómo utilizar el entorno en el que se encuentran; existe cierta cantidad del tiempo dedicado a comprender el funcionamiento del ecosistema y su puesta en marcha inicial para poder comenzar a trabajar en el proyecto.

3.6.2 Fase 3 – Informe de seguimiento

Durante esta tarea se realizaron los documentos entregados en la reunión de seguimiento del proyecto. Inicialmente no se había tenido en consideración ningún tipo de carga para su desarrollo, finalmente se comprobó que hubo que invertir cierta cantidad de tiempo, es por ello que se creó esta fase.

3.7 Gantt definitivo



	Task Name	Duration	Start	Finish
1	TFG [Migración de un proceso ETL a un ecosistema Hadoop]	128 days	Thu 18/12/14	Mon 15/06/15
2	Fase 1	22 days	Thu 18/12/14	Fri 16/01/15
3	Análisis inicial	15 days	Thu 18/12/14	Wed 07/01/15
4	Contexto inicial	7 days	Thu 08/01/15	Fri 16/01/15
5	Fase 2	66 days	Mon 19/01/15	Mon 20/04/15
6	Adaptacion al entorno	20 days	Mon 19/01/15	Fri 13/02/15
7	Carga de datos volcados	40 days	Mon 19/01/15	Fri 13/03/15
8	GEP	25 days	Mon 16/02/15	Fri 20/03/15
9	Procesado de datos volcados	18 days	Mon 23/03/15	Wed 15/04/15
10	Evaluación con datos volcados	3 days	Thu 16/04/15	Mon 20/04/15
11	Fase 3	19 days	Tue 21/04/15	Fri 15/05/15
12	Informe de seguimiento	8 days	Tue 21/04/15	Thu 30/04/15
13	Carga de datos mediante Sqoop	4 days	Fri 01/05/15	Wed 06/05/15
14	Procesado de datos de Sqoop	1 day	Thu 07/05/15	Thu 07/05/15
15	Evaluación de datos de Sqoop	6 days	Fri 08/05/15	Fri 15/05/15
16	Fase 4	21 days	Mon 18/05/15	Mon 15/06/15
17	Conclusiones	21 days	Mon 18/05/15	Mon 15/06/15

Figura 3: Diagrama de Gantt definitivo realizado en Microsoft Project.

Como se puede ver en este diagrama de Gantt han sido añadidas las fases nuevas correspondientes. Además se han reajustado las duraciones para coincidir con la duración real del proyecto.

3.7.1 Cambios en la duración

Aquí podemos observar las diferencias que se han generado en cantidad de horas respecto la planificación inicial.

Planificación en días		Inicial	Real	Desviación	Desviación
	Análisis Inicial	60	60	0	
	Contexto Inicial	28	28	0	
Fase 1	**Subtotal Fase 1**	**88**	**88**	**0**	**0,00%**
	Adaptación al entorno		40	-40	
	Carga de datos volcados	90	64	26	
	GEP	94	76	18	
	Procesado de datos volcados	52	72	-20	
	Evaluación con datos volcados	36	12	24	
Fase 2	**Subtotal Fase 2**	**272**	**264**	**8**	**2,94%**
	Informe Seguimiento		32	-32	
	Carga de datos mediante Sqoo	48	16	32	
	Procesado de datos mediante S	32	4	28	
	Evaluación de datos de Sqoop	32	24	8	
Fase 3	**Subtotal Fase 3**	**112**	**76**	**36**	**32,14%**
	Conclusiones	40	84	-44	
Fase 4	**Subtotal Fase 4**	**40**	**84**	**-44**	**-110,00%**
Total		**512**	**512**	**0**	**0,00%**

Tabla 1: Cambios en la duración

Como se puede observar en la tabla anterior han surgido diferencias entre las horas asignadas en la planificación inicial y las realizadas finalmente. Se puede observar el efecto de la reutilización del código de la Fase 2 para la realización de la Fase 3, en la disminución del coste de tiempo finalmente utilizado en las tareas que la componen.

Además se puede observar como la tarea de Conclusiones de la Fase 4 ha sido aumentada en su tamaño con el objetivo de cubrir el tiempo asignado al proyecto, así aumentando la documentación y análisis del mismo.

4 Gestión económica

Es necesario realizar un estudio sobre los costes que se han de asumir en la realización del proyecto. Debido a cierta protección de detalles internos de la compañía donde se realiza el proyecto se ha decidido realizar un análisis de costes de la realización de este proyecto en una compañía cualquiera. Gracias a este estudio se podrá gestionar el coste del proyecto y evitar que aumente sin control. Para ello es necesario recopilar y considerar una serie de costes.

4.1 Análisis de costes

Los costes han sido calculados a partir de las siguientes remuneraciones, extraídas de un documento actual sobre las mismas [13]:

	Anual	Meses en un año	Mensual	Días en un mes	Diario	Horas en un día	Hora
Jefe de proyecto	40.000,00 €	12	3.333,33 €	20	166,67 €	8	20,83 €
Ingeniero Software	33.000,00 €	12	2.750,00 €	20	137,50 €	8	17,19 €
Desarrollador	21.000,00 €	12	1.750,00 €	20	87,50 €	8	10,94 €

Tabla 2: Remuneraciones de los diferentes perfiles.

4.1.1 Directos

Los costes directos por actividad son aquellos costes los cuales provienen de la realización del proyecto, tales costes incluyen los recursos humanos que se han utilizado para llevar a cabo el proyecto. Se ha considerado la duración del proyecto como el tiempo que dedica el alumno a la realización del proyecto en la compañía. Además, se ha tenido en cuenta una estimación de una hora semanal dedicada por el director del TFG al proyecto. Estos cálculos han sido estimados gracias a las fases incluidas en el Gantt del proyecto, por tanto calculamos los costes directamente atribuibles a las diferentes actividades del Gantt.

Concepto	Días	Horas	Rol	Coste Horas	Coste
TFG	**128**	**539**	**XXXXX**	**XXXXX**	**6.963 €**
Fase 1	22	88	Ingeniero Software	17,19 €	1.513 €
Análisis inicial	15	60	Ingeniero Software	17,19 €	1.031 €
Contexto inicial	7	28	Ingeniero Software	17,19 €	481 €
Fase 2	68	272	Desarrollador	10,94 €	2.975 €
Carga de datos volcados	40	90	Desarrollador	10,94 €	984 €
GEP	26	94	Desarrollador	10,94 €	1.028 €
Procesado de datos volcados	13	52	Desarrollador	10,94 €	569 €
Evaluación con datos volcados	9	36	Desarrollador	10,94 €	394 €
Fase 3	28	112	Desarrollador	10,94 €	1.225 €
Carga de datos mediante Sqoop	12	48	Desarrollador	10,94 €	525 €
Procesado de datos de Sqoop	8	32	Desarrollador	10,94 €	350 €
Evaluación de datos de Sqoop	8	32	Desarrollador	10,94 €	350 €
Fase 4	10	40	Ingeniero Software	17,19 €	688 €
Conclusiones	10	40	Ingeniero Software	17,19 €	688 €
Gestión del Proyecto	27	27	Jefe de proyecto	20,83 €	563 €

Tabla 3: Costes directos del proyecto

4.1.2 Indirectos

Estos son los costes derivados de los materiales y/o dispositivos utilizados para llevar a cabo el proyecto o que dan soporte a este.

Concepto	Meses	Años	Coste total	Coste Mensual	Coste Parcial
Recursos de everis (internet, luz, etc.)	6			260,00 €	1.560,00 €
Portátil	6	4	1200	25,00 €	150,00 €
Alquiler Servidor	2			300,00 €	600,00 €
Suma	**14**			**585,00 €**	**2.310,00 €**

Tabla 4: Costes indirectos del proyecto

4.1.3 Contingencias

Aquí vemos una cantidad de contingencia por si surgiese algún problema, siendo esta del 15% de los costes directos más los costes indirectos.

Concepto	Meses
Directos	6.962,50 €
Indirectos	2.310,00 €
Suma Parcial	**9.272,50 €**
15%	**1.390,88 €**

Tabla 5: Presupuesto de contingencia

4.1.4 Imprevistos

Cuantía que supondría el realizar durante un retardo de 7 días más el proyecto.

Concepto	Probabilidad	Coste Semana Directos + Indirectos
Retardo 7 días	10%	2.318,13 €
Suma	**10%**	**2.318,13 €**

Tabla 6: Presupuesto para imprevistos

4.1.5 Presupuesto

Suma del total de los diferentes componentes que estiman el coste total de la realización del proyecto y su realización.

Concepto	Coste
Directos	6.962,50 €
Indirectos	2.310,00 €
Contingencias	1.390,88 €
Imprevistos	2.318,13 €
Suma	**12.981,50 €**

Tabla 7: Presupuesto definitivo

4.2 Control de gestión

La realización del proyecto se llevará a cabo según lo mencionado en los apartados anteriores. En el caso de que surgiera algún imprevisto que supusiera una diferencia en alguna de las estimaciones realizadas anteriormente, se procederá a tomar constancia del tiempo que ha durado, y de los recursos que se han utilizado para ello. Este imprevisto supondría una desviación respecto a la planificación inicial, tanto a nivel de tiempo como a nivel de costes, es por ello que se deberá analizar si este imprevisto es causado debido a causas internas del proyecto o ajenas al mismo.

4.3 Coste real

El coste real del proyecto ha sido calculado utilizando las mismas remuneraciones de los perfiles que los vistos en el apartado anterior.

4.3.1 Directos

Los costes directos han sido recalculados a partir de los datos de tiempo de real que se han utilizado en la realización del proyecto. Los costes directos sufren cambios debido al tiempo dedicado a cada una de las tareas, junto con una tarea dedicada a la relación de la fase de seguimiento, la cual no se había tenido en consideración en la planificación inicial.

	Concepto	Días	Horas	Rol	Coste Horas	Coste
TFG		**128**	**539**	**XXXXX**	**XXXXX**	**7.237,50 €**
Fase 1		22	88	Ingeniero Software	17,19 €	1.512,50 €
	Análisis inicial	15	60	Ingeniero Software	17,19 €	1.031,25 €
	Contexto inicial	7	28	Ingeniero Software	17,19 €	481,25 €
Fase 2		66	264	Desarrollador	10,94 €	2.887,50 €
	Adaptación al entorno	20	40	Desarrollador	10,94 €	437,50 €
	Carga de datos volcados	40	64	Desarrollador	10,94 €	700,00 €
	GEP	25	76	Desarrollador	10,94 €	831,25 €
	Procesado de datos volcados	18	72	Desarrollador	10,94 €	787,50 €
	Evaluación con datos volcados	3	12	Desarrollador	10,94 €	131,25 €
Fase 3		19	76	Desarrollador	10,94 €	831,25 €
	Informe de seguimiento	8	32	Desarrollador	10,94 €	87,50 €
	Carga de datos mediante Sqoop	4	16	Desarrollador	10,94 €	175,00 €
	Procesado de datos de Sqoop	1	4	Desarrollador	10,94 €	43,75 €
	Evaluación de datos de Sqoop	6	24	Desarrollador	10,94 €	262,50 €
Fase 4		21	84	Ingeniero Software	17,19 €	1.443,75 €
	Conclusiones	21	84	Ingeniero Software	17,19 €	1.443,75 €
Gestión del Proyecto		27	27	Jefe de proyecto	20,83 €	562,50 €

Tabla 8: Costes directos del proyecto real

4.3.2 Indirectos

Los costes indirectos se han visto reducidos debido a no haber tenido acceso a un servidor como se detalló en el análisis inicial. Por ello los costes indirectos quedan tal que:

Concepto	Meses	Años	Coste total	Coste Mensual	Coste Parcial
Recursos de everis (internet, luz, etc.)	6			260,00 €	1.560,00 €
Portátil	6	4	1200	25,00 €	150,00 €
Suma	**12**			**285,00 €**	**1.710,00 €**

Tabla 9: Costes indirectos del proyecto real

4.3.3 Presupuesto

Real	
Concepto	**Coste**
Directos	7.237,50 €
Indirectos	1.710,00 €
Contingencias	1.390,88 €
Imprevistos	2.318,13 €
Suma	**12.656,50 €**

Tabla 10: Presupuesto del proyecto real

Diferencia		
Concepto		**Coste**
Directos		-275,00 €
Indirectos		600,00 €
Contingencias		0,00 €
Imprevistos		0,00 €
Suma		**325,00 €**

Tabla 11: Diferencia de presupuestos

En las tablas anteriores podemos observar como aun habiendo sufrido un incremento en los costes directos, el coste total del proyecto se ha visto reducido. Esto es así debido a que la reducción de los costes indirectos es suficientemente amplia como para compensarlo. Estas modificaciones no han obligado a utilizar ninguna parte del presupuesto de contingencias ni de imprevistos, por ellas estas partes se mantienen intactas.

5 Solución original

La solución *ETL* original es un conjunto de diversos procesos *ETL* de diferentes fuentes. Debido a motivos de privacidad de los datos contenidos no ha sido posible acceder a todas ellas, por ello se ha decidido centrarse en un única tabla a la cual si se ha permitido tener acceso. De esta manera se pretende analizar la migración de uno de los procesos ETL y generalizar las conclusiones obtenidas, permitiendo decidir si realizar posteriores migraciones del resto de proceso a herramientas de Big Data.

5.1 Selección del proceso a migrar

Para seleccionar el proceso el cual iba a ser migrado a Hadoop, se realizó un volcado de la tabla que contiene los tiempos de procesado de los diferentes procesos que conforman el ETL original completo, ordenando según los minutos que tarda en ejecutarse en orden descendiente.

id_proce…	fc_inicio_ejecucion	fc_mensaje	de_mensaje	minutos
XXX	2015-01-01 XX:XX:XX.XXX	2015-01-01 XX:XX:XX.XXX	----------	13
XXX	2015-01-01 XX:XX:XX.XXX	2015-01-01 XX:XX:XX.XXX	----------	12
XXX	2015-01-01 XX:XX:XX.XXX	2015-01-01 XX:XX:XX.XXX	----------	9
XXX	2015-01-01 XX:XX:XX.XXX	2015-01-01 XX:XX:XX.XXX	----------	9
XXX	2015-01-01 XX:XX:XX.XXX	2015-01-01 XX:XX:XX.XXX	----------	7
XXX	2015-01-01 XX:XX:XX.XXX	2015-01-01 XX:XX:XX.XXX	----------	7
XXX	2015-01-01 XX:XX:XX.XXX	2015-01-01 XX:XX:XX.XXX	----------	4
XXX	2015-01-01 XX:XX:XX.XXX	2015-01-01 XX:XX:XX.XXX	----------	4
234	**2015-01-05 06:49:15.020**	**2015-01-05 06:53:15.873**	**p_carga_tbl_maestro_factores_stamp**	**4**
XXX	2015-01-01 XX:XX:XX.XXX	2015-01-01 XX:XX:XX.XXX	----------	4

Tabla 12: Procesos del ETL original

Debido a que mucha de la información en estos procesos es confidencial, se procedió a seleccionar el único del cual se muestra el nombre en la tabla, siendo este uno de los procesos que tardan más tiempo de entre todos los procesos que se realizan en el *ETL*. Una vez decidido el proceso que se migraría, se anotaron los datos de este para su posterior comparación.

5.2 Características del proceso

El proceso seleccionado necesita como fuente de información una serie de datos, los cuales habría que generar a partir de una serie de datos de origen (mediante el uso de otros procesos relacionados).

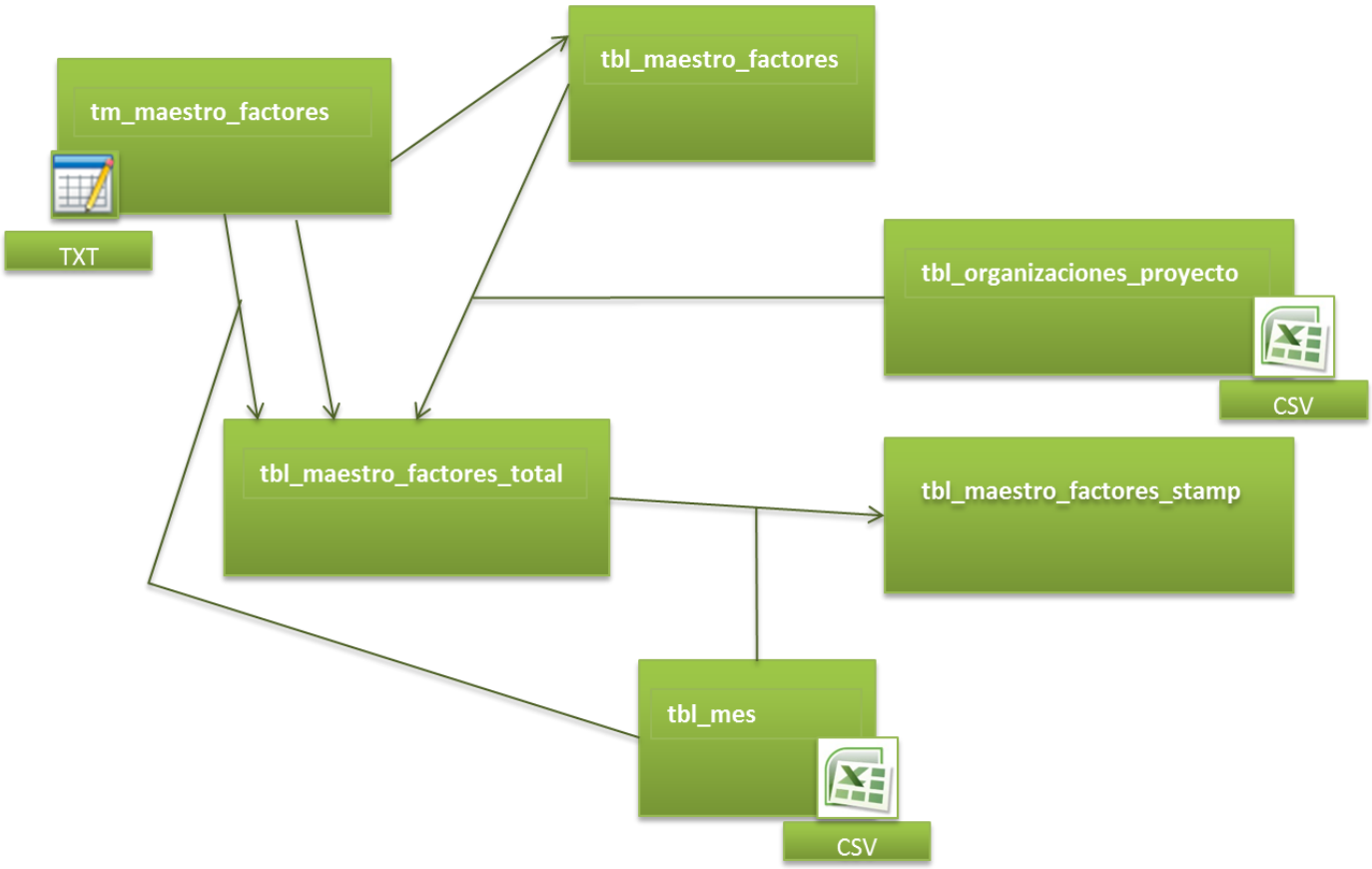


Figura 4: Dependencias de los datos

Para realizar el proceso ETL es necesario:

- Cargar los datos de tbl_maestro_factores mediante el proceso p_carga_tbl_maestro_factores.
  - Los datos son obtenidos desde tm_maestro_factores.
- Cargarlos datos de tbl_maestro_factores_total mediante el proceso p_carga_tbl_maestro_factores_total.
  - Los datos son obtenidos desde tm_maestro_factores, tbl_maestro_factores, tbl_orgnizaciones_proyecto y tbl_mes.
- Cargar los datos de tbl_maestro_factores_stamp mediante el proceso p_carga_tbl_maestro_factores_total
  - Los datos son obtenidos desde tbl_maestro_factores_total.

Para poder realizar los procesos, los datos contenidos en las siguientes tablas deben haber sido obtenidos mediante algún método:

- tbl_organizaciones_proyecto
- tbl_mes
- tm_maestro_factores

A continuación se listan las tablas iniciales y final del proceso, junto con el número de filas de que disponen al finalizar la ejecución.

Nombre de la tabla	Nº Filas
tbl_organizaciones_proyecto	339
tbl_mes	365
tm_maestro_factores	29253
tbl_maestro_factores_stamp	2664200

Tabla 13: Listado de tablas del proceso

Gracias a la tabla anterior podemos percatarnos de que el número de filas de tbl_maestro_factores_stamp es bastante superior al número de filas de las tablas iniciales.

6 Requisitos

En este proyecto existe un alto grado de margen a la hora de su realización ya que las características de la solución a crear disponen de suficiente margen para realizar cambios respecto al original. Debido a la carencia de información sobre el funcionamiento del proceso ETL original, además se va a realizar mediante el uso de herramientas diferentes a las del sistema original; el listado de requisitos no es un listado demasiado amplio ni exhaustivo.

6.1 Funcionales

Los requisitos funcionales son aquellos que definen las funcionalidades que debe ser capaz de realizar la solución.

- Los datos utilizados por la migración del proceso *ETL* utilizando los volcados de datos deben cargarse exactamente con el formato de salida del sistema original, sin posibilidad de recibir tratamiento externo antes de la carga de los datos.
- La solución debe alertar en caso de surgir algún error.
- La solución debe a partir de los datos entregados, generar los datos necesarios para la realización del proceso a migrar.
- La migración realizada debe ser capaz de obtener los mismos datos resultantes que la solución original.

6.2 No funcionales

Los requisitos no funcionales son aquellos que definen las restricciones sobre la solución a desarrollar.

- La migración del proceso *ETL* debe ser una solución automatizada y adaptable.
- Se ha de realizar el proyecto sin interferir en la base de datos oficial de la compañía.
- La solución debe tener un coste en tiempo menor respecto al tiempo del sistema original.
- La ejecución de la solución obtenida debe ser transparente a nivel del usuario.
- La solución obtenida debe permitir cierta reutilización de sus componentes.
- Se debe mantener la máxima similitud posible con el proceso original.

7 Diseño lógico

7.1 Arquitectura

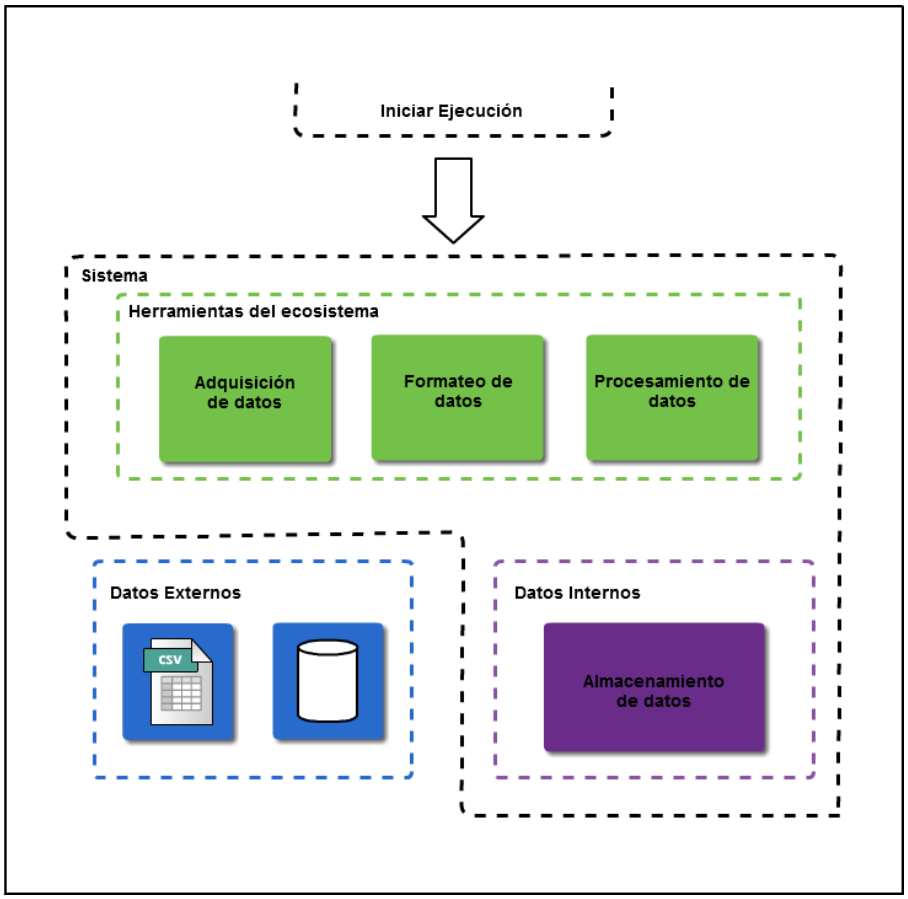


Figura 5: Arquitectura de la solución

En la figura anterior podemos observar la arquitectura de la solución que se pretende desarrollar, se pueden observar dos capas. Una primera capa con las herramientas del ecosistema que se utilizarán para realizar las fases del proceso *ETL*. Una segunda capa con los datos, dónde podemos observar los datos internos del sistema y los datos externos al sistema a los cuáles habrá que acceder.

7.2 Actores y diagrama de casos de uso

En la siguiente figura podemos ver como únicamente existe una interacción con el sistema, debido a que uno de los requisitos de la solución es que debe ser automatizada. Por ello la solución se trata de un proceso *batch* (o procesamiento por lotes), la principal característica de los mismos es la ejecución sin el control ni supervisión por parte del usuario. Los procesos *batch* se utilizan en tareas repetitivas para así reducir la posibilidad de generar errores, estos suelen estar conformados de uno o más *scripts* transparentes desde el punto de vista del usuario. [14]

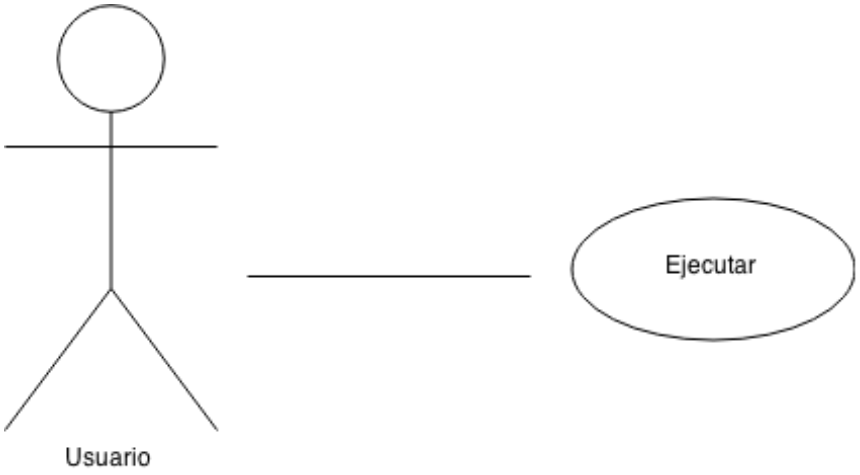


Figura 6: Muestra los casos de uso del sistema

7.3 Diagrama de estados

Debido a las características de los datos de origen, se ha optado por utilizar una variante de los procesos *ETL*, los procesos ELT. La principal diferencia entre un proceso *ETL* y lo mostrado en este proyecto es la fase de carga existente entre la extracción y la transformación, en la cual se preparan los datos para ser posible almacenarlos y tratarlos dentro del sistema.

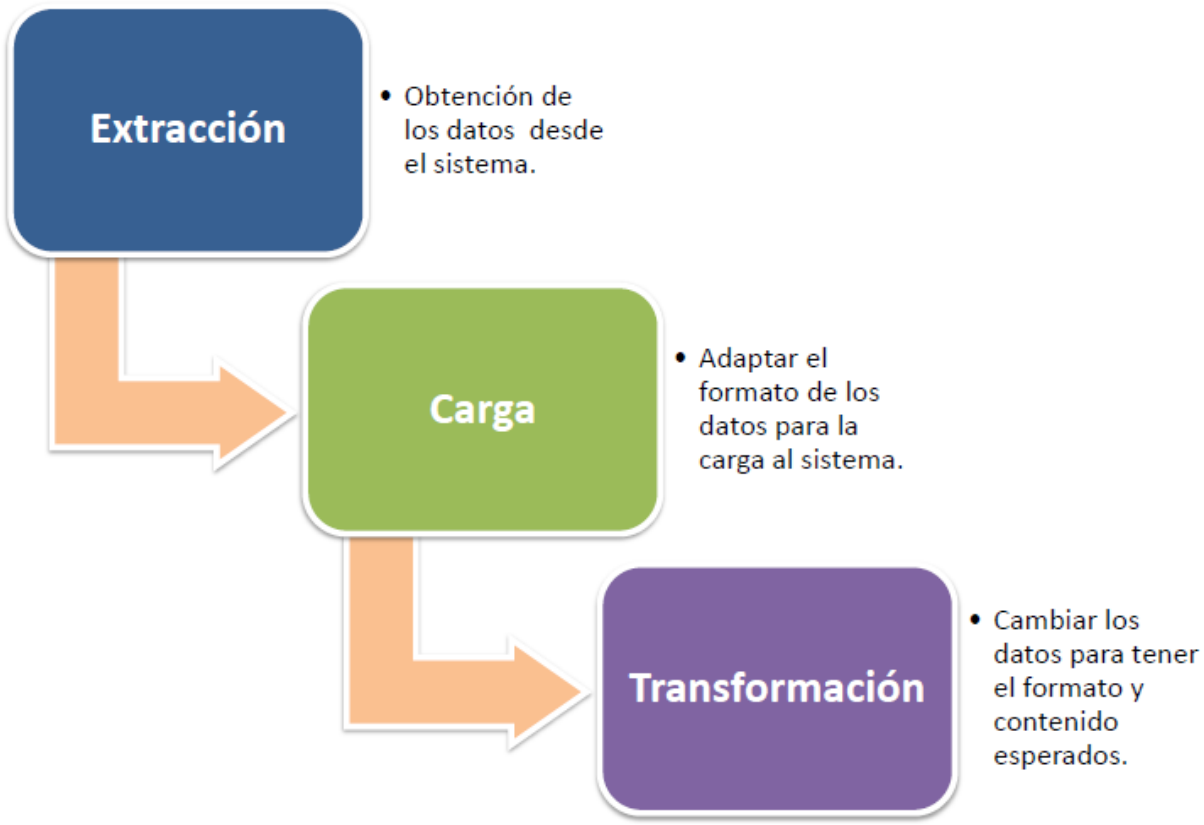


Figura 7: Fases de un proceso ELT

A continuación podemos ver el diagrama de estados a alto nivel del sistema.

Como se puede observar existe un inicio centralizado, el cual se encarga de ir realizando las llamadas necesarias para realizar el proceso que se está migrando. Todas las partes del sistema podrán finalizar en cualquier momento en caso de encontrarse un error.

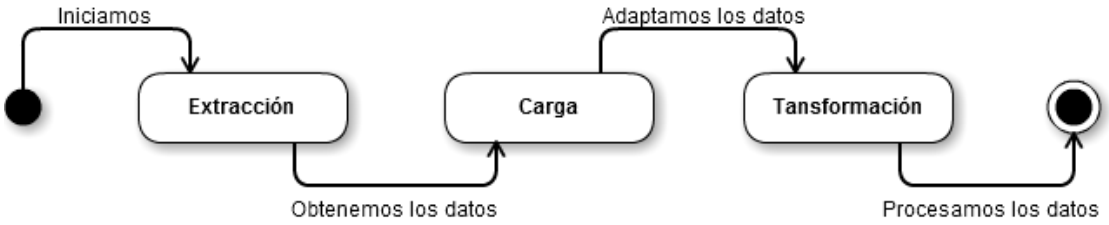


Figura 8: Diagrama de estados de alto nivel

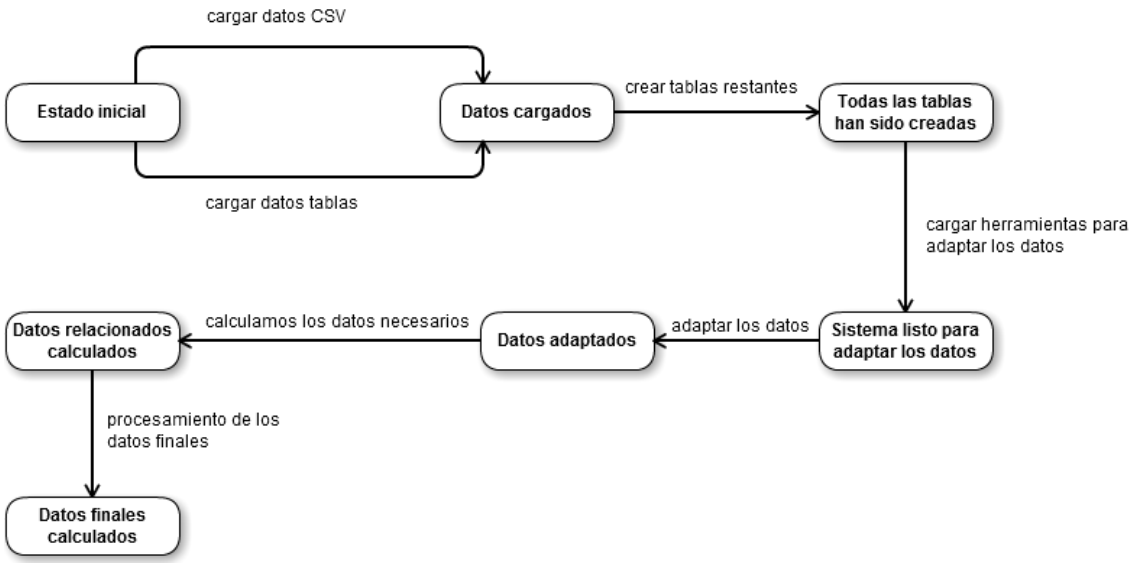


Figura 9: Diagrama de estados de bajo nivel

En la figura anterior podemos observar los diferentes estados que se producen al llevarse a cabo la solución que se ha planteado para realizar la migración. Podemos observar como el procesamiento de los datos finales ha sido separado del de los datos necesarios para facilitar la futura evaluación del mismo respecto al del sistema original.

7.4 Diagrama de secuencia

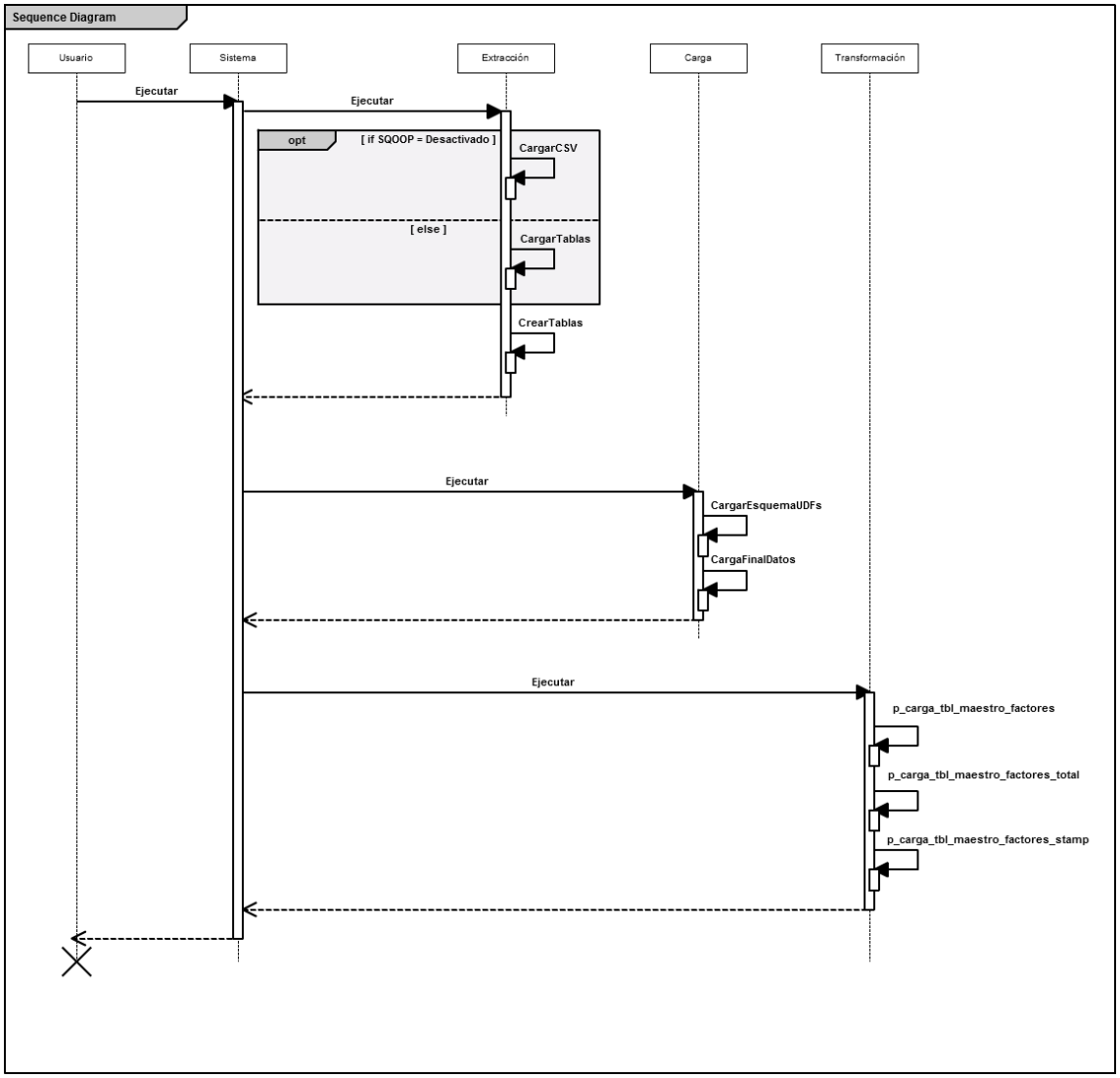


Figura 10: Diagrama de secuencia

En la anterior figura podemos observar el diagrama principal de secuencia del sistema donde se puede observar cómo no precisa de interacción por parte del usuario durante la ejecución.

En el caso de que fuese necesario automatizar la ejecución del proceso, este podría ser llamado mediante otro proceso, o mediante algún programa como cron disponible en los sistemas GNU/Linux.

8 Solución generada

8.1 Decisiones tecnológicas

8.1.1 Entorno de trabajo

VirtualBox

Inicialmente se tenía pensado realizar el proyecto utilizando una imagen destinada a VirtualBox, ya que su licencia era mucho más abierta para el proyecto. Se decidió finalmente no utilizar la herramienta debido a que existía otra herramienta en la cual la imagen virtual tenía mayor fluidez.

VMWare

Inicialmente este producto fue descartado por la licencia comercial que utiliza, pero una vez se hubieron realizado los primeros intentos mediante alguna de las otras herramientas se vio que esta era la que ofrecía mejor fluidez en el uso.

Hortonworks

Se desestimó la utilización de Hortonworks debido al pequeño número de tutoriales existentes comparándolo con Cloudera.

Versión de Cloudera

Se utilizó una imagen, configurada para ser utilizada en un único nodo, de la versión de Cloudera 4.7.0 por encima de la actual versión 5.3.0 debido al rendimiento que se obtuvo con esta última, ralentizando la máquina de trabajo llegando a imposibilitar realizar ningún tipo de tarea.

8.1.2 Ecosistema

HDFS

Se optó por HDFS para la ingestión de los datos por disponer su similitud respecto a un sistema de ficheros de Linux. Además, HDFS si se utiliza en un sistema con múltiples nodos permite replicar la información en ellos facilitando el acceso a los datos y su disponibilidad.

MapReduce

Una vez se ha decidido utilizar Hadoop es usual utilizar, de manera directa o indirecta; el modelo de programación MapReduce. Simplificadamente MapReduce funciona generando un conjunto de *Mappers* y otro de *Reducers*. Los *Mappers* se encargan de generar, a partir de los datos en HDFS; una lista de claves-valor. Los *Reducers* se encargan de a partir de una lista de claves-valor generar una lista de valores.

Hive

Se decidió utilizar Hive debido a que las *User-Defined Functions* (*UDFs*) únicamente funcionan mediante Hive, ya que es en este sistema donde se agregan las funcionalidades incorporadas.

Hive permite la traducción automática de consultas en su lenguaje HiveQL, muy similar al *SQL*; a conjuntos de *Mappers* y *Reducers*.

Spark

No se realizó el proyecto dentro de la herramienta de Spark ya que una de las principales características de Spark la realización de procesos en memoria RAM. Este proyecto debía comenzarse a realizarse en la máquina de trabajo del usuario, la cual no disponía de una gran cantidad de memoria RAM, por lo que decidió utilizar otras opciones.

Sqoop

Debido a las características de los datos de entrada se decidió que se debía probar alguna alternativa diferente respecto a la carga de los datos. Por ello se investigó el tema y se vio que existían alternativas a utilizar volcados de datos, accediendo estas alternativas directamente a la fuente de los datos. La principal ventaja de estas era su mayor simpleza a la hora de cargar los datos en el sistema. Se escogió Sqoop ya que venía incluido en la imagen de Cloudera proporcionada, junto con los drivers para Sqoop tanto de MySQL como de PostgreSQL.

Debido a las características del proceso que se está migrando no se obtendría un beneficio de utilizar alguno de los siguientes gestores NoSQL (los más populares):

HBase

Esta herramienta tiene la virtud de ser capaz de responder de manera rápida a las consultas sobre rangos. Esto es debido a que incluye de manera interna un índice B+ el cual le permite de manera rápida acceder a la información que se está buscando evitando realizar una lectura completa de los datos.

Cassandra

Por otra parte existen herramientas que implementan otros tipos de mecanismos, los cuales permiten que sabiendo el dato que necesitamos buscar ser capaces de encontrarlo de forma muy eficiente. El *consistent Hashing*, permite acceder a la información relacionada de manera rápida y prácticamente directa.

8.2 Esquemas

Debido al hecho que no existe un gran conocimiento sobre las herramientas utilizadas para este proyecto, se procederá a explicar la interacción existente entre cada una de ellas a fin de facilitar la comprensión del proyecto. Inicialmente desde un punto de vista de capas, y de manera posterior desde las herramientas en concreto utilizadas.

8.2.1 Esquema de interacción

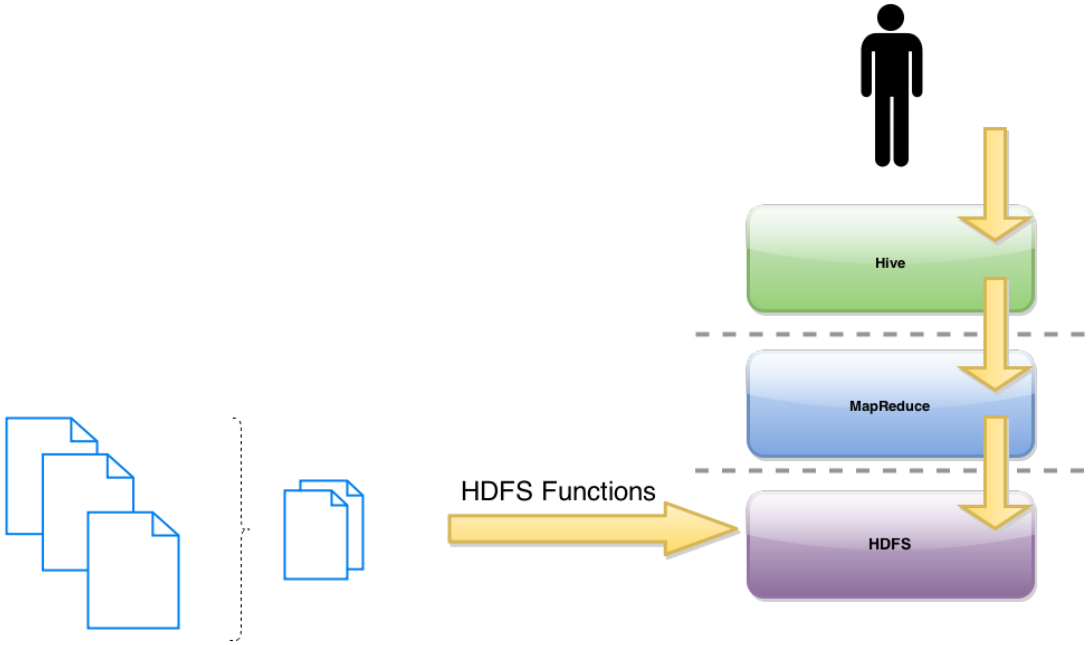


Figura 11: Esquema de la solución mediante *CSVs*[1]

En la figura anterior somos capaces de observar un resumen de la solución creada mediante los ficheros del volcado de datos en formato *CSV*. Como se puede ver se cargan los datos desde los ficheros de origen al sistema de ficheros, HDFS o *Hadoop Distributed File System*; para ello utilizando las funciones disponibles en el propio sistema de ficheros.

De manera posterior a la Extracción de los datos, también conocido en *Big Data* como ingestión; se procede por motivos de formato a cargar los datos en unas tablas adaptando los datos disponibles al formato adecuado de las tablas. Esta carga de los datos a sus tablas definitivas se realiza mediante funciones nativas de Hive y alguna función expresamente desarrollada.

Finalmente se realiza también mediante Hive una serie de *querys*, las cuales son traducidas a un conjunto de *Mappers y Reducers*; para así las transformaciones necesarias de los datos obteniendo los resultados esperados del proceso *ETL*.

[1] Diagrama realizado mediante Draw.io

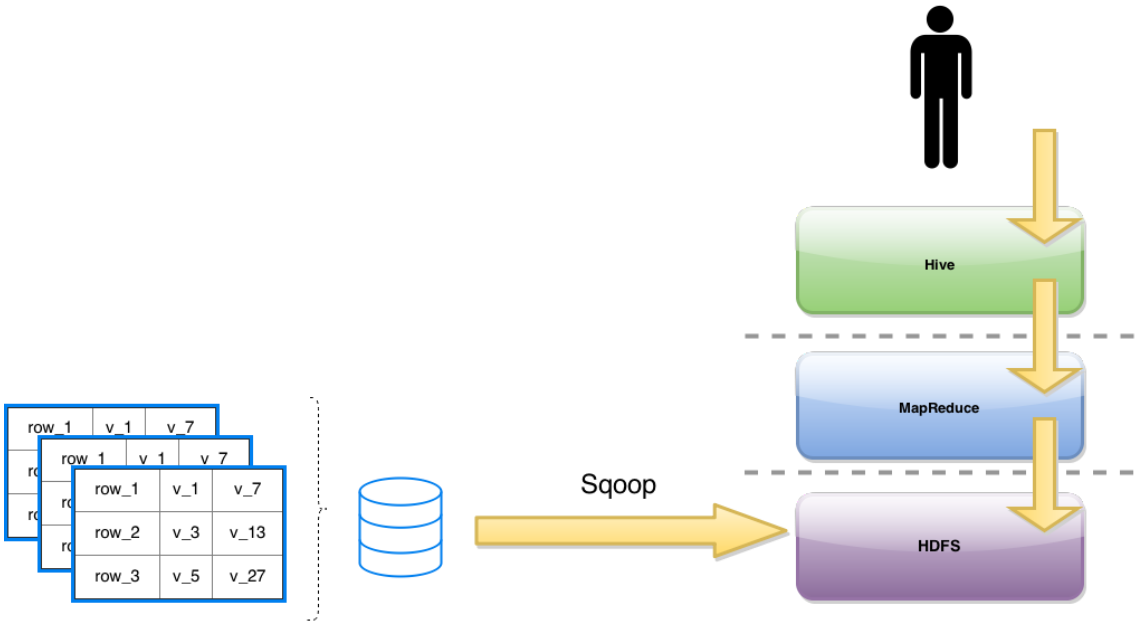


Figura 12: Esquema de la solución utilizando Sqoop[2]

En esta última figura podemos apreciar algunas diferencias.

Partimos de la base de que disponemos de una base de datos relacional, con las tablas que nos son necesarias para llevar a cabo nuestro proceso *ETL*. Mediante la herramienta Sqoop, se realiza la ingestión de los datos desde estas tablas a ficheros dentro del sistema de HDFS.

Los pasos que le siguen son idénticos a los del anterior, ya que también se deben adaptar los datos de estos ficheros al formato adecuado aun habiendo sido extraídos directamente de una base de datos.

[2] Diagrama realizado mediante Draw.io

8.2.2 Diseño técnico

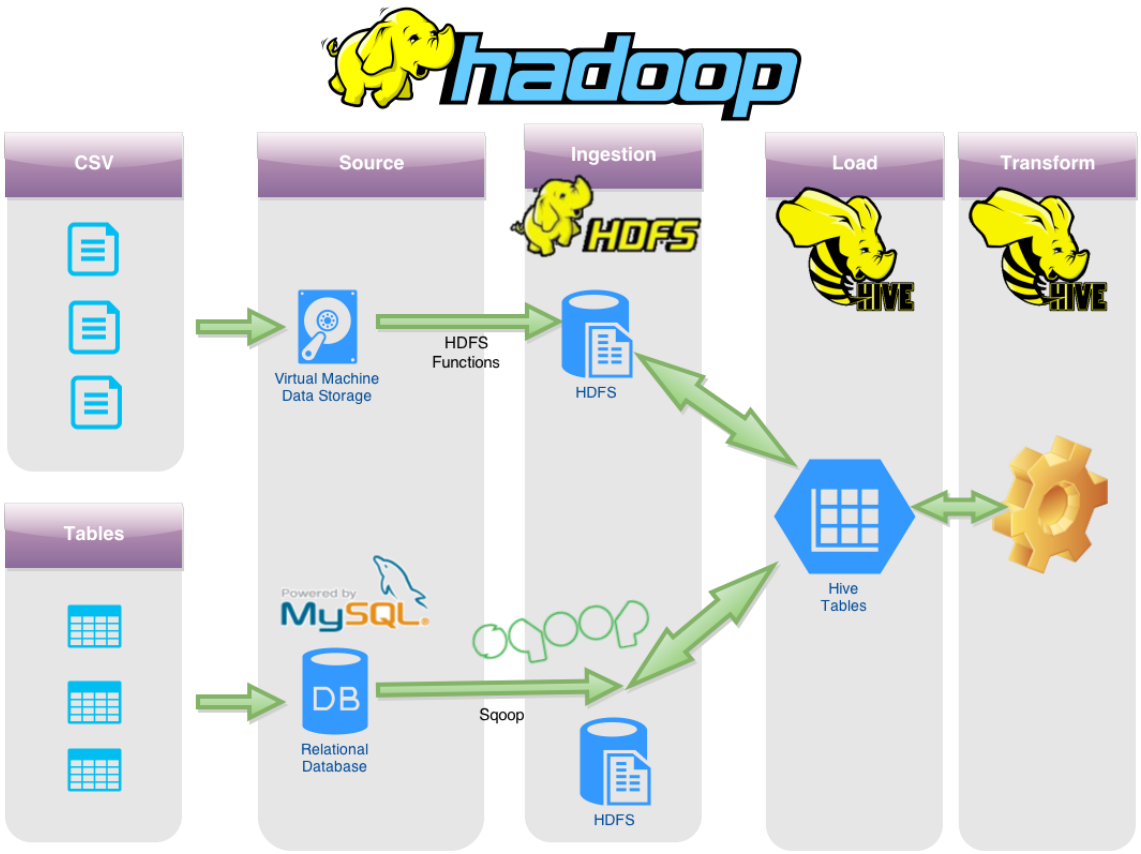


Figura 13: Esquema de las herramientas utilizadas siguiendo ELT[3][4][5][6][7]

En la figura anterior somos capaces de apreciar, de manera gráfica; la relación existente entre los dos esquemas previamente explicados, respecto a las herramientas utilizadas para llevar a cabo la migración del proceso.

Además, somos capaces de apreciar que debido a la migración hemos pasado de tener un proceso *ETL*, a tener un proceso *ELT*. Con una fase de *Load* debido al formato de algunos de los datos y su incompatibilidad con los tipos del sistema migrado.

[3] Diagrama realizado mediante Draw.io
[4] El logotipo de Hadoop ha sido extraído de la Wikipedia
[5] El logotipo de HDFS es una modificación del disponible en la página web de Hortonworks
[6] El logotipo de MySQL ha sido obtenido de su web, escogiendo el más adecuado según las propias licencias.
[7] El logotipo de Hive es una adaptación del disponible en su propia página web oficial.

8.3 Extracción de los datos

8.3.1 Fuentes de datos

Dos de las fuentes de datos disponían de un formato no apto para ser cargado directamente en el sistema. Por ello se hubo de realizar la extracción de los datos en dos pasos.

Un primer paso, donde las tablas son colocadas en el sistema de ficheros de Hadoop, HDFS. Después de esto se procede a la generación de unas tablas de datos no definitivas, las cuáles permitirán acceder a los datos de los ficheros, indicando el delimitador de los campos; que previamente habíamos cargado.

Un segundo paso dónde accediendo a las tablas que hemos creado, generamos una serie de modificaciones que permitirán a los datos obtener el formato necesario para poder ser tratados.

8.3.2 Tablas de datos

Debido a que los datos se encuentran en una base de datos relacional, el propio Sqoop es capaz de obtener los tipos y nombres de cada una de las columnas de las tablas que hemos creado. Aun así permite obtener los datos mediante la realización de una consulta, por tanto permite redefinir los tipos, las columnas al igual que obtener los datos deseados únicamente.

En este apartado también existe un segundo paso de transformación, donde se adaptan algunos de los datos de manera semejante a la anterior, pero con alguna simplificación debido a la ausencia de delimitadores.

8.4 Procesado de los datos

Los datos en ambas fases fueron procesados de la misma manera, ya que después de la carga de los datos al formato correcto los datos almacenados son idénticos. Dado que los datos son idénticos esto permite utilizar la misma solución para el procesado de datos para ambas fases, reduciendo el tiempo de la segunda fase.

El procesado de los datos se realizó adaptando los *scripts* del proceso *ETL* original, para poder así ser lanzado en el nuevo sistema. Para ello hubo que adaptar el lenguaje *SQL* de la solución anterior para ser posible utilizarlo mediante Hive.

Para adaptar los diferentes *scripts* hubo que realizar una serie de modificaciones:

- Adaptar todas las conversiones a otros tipos que se realizaban
  - Hubo que asegurar que al almacenarse los datos en las tablas, la conversión automática no dañaba los datos almacenados.
- Crear un *UDF* para poder generar un identificador específico que se genera en tiempo real a partir de los datos.
- Convertir ciertos valores que dependían de la fecha en valores estáticos para ser posible comparar los datos y asegurar su rigor.
- Convertir una *subquery* dentro de un *where*, en un *left outer join*.[8]

Antes de haber realizado algunas de las modificaciones anteriores de manera adecuada, los datos que se obtenían del proceso *ETL* eran completa o parcialmente erróneos, por tanto teniendo que intuir dónde posiblemente existía el error para solucionarlo.

8.5 Validación de los datos

La validación de los datos fue un proceso realizado de manera incremental. Inicialmente se comprobaban una serie de valores de los datos, los cuales de manera posterior fueron incrementándose hasta alcanzar un conjunto significativo. Esto fue así debido a que el volcado de la solución que se recibió contenía una serie de formatos en los elementos distintos al formato con el que se obtenían los datos obtenidos desde la solución.

Durante la validación se fue comprobando que elementos que debían encontrarse en la solución no aparecían, y que elementos había en la solución que no debían encontrarse en la solución. Se iba ejecutando tanto el proceso como los procesos relacionados de manera parcial, realizando volcados de las tablas, de manera que se podía observar el contenido de estas en busca de los elementos erróneos o faltantes.

Para realizar la validación, se generaron diferentes scripts, los cuales a partir de los datos obtenidos y los originales, se generaban unos conjuntos de datos comparables. Estos conjuntos eran comprobados entre ellos a fin de detectar las diferencias, determinar el número de las mismas, y calcular el porcentaje de diferencias respecto al resultado que se esperaba obtener. Además se extraía de los ficheros originales los datos de los cuáles se sospechaba que tenían relación.

[8] Más información en el apartado de herramientas dentro del tema sobre limitaciones.

Así una vez el subconjunto analizado coincidía completamente con el subconjunto de la solución, se aumentaba este subconjunto. Una vez no fue posible aumentar este subconjunto debido al formato, se procedió a contar el número de filas resultantes para comprobar que el número coincidía con el de la solución.

8.6 Evaluación de los datos

Los datos obtenidos después de la ejecución del procesado de los datos, han sido revisados para asegurar su rigurosidad tal y como se especificó en el apartado de metodología. Para realizar las comprobaciones fue necesario solicitar de nuevo un volcado de datos, ya que el volcado de datos inicial era insuficiente para asegurar la rigurosidad del conjunto de los datos.

Para realizar las comprobaciones fueron creados una serie de *scripts*, los cuales permitían teniendo el volcado de datos original, y el volcado de datos conseguido, comprobar las diferencias entre lo obtenido y lo que se esperaba. Además estos *scripts*, generaban un fichero con las filas que faltaban por generar, y otro con las filas las cuales no debían encontrarse entre los datos obtenidos.

Durante la realización de este proyecto se han comprobado además una serie de factores, de los cuales en buena parte determinan la solución final, y por tanto las conclusiones de este proyecto.

9 Análisis

En este apartado se expondrán los diferentes análisis realizados sobre la solución obtenida al realizar la migración del proceso ETL seleccionado.

9.1 Tiempo mínimo

Uno de los análisis que se llevaron a cabo fue analizar cuanto era el tiempo mínimo que tardaba el sistema en responder la *query* más simple posible.

```
SELECT '0' AS x FROM tabla LIMIT 1;
```

Tabla 14: *Query* más simple posible

La realización de una *query* mediante Hive, implica la realización de una serie de pasos: traducción del HiveQL a *Mappers y Reducers,* análisis del volumen de datos para determinar el número de *Mappers y Reducers* a utilizar, y la ejecución de los *Mappers y Reducers.*

Debido a todo esto existe una parte inicial de tiempo, la cual no tiene una relación directa con la complejidad de la *query*, es por ello que limita a un mínimo el tiempo de respuesta de cualquier *query*.

A continuación podemos ver la tabla con los datos resultantes de lanzar esta *query* mínima.

segundos	nºiteración	tiempo-inicio	tiempo-fin	nºfilas
25	1	1430130534	1430130559	1
26	2	1430130679	1430130705	1
27	3	1430130885	1430130912	1
25	4	1430130972	1430130997	1
24	5	1430131117	1430131141	1
27	6	1430131321	1430131348	1
23	7	1430131468	1430131491	1
27	8	1430131671	1430131698	1
24	9	1430131758	1430131782	1
25	10	1430131902	1430131927	1

Máximo	Promedio	Mínimo
27	25,3	23

Suma total	253
Nº valores total	10
Valor Máximo	27
Valor Mínimo	23
Suma valores aceptados	203
Nº valores aceptados	8
Tiempo query mínima	**25,38**

Tabla 15: Datos sobre la *query* mínima

Como podemos ver, no existe ninguna relación entre el número de iteración y el tiempo de ejecución. Además podemos considerar según los datos observados que el tiempo mínimo es de aproximadamente 26 segundos.

Debido a que el tiempo mínimo es de 26 segundos, se descarta que se pueda migrar de manera eficiente, a la solución desarrollada en el ecosistema Hadoop; uno de los procesos relacionados con el proceso principal, p_carga_tbl_maestro_factores, ya que el tiempo de este, en el sistema original; era menor a un segundo.

9.2 Tiempo del proceso migrado

Primero se obtuvo el tiempo que tardo el proceso principal en el sistema original en ejecutarse.

	Fecha	Hora
Hora inicio proceso original	05/01/2015	6:49:15
Hora fin proceso original	05/01/2015	6:53:15
Tiempo proceso original	0	0:04:00
Tiempo proceso original segundos	**0**	**240**

Tabla 16: Tiempo proceso original

Como se puede observar, el proceso tarda 4 minutos exactos, o lo que es lo mismo; 240 segundos en ejecutarse.

En la siguiente tabla podemos ver las características de la máquina virtual que se utilizó para la ejecución del proceso migrado.

Características máquina virtual	
Memoria	4.9 GB
Procesadores	2
Disco Duro (SCSI)	64 GB
Software Dentro Máquina Virtual	CHD4
Software	VMWare 11

Tabla 17: Características máquina virtual

En la siguiente tabla podemos observar una serie de columnas con: el tiempo de ejecución en segundos, el número de iteración, identificador, el tiempo de inicio, el tiempo de fin, y el número de filas de la tabla con los datos finales.

En el caso que el identificador sea igual que 0, únicamente se lanzaba el proceso migrado; en el caso que fuese igual a 1, se realizaba una limpieza de las tablas del sistema, se realizaba todos los pasos precedentes del proceso y finalmente se lanzaba el proceso migrado para evaluarlo. La asignación del identificador es completamente aleatoria para cada iteración. Tal y como se puede observar no existe una dependencia entre el tiempo de ejecución y el valor del identificador.

segundos	nºiteración	identificador	tiempo-inicio	tiempo-fin	nºfilas
187	7	0	1429037947	1429038134	2664200
185	10	0	1429040485	1429040670	2664200
187	14	0	1429044429	1429044616	2664200
182	16	0	1429045849	1429046031	2664200
184	17	0	1429046514	1429046698	2664200
191	19	0	1429047993	1429048184	2664200
178	22	0	1429050410	1429050588	2664200
182	23	0	1429051073	1429051255	2664200
184	24	0	1429051799	1429051983	2664200
179	30	0	1429057788	1429057967	2664200
195	1	1	1429031108	1429031303	2664200
191	2	1	1429032490	1429032681	2664200
189	3	1	1429033681	1429033870	2664200
187	4	1	1429035117	1429035304	2664200
183	5	1	1429036353	1429036536	2664200
182	6	1	1429037162	1429037344	2664200
180	8	1	1429039191	1429039371	2664200
182	9	1	1429039879	1429040061	2664200
180	11	1	1429041243	1429041423	2664200
182	12	1	1429042711	1429042893	2664200
180	13	1	1429043886	1429044066	2664200
186	15	1	1429045120	1429045306	2664200
183	18	1	1429047686	1429047869	2664200
182	20	1	1429049047	1429049229	2664200
183	21	1	1429050043	1429050226	2664200
188	25	1	1429052860	1429053048	2664200
176	26	1	1429053862	1429054038	2664200
181	27	1	1429054967	1429055148	2664200
184	28	1	1429056015	1429056199	2664200
185	29	1	1429057299	1429057484	2664200

Máximo	Promedio	Mínimo
195	183,9	176

Tabla 18: Datos proceso migrado

Suma total	5518
Nº valores total	30
Valor Máximo	195
Valor Mínimo	176
Suma valores aceptados	5147
Nº valores aceptados	28
Tiempo proceso migrado	**183,82**

Tabla 19: Análisis tiempo proceso migrado

En la tabla anterior podemos observar ciertos detalles importantes sobre los datos obtenidos. Primeramente tenemos la suma total del tiempo de todas las iteraciones seguido por el número total de iteraciones realizadas.

Le siguen el valor máximo y el mínimo de todos los datos, los cuales serán descartados en el cálculo de del tiempo del proceso migrado. Estos junto la suma total, obtenemos la suma de valores aceptados, y el número de valores aceptados. Finalmente extraemos que el tiempo del proceso migrado es de aproximadamente 183,82 segundos.

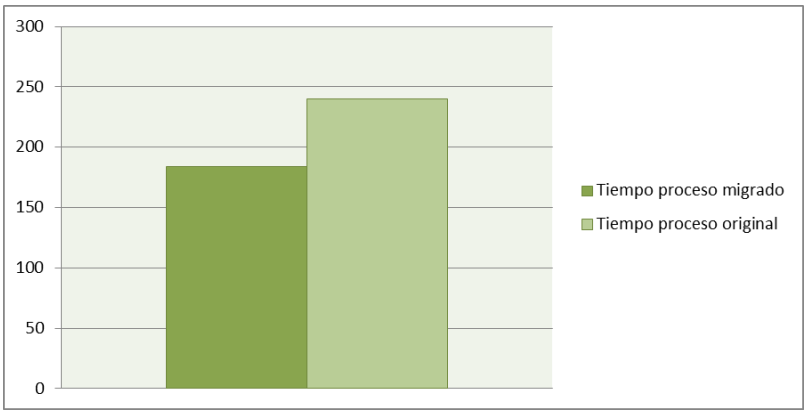


Figura 14: Gráfico de tiempos

Cantidad de tiempo ahorrado (segundos)	56,18
% de tiempo respecto al original	76,59%
% de ahorro respecto al original	23,41%
Speedup (veces más rápido)	1,31

Tabla 20: Análisis relativo

En la figura anterior podemos ver en un gráfico de barras el tiempo de procesado original, y el tiempo de proceso migrado; de este modo podemos observar de manera gráfica el ahorro de tiempo que supone la migración.

En la tabla analizamos la cantidad estimada de tiempo ahorrado, unos 56,18 segundos; junto con el porcentaje de tiempo que tarda respecto al original, aproximadamente un 76,59%; y el porcentaje de tiempo que se ahorra respecto al original, alrededor de un 23,41%. Además calculando el *speedup* como si se tratase de la misma máquina la migración es 1,31 veces más rápida que el proceso original.

Se ha de tener en consideración que existen ciertas limitaciones en el análisis de la solución, y que una de las ventajas del ecosistema Hadoop es que, mediante su sencilla escalabilidad horizontal; permite replicar y distribuir la información entre diferentes máquinas evitando de esta manera un cuello de botella, en comparación con un sistema centralizado de una única máquina. En la solución aquí generada se ha utilizado únicamente un sola máquina, por lo que es de esperar que escalando el sistema de forma horizontal se consiguiera reducir aún más este tiempo.

9.3 Tiempo proceso relacionado

Se estudió si el proceso relacionado restante, p_carga_tbl_maestro_factores_total; era una buena opción para ser migrado utilizando la solución. Para ello se ejecutó ese proceso y se tomó nota de los tiempos de ejecución del mismo.

segundos	nºiteración	tiempo-inicio	tiempo-fin	nºfilas
240	1	1430139964	1430140204	39305
237	2	1430140232	1430140469	39305
242	3	1430140507	1430140749	39305
268	4	1430140773	1430141041	39305
286	5	1430141078	1430141364	39305

Máximo	286
Promedio	254,6
Mínimo	237

Tabla 21: Datos proceso relacionado migrado

Tal y como se puede observar en la tabla, los tiempos no tienen una relación directa con el número de iteración (segunda columna). Además, podemos observar como cada ejecución del proceso genera el mismo número de resultados en la tabla, validando que el resultado es idéntico.

La tabla inferior se ha calculado a partir de los 3 valores centrales el tiempo aproximado que consume la ejecución de este proceso, unos 250 segundos.

Además, se ha obtenido el tiempo que tardaba el proceso original en ejecutarse en la máquina original, unos 120 segundos en total.

Suma total	1273
Nº valores total	5
Valor Máximo	286
Valor Mínimo	237
Suma valores aceptados	750
Nº valores aceptados	3
Tiempo proceso migrado	**250,00**
Tiempo proceso original	**120,00**

Tabla 22: Análisis tiempo proceso relacionado migrado

En la tabla inferior podemos observar cómo no se obtiene una mejora de tiempo, ya que el proceso realizado en la solución del ecosistema Hadoop consume más tiempo, algo más de un 200% respecto al original; y por tanto es más ineficiente.

Mejora en tiempo	**-130,00**
% respecto el original	**208%**

Tabla 23: Análisis relativo proceso relacionado migrado

Se cree que el motivo principal que provoca un tiempo tan alto en comparación respecto al original es que este proceso está compuesto de tres partes claramente diferenciadas, además estas partes están compuestas principalmente de *joins* de diferentes tablas filtrando los resultados finales. Por tanto se cree que el gran número de *Mappers y Reducers* que se debe generar para poder resolver esta *query* es el principal factor para el tiempo obtenido.

9.4 Formato de los datos

El formato en el cual se almacenan los datos dentro de HDFS es el mismo, se realice mediante las funciones de HDFS o se realice mediante la herramienta de Sqoop. Los datos se almacenan dentro de un fichero de texto plano, con un formato muy similar al de los *CSV*, pero no utiliza comas; utiliza un delimitador estándar "^A" o expresado en formato decimal como "\001".

Existe una diferencia en las rutas que se utilizan para almacenar los datos dentro de HDFS, siendo las rutas utilizadas por Sqoop genéricas, almacenando los ficheros dentro de las carpetas de Hive. Por otro lado si se cargan los datos mediante funciones de HDFS, Hive permite acceder a los datos desde la ruta dónde se encuentren siempre que se encuentre su ruta definida en la tabla creada.

9.5 Tiempo carga de datos

Datos			Tables	Scripts					Scripts	Summary	
				Ingestion Extract			Load			Ingestion Extract	Load
SQOOP	it	SQVal		Cargar Datos CSV	Crear Tablas	Cargar Datos Sqoop	Cargar UDFs	Adaptar Datos			
1	17	SQOOP=1		0	25	105	0	73		130	73
1	15	SQOOP=1		0	24	93	0	77		117	77
1	13	SQOOP=1		0	24	101	0	95		125	95
1	11	SQOOP=1		0	27	94	0	74		121	74
1	9	SQOOP=1		0	26	104	0	78		130	78
1	8	SQOOP=1		0	28	92	1	77		120	78
1	7	SQOOP=1		0	23	101	0	78		124	78
1	5	SQOOP=1		0	27	95	0	79		122	79
1	1	SQOOP=1		0	17	119	1	83		136	84
Max	17			0,00	28,00	119,00	1,00	95,00		136,00	95,00
Min	1			0,00	17,00	92,00	0,00	73,00		117,00	73,00
Sum	86			0,00	221,00	904,00	2,00	714,00		1125,00	716,00
Count	9			9,00	9,00	9,00	9,00	9,00		9,00	9,00
Aprox				**0,00**	**25,14**	**99,00**	**0,14**	**78,00**		**124,57**	**78,29**
CSV	it	SQVal		Cargar Datos CSV	Crear Tablas	Cargar Datos Sqoop	Cargar UDFs	Adaptar Datos			
0	18	SQOOP=0		44	41	0	0	76		85	76
0	16	SQOOP=0		46	33	0	0	76		79	76
0	14	SQOOP=0		43	37	0	0	77		80	77
0	12	SQOOP=0		45	34	1	0	82		80	82
0	10	SQOOP=0		43	33	0	0	80		76	80
0	6	SQOOP=0		46	39	0	0	80		85	80
0	4	SQOOP=0		44	35	0	0	80		79	80
0	3	SQOOP=0		44	36	0	0	78		80	78
0	2	SQOOP=0		45	38	0	1	79		83	80
Max	18			46,00	41,00	1,00	1,00	82,00		85,00	82,00
Min	2			43,00	33,00	0,00	0,00	76,00		76,00	76,00
Sum	85			400,00	326,00	1,00	1,00	708,00		727,00	709,00
Count	9			9,00	9,00	9,00	9,00	9,00		9,00	9,00
Aprox				**44,43**	**36,00**	**0,00**	**0,00**	**78,57**		**80,86**	**78,71**
Dif Sqoop vs CSV				✓ 44,4	✓ 10,9	✗ -99,0	✗ -0,1	✓ 0,6		✗ -43,7	✓ 0,4

Tabla 24: Tabla comparativa *scripts*

En la comparativa podemos observar las dos versiones de la carga, una donde se obtienen los ficheros *CSV* de la máquina original, mediante una carpeta compartida en la máquina virtual copiando el fichero desde el *script* de ejecución; otra desde la que se obtienen los datos desde la base de datos relacional. Ambas cargas de datos se realizan compartiendo recursos con la base de datos MySQL, ya que se encuentra en la misma máquina que la máquina virtual.

Los resultados de la tabla demuestran una clara superioridad en tiempo por la carga de datos mediante los volcados, por encima de la carga de datos mediante la herramienta de Sqoop.

Por un lado la diferencia de tiempo relativa a la fase de *Load* de los datos es despreciable, ya que no alcanza el segundo de duración. Por otro lado existe diferencia entre las fases de *Extract*, siendo el tiempo de creación de tablas menor en el caso de Sqoop , debido a crear un número menor de tablas en ese momento; aunque la tarea de cargar los datos mediante Sqoop sea de más del doble que la de los *CSV*.

Se ha de tener en cuenta que la tarea de carga de datos desde lo ficheros *CSV* es completamente dependiente de los datos, si el número de columnas de los datos variara, se debería cambiar el proceso de carga en consecuencia. La solución mediante Sqoop no es dependiente de los datos en ningún sentido, ya que se cargan los datos, y columnas de los mismos; con el formato con el que se encuentren en la base de datos relacional, solamente necesitando acceso a la misma y el nombre de la tabla que se desee cargar. Además Sqoop permite configurar que datos se cargan de las tablas mediante la definición de una *query*, la cual se resolverá en el lado de la base de datos relacional; que determinará los resultados a cargar.

9.6 Complejidad

La solución que se ha desarrollado es algo más compleja que la original. Esta complejidad extra es debida a la propia complejidad generada al haber utilizado herramientas del ecosistema Hadoop. El utilizar herramientas del ecosistema Hadoop fuerza a realizar ciertos aspectos del proceso de manera distinta a como se realizaba originalmente, y al encontrarse las herramientas aún en las primeras fases de su evolución, estas no incluyen tantas opciones como las herramientas originales, las cuáles simplificaban el proceso.

9.7 Flexibilidad

La flexibilidad de un sistema se puede medir en la capacidad que tiene dicho sistema para adaptarse a posibles cambios. Para medir la flexibilidad se han definido una serie de escenarios en los cuales haría falta realizar modificaciones en el sistema. Se analiza la dificultad de realizar los cambios solicitados, y que modificaciones deberían realizarse.

- Modificar los ficheros y la localización de sus carpetas que se utilizan para la carga mediante *CSVs*.
    - Sería necesario modificar el fichero de configuración que inicializa las variables utilizadas en la solución. [Dificultad: Mínima]
- Partiendo de la solución cargando datos desde *CSVs*, pasar a cargar los datos mediante Sqoop.
    - Modificar la variable que determina el origen de los datos, las tablas que se generan y el tratamiento de los datos. [Dificultad: Mínima]
- Evitar la limpieza de los ficheros, y carpetas generados automáticamente por la ejecución del proceso.
    - Modificar la variable que determina la función a ejecutar al finalizar los *scripts*. [Dificultad: Mínima]
- Modificar la dirección IP y el puerto de acceso de la máquina MySQL.
    - Modificar la variable correspondiente. [Dificultad: Mínima]
- Añadir una tabla nueva para ser cargada mediante Sqoop
    - Añadir el nombre de la misma en el *array* de configuración correspondiente. [Dificultad: Pequeña]
- Añadir una nueva tabla para ser cargada mediante funciones HDFS
    - Modificar las variables de la solución con la localización de los datos a cargar.
    - Modificar el *script* encargado de los *CSVs*, añadiendo las llamadas a las dos funciones correspondientes que serían necesarias.
    - Modificar las *querys* de creación de las tablas para permitir acceder a los datos. [Dificultad: Pequeña]

9.8 Transparencia

La solución que se ha generado es transparente al usuario, de manera que no existe ningún motivo por el cual deba conocer el funcionamiento interno de la misma. Al tratarse de un proceso *batch*, esta solución se ejecuta una vez se le ha ordenado que se ejecute, realizando de manera secuencial y automática cada uno de los pasos necesarios para realizar el proceso migrado. Al realizarse de manera automática se asume que no es necesaria ninguna interacción por parte del usuario durante la ejecución de la misma.

Debido a las características mencionadas se podría utilizar la solución presentada como un módulo el cuál sería responsable de realizar el proceso que hemos migrado. Por tanto se podría utilizar junto con otras soluciones.

10 Limitaciones

10.1.1 Herramientas

Utilizar *Strings*, y no *Varchar*

Una de las versiones de Hive que se utilizaron durante la realización del proyecto no aceptaba la utilización de columnas con tipo *Varchar*. De modo que se utilizó el tipo con mayor semejanza posible, siendo este el tipo *String*. Esto tuvo un impacto despreciable en el tiempo del proyecto, por tanto también en su coste. La utilización de *Strings* en el lugar de los *Varchar* condicionó la solución implementada.

No *subquerys* dentro de cláusulas *where*

La versión de Hive 0.13 utilizada para el desarrollo del procesado de datos no incluye la posibilidad de realizar una *subquery* dentro de una cláusula *where*. Debido a esta carencia hubo que realizar una de las partes de los procesos relacionados con el proceso mediante una solución alternativa al procesado original. Partiendo de un *cross join*, junto con un not in dentro de una cláusula *where*, se acabó realizando un *cross join* junto con un *left outer join*. Esto tuvo cierto impacto en el procesado de los datos a nivel de tiempo. Cabe destacar la complejidad de esta solución frente a la original, y por tanto un incremento de dificultad a la hora de asegurar el rigor de los datos.

Función Date_Sub de Apache Hive

La función del sistema Hive *date_sub*, es una función que recibe una fecha y un entero, realiza una resta de tantos días como defina el entero de la fecha que se le pasa por parámetro.

El obstáculo surge cuando el valor de retorno de esta función no tiene un valor válido para ser convertido a *timestamp* de forma automática. Este obstáculo tuvo un gran consumo de tiempo, debido a que el error no era de fácil localización, ya que si se volcaban los datos que se insertaban en la tabla, antes de ser añadido; estos se mostraban por defecto como *Strings*, teniendo un formato correcto. En cambio al insertarse en la tabla correspondiente se realiza una conversión al tipo de la columna de la tabla. Convirtiéndose en un *timestamp* erróneo, con el valor por defecto de la herramienta.

Una vez encontrado el error fue solventado con cierta facilidad, concatenando un *String* estático al final de lo retornado por *date_sub*, pasando a tener un formato adecuado para ser convertido a *timestamp*.

Hardware limitado

La realización del proyecto incluía el acceso por parte del estudiante a un servidor para la realización de las tareas de evaluación. De este servidor no se tuvieron más noticias, ya que el proyecto que teóricamente debía proporcionarlo no estaba avanzando como se esperaba. Además se solicitó un cambio de máquina de trabajo para disponer de un mayor número de recursos, ya que los recursos de la máquina actual eran bastante limitados llevando la máquina a bloqueos y reinicios algunas veces a la semana. La ampliación de recursos se aceptó, aunque aún no se realizó la correspondiente substitución. Dada la información anterior se ha seguido trabajando con la máquina virtual de Cloudera configurada en un único nodo siendo los recursos de este sistema limitados.

Estos recursos limitados tienen un gran impacto a la hora de utilizar Sqoop, debido a que coexistiendo en la misma máquina se encuentra el Servidor MySQL que contiene los datos a acceder. Esto por un lado aumenta la velocidad de acceso, pero reduce la cantidad de memoria disponible para la máquina virtual, aumentando los tiempos obtenidos al evaluar ambas cargas de datos mientras MySQL se encuentra encendido.

10.1.2 Datos

Primera línea con caracteres no visibles

Algunas de las fuentes de datos disponen en la primera línea de una serie de caracteres no visibles. Estos caracteres no visibles provocan, al realizarse la lectura de los datos; un error al convertir estos datos en el formato adecuado de la tabla donde deben insertarse. Para ser capaces de evitar este error, se ha debido comprobar que si se realiza una lectura de un dato, y este se encuentra con un valor el cual no puede convertir en el formato adecuado, se le coloque el valor por defecto, evitando así la creación de un error.

Delimitadores de tres caracteres

El sistema de carga de datos mediante la utilización de Hive permite la utilización de un carácter, o en su defecto el valor decimal de un carácter de la tabla *ASCII*; como delimitador para los ficheros que se cargan. Uno de los orígenes de datos contenía un delimitador formado mediante tres caracteres *ASCII* unidos. Por ello hubo que analizar a fondo las diferentes opciones, finalmente optando por cargar los datos mediante un único *String*, y realizando de manera posterior un *Split* de este para obtener las diferentes columnas de la tabla.

Formato de las fechas

El formato de las fechas no era compatible con el formato de las fechas del sistema. Se estuvieron observando las diferentes opciones que se podían realizar, optando finalmente por crear una nueva función para el sistema que se encargase de la transformación entre ambos formatos. Para ello se creó una *User-Defined Function (UDF).*

Identificador fecha

Para la realización de algunos de los procesos se utiliza un identificador para las fechas formado mediante la unión del año y el mes. Para la realización de este identificador se utilizó una *User-Defined Function (UDF)*, ya que la realización de este identificador mediante la extracción del año y el mes utilizando las funciones del sistema dificultaba la lectura de las *querys*, ya que forzaba realizar un *CASE*, y unir un “0” en ciertas ocasiones.

11 Competencias técnicas

Siguiendo la normativa referente respecto a los trabajos de fin de grado, la memoria debe contener una justificación de cómo se han desarrollado las competencias técnicas asociadas. Por ello se va a proceder a justificar de forma detallada como se ha trabajado cada competencia técnica en este proyecto.

CES1.1: Desarrollar, mantener y evaluar sistemas y servicios de software complejos y/o críticos. Nivel de cumplimiento: Bastante

El proceso que se ha migrado al ecosistema de Hadoop forma parte de uno de los procesos *ETL* que se realizan diariamente, dónde el tiempo de realización del *ETL* es crítico. Dadas las características del ecosistema la implementación del proceso se vuelve compleja.

CES1.2: Dar solución a problemas de integración en función de las estrategias, los estándares y las tecnologías disponibles. Nivel de cumplimiento: En profundidad

Ya que se partía de unos datos de origen con diferentes formatos, ha sido necesario realizar una integración de los mismos adaptándolos a un formato común, necesario para realizar el proceso en la solución creada.

CES1.3: Identificar, evaluar y gestionar los riesgos potenciales asociados a la construcción de software que se pudieran presentar. Nivel de cumplimiento: Bastante

Debido a las características de las herramientas utilizadas para llevar a cabo este proyecto, ha sido necesario tomar conciencia de la capacidad de las mismas, y gestionar la construcción de la solución de una forma que se dispusiera del tiempo necesario para solventar las diferentes dificultades que surgirían.

CES1.4: Desarrollar, mantener y evaluar servicios y aplicaciones distribuidas con soporte de red. Nivel de cumplimiento: En profundidad

El ecosistema Hadoop y sus herramientas tienen, de manera general; su mayor beneficio cuando trabajan de manera distribuida para solucionar un problema. Inicialmente se pensaba desarrollar una solución y ejecutarla en un conjunto de nodos. Aunque esto finalmente no fuera posible, las herramientas tienen un enfoque para poder realizarse de manera distribuida, por ello la solución mediante unas pequeñas modificaciones sería capaz de ejecutarse de manera distribuida.

CES1.5: Especificar, diseñar, implementar y evaluar bases de datos. Nivel de cumplimiento: En profundidad

La herramienta de Hive, almacena la información de manera que desde el punto de vista externo se trata de tablas relacionales. Estas tablas han sido consultadas y algunas debidamente calculadas, como si de una base de datos se tratasen. Además la solución al tratarse de un proceso *ETL*, y al haberse evaluado este, de manera implícita se ha evaluado el sistema de datos existente que lo sustenta.

CES2.1: Definir y gestionar los requisitos de un sistema software. Nivel de cumplimiento: Un poco

Se realizó el proyecto cumpliendo la serie de objetivos definidos inicialmente, junto con los requisitos que se solicitaron para la solución.

CES3.2: Diseñar y gestionar un almacén de datos (data warehouse). Nivel de cumplimiento: Un poco

Ya que se ha migrado un subconjunto de los procesos *ETL* de la compañía, se obtiene un subconjunto de los datos de su D*ata warehouse*, el cual contiene todos los datos de los procesos *ETL*. Este *Data warehouse* migrado, principalmente se compone de HDFS, para el almacenamiento de datos; y de Hive, para traducir las consultas a *Mappers y Reducers* . Por tanto ha debido de ser diseñado utilizando las herramientas disponibles para este proyecto.

12 Sostenibilidad y compromiso social

12.1 Aspectos económicos

Planificación

Este proyecto consta de una fase de evaluación de los costes de la realización del proyecto, tanto de recursos materiales como humanos. Durante la realización del proyecto, se ha tenido en cuenta la necesidad de poder ajustar ciertos aspectos de la solución que se estaba desarrollando. Este proyecto podría ser viable de manera competitiva, ya que es similar a otros proyectos de *ETL*. Gracias a los resultados obtenidos en este proyecto en el futuro se podría desarrollar proyectos similares a este con un coste menor gracias a los conocimientos aprendidos y documentados en este. Se ha dedicado el tiempo necesario a cada tarea según su importancia respecto al TFG y según su dificultad técnica a la hora de llevarse a cabo, se ha de considerar también que en su totalidad ha sido realizado por un estudiante. Este proyecto y tal como se menciona en páginas anteriores es fruto de una colaboración entre everis y la FIB (Facultad de Informática de Barcelona).

Resultados

Gracias a los resultados del proyecto existe la posibilidad de en futuros proyectos, gracias al conocimiento aprendido; reducir el tiempo necesario para implementar una solución. Todo ello junto con el conocimiento sobre la madurez de las herramientas utilizadas permitiría tomar una decisión más acertada a la hora de decidir las herramientas que se utilizarían en un posible proyecto. Se ha conseguido llevar a cabo el proyecto satisfactoriamente utilizando un ordenador de capacidades limitadas siendo esto uno de los factores económicos importantes en los resultados.

Riesgos

Se ha dedicado una cantidad considerable de tiempo a la carga de datos del proyecto desde ficheros *CSV*, esto es debido a las limitaciones técnicas; se trata de una parte posiblemente reutilizable solo parcialmente debido a la estrecha relación de lo realizado con las características de los datos utilizados. Sí se tratase de datos similares a los utilizados una adaptación de los *scripts* sí sería factible.

12.2 Aspectos sociales

Planificación

Este proyecto tiene el objetivo de reducir los tiempos de duración de los procesos *ETL*, siendo estos en muchos proyectos de *BI (Business Intelligence)* de prácticamente del 80% del tiempo total del proyecto. Ya que en la actualidad ciertas organizaciones en su operativa diaria tienen un gran volumen de datos a analizar cada día, es objetivo de este proyecto el reducir este tiempo, permitiendo a estas organizaciones estar disponibles más tiempo o cargar más datos si es que disponían de ellos. De este modo, las mejoras obtenidas en este proyecto pueden aplicarse de manera transversal a diferentes proyectos, mejorando sus servicios y repercutiendo en última instancia a las personas que dependan de estos servicios.

Resultados

Los resultados del proyecto demuestran que dada una máquina a punto de ser dejada de usar, se le puede dar un uso, por tanto esto crea la posibilidad de que cualquier persona con unos recursos similares pueda realizar experimentos sobre *Big Data* para crear conocimiento y divulgarlo. Las conclusiones de este proyecto son claras, una reducción del tiempo de procesado podría aumentar la disponibilidad de ciertos servicios, en muchos casos públicos; a la población evitando que estos servicios estuviesen limitados temporalmente por tanto mejorando su accesibilidad.

Riesgos

No existen riesgos para la sociedad asociados a este proyecto, ya que si el proyecto hubiese sido un fracaso, esto no tendría una repercusión negativa para la sociedad, ya que se trata de un proyecto interno. Aún si se hubiesen obtenido unas migraciones no satisfactorias del proceso, todo el conocimiento adquirido habría sido útil a la hora de tomar decisiones futuras respecto a las herramientas empleadas.

12.3 Aspectos ambientales

Planificación

Los recursos de los cuales hará falta disponer para la realización del proyecto son aquellos mencionados en el apartado de recursos de este mismo. El ahorro de energía no sería un factor diferencial, ya que la solución que ofrece mi proyecto será utilizada con tanta asiduidad como se requiera es por ello que dependerá de en qué clase de proceso se aplique. Se podría reutilizar un servidor que se utilizó anteriormente para la realización de un TFG sobre *Big Data*, aunque en momentos anteriores estaba disponible, actualmente se encuentra caído y sin noticias de futura actividad. Una parte del proyecto podría reutilizarse para otros proyectos, utilizando como base el material creado durante la realización del TFG.

Resultados

Dados los resultados del proyecto podemos determinar que el proyecto resultante es valioso y además es muy viable, ya que durante la realización del mismo se ha comprobado que las capacidades de cálculo obtenidas en la solución son suficientes como para utilizando una máquina a punto de ser cambiada por obsoleta, realizar y reducir el tiempo de cálculo del proceso *ETL* siendo esta máquina seguro de menores características.

Riesgos

Existe una enorme posibilidad de si se decide migrar la totalidad de procesos *ETL* a un ecosistema Hadoop, sea necesario disponer de mayores recursos a la hora de realizarse, para así ser capaces de reducir de manera significativa el tiempo total de ejecución del conjunto de procesos *ETL* que se realizan. Por tanto sería necesario un mayor número de máquinas, las cuales generarían residuos al finalizar su uso.

12.4 Matriz de sostenibilidad

Dada la información y puntuaciones explicada anteriormente, se ha procedido a rellenar la siguiente matriz de sostenibilidad.

¿ Sostenible ?	Económica	Social	Ambiental
Planificación	Viabilidad Económica	Mejora en calidad de vida	Análisis de recursos
valoración	9	7	3
Resultados	Coste final versus previsión	Impacto en entorno social	Consumo de recursos
valoración	7	6	7
Riesgos	Adaptación a cambios de escenario	Daños sociales	Daños ambientales
valoración	-2	0	-5
Suma Columna	**14**	**13**	**5**
valoración total	32		

Tabla 25: Matriz de sostenibilidad

El resultado de la matriz de sostenibilidad es de 32, siendo este número la suma total de las puntuaciones otorgadas a las diferentes celdas, aportando una idea del nivel de sostenibilidad global del proyecto.

13 Conclusiones del proyecto

13.1 Conclusiones

Grandes volúmenes de datos

Dada la información anterior, junto con lo visto en este proyecto; se extrae claramente la conclusión de la gran utilidad existente de este tipo de sistema a la hora de realizar cálculos con grandes volúmenes de datos, siendo capaz de reducir su tiempo de ejecución respecto al tiempo original. Con cierta cantidad de tiempo inicial, aunque siendo capaz de compensarlo durante la realización del proceso.

13.2 Trabajo futuro

Particionado, distribución de carga

Dado que actualmente no existe disponibilidad para realizar la migración del proceso desde un servidor, por ello se ha de realizar en un único nodo. Al realizarse la migración en un único nodo no existe la posibilidad de realizar técnicas de particionado de los datos, ni de distribución de carga entre los diferentes nodos para intentar reducir el tiempo que tarda en realizarse el proceso migrado.

Carga de los datos en un sistema relacional

Un paso interesante que se podría analizar a continuación sería la posibilidad de cargar los datos mediante Sqoop en una base de datos relacional, de manera que se pudiera sustituir el proceso original en la máquina original, por el realizado mediante las herramientas de Hadoop. De manera que el proceso original, y más completo; incluyera el proceso migrado dentro de sus subprocesos.

13.3 Conclusiones personales

La realización de este proyecto dentro de la compañía everis me ha ayudado a tener un punto de vista más realista del sector. Me ha facilitado el tener un contacto directo con personas que se dedican a temas relacionados con lo que me gusta hacer, y eso es una experiencia que de otra forma no habría sido capaz de conseguir.

Este proyecto al tratarse de un TFG, y al haber sido realizado de manera individual; me ha enseñado lo duro que puede ser estar trabajando constantemente en un problema muy concreto, y al no ser capaz de solucionarlo tener que ser capaz de observar el problema con perspectiva y desde diferentes ángulos, tomarlo con calma y ver todas las diferentes posibilidades que podían ser el origen. La falta de información sobre las herramientas aquí utilizadas, me ha forzado a navegar por internet buscando ideas para una posible solución una considerable cantidad de tiempo.

En ocasiones no ha habido forma de realizar lo que se quería con las herramientas de las que se disponía, siendo necesario utilizar la creatividad y el ingenio para dar con una solución factible para el problema que se presentaba.

El aprendizaje de las herramientas utilizadas, junto con la libertad de tomar las decisiones del proyecto, me han ayudado a alcanzar un grado de conocimiento superior que me será útil en mi vida laboral futura.

14 Referencias

[1] «Extract, transform and load - Wikipedia, la enciclopedia libre.» [En línea]. Disponible en: http://es.wikipedia.org/wiki/Extract,_transform_and_load. [Accedido: 09-mar-2015].

[2] «What is big data? - O'Reilly Radar.» [En línea]. Disponible en: http://radar.oreilly.com/2012/01/what-is-big-data.html. [Accedido: 09-mar-2015].

[3] «Big Data Analytics - Free Gartner Research.» [En línea]. Disponible en: http://www.gartner.com/it-glossary/big-data/. [Accedido: 09-mar-2015].

[4] «Welcome to Apache™ Hadoop®!» [En línea]. Disponible en: http://hadoop.apache.org/. [Accedido: 09-mar-2015].

[5] «Apache Hadoop YARN – Concepts and Applications - Hortonworks.» [En línea]. Disponible en: http://hortonworks.com/blog/apache-hadoop-yarn-concepts-and-applications/. [Accedido: 10-mar-2015].

[6] «Apache Hadoop - Wikipedia.» [En línea]. Disponible en: http://en.wikipedia.org/wiki/Apache_Hadoop. [Accedido: 10-mar-2015].

[7] Intel, «Extract, Transform, and Load Big Data with Apache Hadoop*», 2013.

[8] X. Liu, C. Thomsen, y T. B. Pedersen, «ETLMR: A Highly Scalable Dimensional ETL Framework Based on MapReduce», en *Data Warehousing and Knowledge Discovery*, 2011, pp. 96-111.

[9] X. Liu, C. Thomsen, y T. B. Pedersen, «CloudETL: Scalable Dimensional ETL for Hadoop and Hive», 2012.

[10] S. Misra, S. K. Saha, y C. Mazumdar, «Performance Comparison of Hadoop Based Tools with Commercial ETL Tools – A Case Study», en *Big Data Anlaytics*, 2013, pp. 176-184.

[11] «Syncsort - Big Data Solutions for Hadoop, Linux, Unix, Windows, and Mainframes.» [En línea]. Disponible en: http://www.syncsort.com/en/Home#. [Accedido: 18-mar-2015].

[12] «Hadoop Tutorial: Deploying Hadoop ETL.» [En línea]. Disponible en: http://hortonworks.com/hadoop-tutorial/deploying-hadoop-etl-in-the-hortonworks-sandbox/. [Accedido: 18-mar-2015].

[13] «Estudios de remuneración 2015 TECNOLOGÍA», p. 16, 2015.

[14] «Procesamiento por lotes - Wikipedia, la enciclopedia libre.» [En línea]. Disponible en: http://es.wikipedia.org/wiki/Procesamiento_por_lotes. [Accedido: 19-may-2015].

15 Ilustraciones

15.1 Figuras

15.2 Tablas