# Implementación de un Ambiente Big Data para el Procesamiento de Datos Masivos

Julia Patricia Sánchez Solís[1], Vicente García Jiménez[2],
Carlos Daniel Luna Garza[3], Israel Medina Gómez[4] y Jonathan Adrián Herrera Castro[5]

***Resumen*—El volumen de datos digitales en el mundo crece de manera exponencial. Procesar esta cantidad de datos se ha convertido en un problema que debe ser abordado mediante herramientas y tecnologías nuevas. En el presente trabajo se describe la Implementación de un Ambiente Big Data para el Procesamiento de Datos Masivos. El ambiente está basado en una arquitectura maestro-esclavo, se evaluó su desempeño en base al tiempo de respuesta al procesar conjuntos de archivos de texto de diferente tamaño del dataset Gutenberg. Actualmente, no existe un estándar definido para implementar un ambiente Big Data, por lo que esta investigación contribuye a la comunidad ofreciendo una guía que describe su desarrollo. El procesamiento distribuido de datos reduce el tiempo de cómputo, efectuando de manera más eficiente las tareas a realizar.**

***Palabras clave*— clúster, Big Data, Hadoop, YARN, cómputo distribuido.**

## Introducción

En la actualidad se vive en una generación donde el manejo de datos es muy importante. Los servicios ofrecidos tanto en Internet como en aplicaciones ya sean de aspecto social, comercial o científicas, han dado lugar a una problemática de crecimiento exponencial de datos. Como menciona Joyanes (Joyanes Aguilar, 2013), estos datos pueden originarse a partir de cinco diferentes fuentes: 1) Web y medios sociales, 2) Máquina a máquina (Internet de las cosas), 3) Datos de grandes transacciones, 4) Biometría y 5) Generados por los humanos.

En (Cuevas, 2013), IDC (International Data Corporation) proyecta que, hacía el año 2020, el universo digital alcanzará 40 ZettaBytes (ZB), lo que da como resultado un crecimiento 50 veces mayor a partir del inicio del 2010. Según el estudio, 2.8 ZB de datos se generaron y replicaron en 2012. Este crecimiento ha desencadenado el desarrollo de nuevas tecnologías para el procesamiento de información que puedan ayudar a que los tiempos de respuesta sean menores, esto en cuanto al manejo de datos masivos, ya que con las tecnologías tradicionales sería muy costoso o limitado realizar el procesamiento de datos. Teniendo en cuenta lo anterior y la problemática que esto ha causado, en la literatura se ha considerado que las bases de datos tradicionales han quedado limitadas o insuficientes para el almacenamiento de grandes cantidades de datos (Pérez Marqués, 2015). A raíz de esto, las herramientas para la gestión de datos de gran magnitud (Big Data), permiten tener una mejora en cuanto a la capacidad de procesar y almacenar este tipo de información de manera distribuida. McKinsey Global Institute (Bobbs, et al. Mayo 2011) define Big Data como "Conjunto de datos cuyo tamaño está más allá de la capacidad de capturar, almacenar, administrar y analizar, con las herramientas típicas de software de base de datos".

Dentro de las herramientas que brindan ayuda para la implementación de procesos en centros de datos distribuidos existe una gran variedad de aplicaciones y algoritmos que generan beneficios de manera distinta, muchos de ellos basados en las arquitecturas distribuidas en ambientes Hadoop, el cual es un framework que ayuda a procesar grandes cantidades de datos en clústeres de computadoras (Joyanes Aguilar, 2013).

La necesidad de crear arquitecturas distribuidas cada día está más presente. Las empresas se preocupan por tener acceso a toda la información que ellos consideran relevante y por ende el poder procesarla es indispensable para la toma de decisiones. En la actualidad no existe un estándar definido que sea útil para, desarrollar una arquitectura de clúster Big Data, saber qué herramientas utilizar, determinar cuál tipo de hardware sería el más adecuado para cada herramienta, entre otros aspectos. Como se menciona en (Mondaca et al. 2017), los beneficios que aportan las aplicaciones de cómputo distribuido para datos masivos no solo impactan de manera positiva a las empresas privadas, sino que también llegan a ser utilizadas por las instituciones públicas. Esto lleva a que surja una

[1] Julia Patricia Sánchez Solís es Profesora Investigadora del Departamento de Ingeniería Eléctrica y Computación de la Universidad Autónoma de Ciudad Juárez, Chihuahua, México julia.sanchez@uacj.mx (**autor corresponsal**)
[2] Vicente García Jiménez es Profesor Investigador del Departamento de Ingeniería Eléctrica y Computación de la Universidad Autónoma de Ciudad Juárez, Chihuahua, México vicente.jimenez@uacj.mx
[3] Carlos Daniel Luna Garza es egresado de la carrera de Ingeniería en Sistemas Computacionales de la Universidad Autónoma de Ciudad Juárez al122377@alumnos.uacj.mx
[4] Israel Medina Gómez es egresado de la carrera de Ingeniería en Sistemas Computacionales de la Universidad Autónoma de Ciudad Juárez al122485@alumnos.uacj.mx
[5] Jonathan Adrián Herrera Castro es estudiante de la carrera de Ingeniería en Sistemas Computacionales de la Universidad Autónoma de Ciudad Juárez al150187@alumnos.uacj.mx

gran demanda sobre expertos en la materia para poder trabajar con grandes volúmenes de datos (Fuentes Pino, 2018), menciona que el motivo por el cual no hay suficientes profesionistas se encuentra en un modelo educativo poco específico, muy teórico y nada alineado a las tendencias del sector tecnológico. Sin mencionar el precio que conlleva especializarse en esta área y a la carencia educativa.

Considerando lo antes expuesto, en este trabajo se presenta un prototipo de clúster que permite realizar procesamiento de datos en un ambiente Big Data. Se ofrece una guía sobre cómo configurar el software para crear el ambiente Big Data y cómo administrar el procesamiento distribuido en el clúster. El principal motivo para documentar esta guía es apoyar a la toma de decisiones al momento de crear un centro de datos de procesamiento distribuido. Así como también, impulsar la investigación de nuevas herramientas para el procesamiento y análisis distribuido de datos.

## Descripción del método

El desarrollo de este proyecto se basó en la Metodología en Cascada. En este modelo las fases se procesan y se completan una a la vez, no se superponen una sobre otra. La metodología se divide en 5 fases, las cuales son: Requisitos, Diseño, Implementación, Verificación y Mantenimiento. A continuación, se describen las actividades realizadas en cada fase de la metodología.

*Requisitos*

Se estableció como producto final un prototipo de clúster para la implementación de un ambiente *Big Data.* A su vez, este se dividió en productos parciales los cuales consistieron en la configuración de un nodo maestro y dos nodos esclavo. En base a un análisis previo, se realizó la selección de las herramientas adecuadas para la implementación del clúster, las cuales se mencionan a continuación:

Para el sistema operativo se utilizó una distribución de Linux Ubuntu 18.04.1 LTS. Se eligió este sistema operativo debido a su amplia flexibilidad para el desarrollo de sistemas de cómputo distribuido.

Por otro lado, para la implementación del clúster, se seleccionó como framework principal, Hadoop en su versión 2.7.6, el cual permite realizar cómputo distribuido al ejecutar tareas en múltiples nodos simultáneamente y que, a su vez, cuenta con un sistema propio de archivos distribuidos (HDFS, Hadoop Distributed File System).

Como herramienta principal para la verificación del adecuado funcionamiento del clúster se seleccionó el programa llamado *WordCount,* el cual por medio de funciones programadas permite realizar la distribución de las tareas por medio de procesos *MapReduce* dentro de Framework *YARN*, el cual es una tecnología que ayuda en la gestión de aplicaciones de procesamiento Big Data.

*Diseño*

Para la infraestructura del ambiente distribuido se utilizó un servidor de la marca Dell modelo PowerEdge T310 que funge como nodo maestro, el cual se encarga de la administración de los datos por medio del HDFS, y lleva a cabo la distribución de las tareas o procesos a los nodos esclavo bajo la arquitectura del framework YARN. Para los nodos esclavo se utilizó un servidor marca Dell modelo T310 y una computadora marca Dell modelo XPS los cuales se encargaron del procesamiento de información en base a las funciones Map y Reduce de la aplicación *WordCount.* El sistema distribuido está interconectado por medio de un Router NEXXT Solutions, utilizado como principal medio de transferencia de datos y comunicación. El router asigna direcciones de red especificas a cada uno de los nodos. En la Figura 1 se muestra la estructura del prototipo para el ambiente *Big Data*.

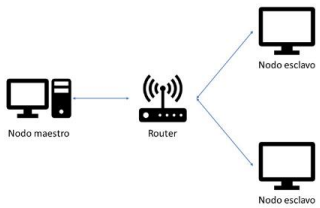


Figura 1. Prototipo de clúster para *Big Data*.

Donde:

Nodo maestro (master): es el encargado de mantener en pie dos servicios principales:

- NameNode: maneja el sistema de archivos distribuidos manteniendo el conocimiento de donde está alojada la información en base a bloques dentro del clúster.
- ResourceManager: administra las tareas bajo YARN y se encarga de programar y ejecutar los procesos en los nodos esclavo.

Nodos Esclavo (Node1, Node2): estos nodos almacenan los datos y proveen el procesamiento de las tareas programadas. También, dan a su vez el recurso para los siguientes servicios:

- DataNode: el cual se encarga del almacenamiento físico de los datos.
- NodeManager: el que da el soporte para la ejecución de las tareas.
- Configuración del archivo hosts.

*Implementación*

En esta fase del proyecto se hicieron las instalaciones y configuraciones de los requisitos ya especificados en la fase anterior. A continuación, se describen cada una de las actividades para la instalación de Java y Haddop.

1. **Instalación del sistema operativo Ubuntu:** en cada nodo del prototipo de clúster, se instaló el sistema operativo Ubuntu Desktop versión 18.04.1 LTS.
2. **Instalación de Java**: antes de instalar Java JDK, se actualizó la lista de paquetes del sistema con el comando `sudo apt-get update`, posteriormente, se instaló OpenJDK utilizando el comando `sudo apt-get install default-jdk`. Por último, se verificó la correcta instalación de OpenJDK usando el comando `$ java -version`.
3. **Configuración de la ruta Java**: se ubicó la ruta de JAVA_HOME utilizando el comando `readlink -f /usr/bin/java|sed "s:bin/java::"`. Posteriormente, se utilizó el comando `sudo nano /usr/local/hadoop/etc/hadoop/hadoop-env.sh`, el cual abrió un archivo sh. En este archivo, se añadieron estas líneas: `#export JAVA_HOME=${JAVA_HOME}` y `#export JAVA_HOME=$(readlink -f /usr/bin/java|sed "s:bin/java::")`
4. **Instalación de Hadoop**: se descargaron dos archivos utilizando `wget http://apache.mirrors.tds.net/hadoop/common/hadoop-2.7.6/hadoop-2.7.6.tar.gz` y `wget https://dist.apache.org/repos/dist/release/hadoop/common/hadoop-2.7.6/hadoop-2.7.6.tar.gz.mds`. Posteriormente, se descomprimió el archivo gz utilizando `tar -xzvf hadoop-2.7.6.tar.gz`. Por último, los archivos extraídos se movieron a la carpeta `/usr/local` con el comando `sudo mv hadoop-2.7.6 /usr/local/hadoop`.
5. **Preparación de Hadoop*:*** para habilitar Hadoop se utilizó el siguiente comando `usr/local/Hadoop/bin/Hadoop`. La salida producida por el comando indicó que Hadoop se habilitó correctamente para ser ejecutado de forma stand-alone.

*Verificación*

Dentro de este apartado se encuentra una introducción hacia la validación de la correcta configuración de los nodos y el sistema distribuido como producto final.

1. **Verificación de la instalación de Hadoop:** para esto, se creó una carpeta llamada *prueba* dentro del directorio home de Hadoop con `mkdir ~/prueba`. Después de haber creado la carpeta, se copiaron dentro de esta, los archivos xml que se encuentran en la carpeta de Hadoop utilizando el comando `cp /usr/local/hadoop/etc/hadoop/*.xml ~/input`. Seguido de esto, se utilizó un programa llamado Grep, el cual cuenta las veces que se repite una palabra en un texto. Para realizar la prueba se ejecutó el comando `/usr/local/hadoop/bin/hadoop jar /usr/local/hadoop/share/hadoop/mapreduce/hadoop-mapreduce-examples-2.7.3.jar grep ~/input ~/grep_example 'principal[.]*'`. Para ver el resultado se ejecutó `cat ~/grep_example/*`, mostrando como salida: '6 principal' y '1 principal.', lo que significa que la tarea MapReduce encontró una coincidencia de la palabra *principal* seguida de un punto y seis coincidencias en las que no tenía punto. Con esto se confirmó la correcta instalación de Hadoop.
2. **Creación de un usuario*:*** se creó un usuario llamado hadoop en cada uno de los nodos del clúster (master, node1, node2) mediante el comando `sudo adduser hadoop admin`, después se configuró una contraseña para el mismo usando `sudo addpassword`.
3. **Distribución de claves key-pairs para los usuarios hadoop**: para generar una conexión y comunicación entre los distintos nodos, Hadoop utiliza un protocolo de comunicación SSH el cual debe estar instalado en cada uno de estos. Primero, se inició sesión con el usuario hadoop desde la terminal en el nodo master usando `su hadoop`. Iniciada la sesión, se generó una llave de autenticación con `ssh-keygen -b 4096` y se estableció una contraseña vacía utilizando el comando `hadoop@master (password's):`. La llave se guardó en `HOME/.ssh/id_rsa.pub` y se copió a cada uno de los

nodos utilizando los comandos `ssh-copy-id -i $HOME/ .ssh/id_rsa.pub hadoop@master`, `ssh-copy-id -i $HOME/ .ssh/id_rsa.pub hadoop@node1` y este último también para `node2`. También se copió la llave dentro del mismo nodo master para que se generará una copia de seguridad y por si el mismo equipo se utiliza al mismo tiempo como nodo esclavo. Posteriormente, se verificó la conexión desde el nodo master utilizando los comandos `ssh hadoop@node1` y lo mismo para `node2`.

4. **Descarga y descompresión de los archivos binarios de Hadoop***:* desde el nodo maestro y utilizando el usuario hadoop, se cambió con `cd/` al directorio en donde se descargaron los archivos binarios. Se descargó un archivo comprimido que contenía los archivos binarios de Hadoop utilizando el comando `wget: https://www-eu.apache.org/dist/hadoop/common/hadoop-2.7.6/hadoop-2.7.6.tar.gz`. Una vez descargado el archivo se descomprimió utilizando `tar -xzf hadoop-2.7.6.tar.gz`. Los archivos descomprimidos fueron movidos a la carpeta del usuario hadoop utilizando `mv hadoop-2.7.6 hadoop`.
5. **Configuración de las variables de entorno***:* para utilizar los comandos de Hadoop se configuraron las variables de entorno. Para acceder al archivo de configuración se ejecutó `nano /home/hadoop/.profile`. Se agregó al archivo la siguiente línea `PATH=/home/hadoop/hadoop/bin:/home/hadoop/hadoop/sbin:$PATH`.
6. **Configuración del NameNode:** esta configuración se realizó en el nodo *master* y se replicó en los nodos esclavo. Se accedió al archivo `core-site.xml` utilizando el comando `sudo nano /usr/local/hadoop/etc/hadoop/core-site.xml` y se añadieron a este las siguientes líneas `<name>fs.defautl.name</name>` y `<value>hdfs://master:9000</value>`.
7. **Configuración de la ruta para el HDFS***:* se accedió al archivo `hdfs-site.xml` utilizando el comando `sudo nano /usr/local/hadoop/etc/hadoop/hdfs-site.xml` y se añadieron a este las siguientes líneas `<name>dfs.namenode.name.dir</name>`, `<value>/home/hadoop/data/nameNode</value>`, `<name>dfs.datanode.data.dir</name>`, `<value>/home/hadoop/data/dataNode</value>`, `<name>dfs.replication</name>` y `<value>1</value>`. La propiedad (`dfs.replication`) indica las veces que la información es duplicada dentro del clúster, se recomienda que este valor nunca exceda de la cantidad de nodos que se tienen.
8. **Configuración de YARN como el programador de trabajos:** se utilizó uno de los archivos binarios ingresando a su ubicación con el comando `cd ~/hadoop/etc/hadoop`. Se renombró el siguiente archivo `mapred-site.xml.template` como `mapred-site.xml`, este se accedió con el comando `sudo nano mapred-site.xml` y se añadieron las siguientes líneas `<name>mapreduce.framework.name</name>` y `<value>yarn</value>`.
9. **Configuración de los parámetros de YARN:** se accedió al archivo de YARN utilizando el comando `sudo nano /usr/local/hadoop/etc/hadoop/yarn-site.xml` y añadiendo a este las siguientes líneas `<name>yarn.acl.enable</name>`, `<value>0</value>`, `<name>yarn.resourcemanager.hostname</name>`, `<value>node-master</value>`, `<name>yarn.nodemanager.aux-services</name>` y `<value>mapreduce_shuffle</value>`.
10. **Configuración de los nodos esclavo:** en el nodo master se accedió al archivo llamado `slaves` por medio del siguiente comando `sudo nano /usr/local/hadoop/etc/hadoop/slaves`, y se agregó a este el nombre de los nodos esclavos, en este caso Node1 y Node2. También, se creó en el nodo master una copia comprimida (.gz) de los archivos binarios mediante el comando `tar -xzvf hadoop_config.tar.gz /usr/local/Hadoop/`. Una vez generado el archivo se copió a cada uno de nodos esclavo utilizando el siguiente comando `scp hadoop_config.tar.gz node1:/home/Hadoop`. Luego se ingresó a cada nodo por medio de SSH y se descomprimieron los archivos, esto se realizó usando los comandos `ssh node1`, `tar -xzf hadoop_config.tar.gz`, `mv hadoop_config hadoop` y `exit`.
11. **Formateo del HDFS***:* dentro del nodo maestro y usando el usuario hadoop, el HDFS se formateó con el comando `/usr/local/hadoop/sbin/hdfs namenode -format`, con lo cual Hadoop quedó configurado y listo para utilizarse.

12. **Iniciar y detener el HDFS:** estas acciones se hacen desde el nodo master. Para inicializar el HDFS se ejecutó el siguiente script `./start-dfs.sh`. Esto hizo que se inicializaran los demonios asociados a las tareas del HDFS. Para detener el proceso se ejecutó el siguiente script `./stop-dfs.sh`.
13. **Monitorear el HDFS**: para obtener información sobre el sistema de archivos distribuidos, se utilizaron los siguientes comandos `/usr/local/hadoop/sbin/hdfs dfsadmin –report` y `/usr/local/hadoop/sbin/hdfs dfsadmin –help`. Otra forma en la que se obtuvo información sobre el sistema fue ingresando desde un navegador web a la dirección del nodo maestro así `http://node-master-IP:50070`. Dentro de este sitio web se mostró una interfaz gráfica con la cual se puede interactuar y obtener información sobre el sistema.
14. **Agregar y obtener datos del HDFS: p**ara agregar datos al sistema de archivos lo primero que se realizó fue crear una carpeta dentro del sistema utilizando el siguiente comando `/usr/local/hadoop/sbin/hdfs dfs –mkdir nombre_del_directorio`. Para la carga de archivos del master al HDFS se utilizó el comando `/usr/local/hadoop/sbin/hdfs dfs –put ruta_del_archivo_a_agregar nombre_del_directorio`. Para obtener una lista completa de los comandos que se pueden utilizar dentro del HDFS se empleó el comando `/usr/local/hadoop/sbin/hdfs dfs –help`.
15. **Ejecutar y monitorear YARN**
    Se utilizó el comando `cd /usr/local/hadoop/bin` para ubicarse en el directorio donde se encuentra el script que inicializa los demonios de YARN. Posteriormente con el comando `./start-yarn.sh` se inicializaron los demonios. Para verificar que los demonios de YARN se inicializaron correctamente se utilizó el comando `jps`. Los demonios se detuvieron utilizando el comando `./stop-yarn.sh`. YARN también cuenta con una serie de comandos específicos los cuales ayudan a obtener información sobre nodos y aplicaciones activas. Los comandos son los siguientes `/usr/local/hadoop/sbin/yarn node –list` y `/usr/local/hadoop/sbin/yarn application –list`. YARN también cuenta con una interfaz gráfica la cual puede ser consultada accediendo a la dirección `http://master:8088`.

*Mantenimiento*

Como parte de los objetivos de este proyecto, se realizó una guía detallada para la implementación y configuración de un clúster para procesamiento de datos masivos, la cual incluye:

- Instalación de las paqueterías de JAVA para la interacción de la aplicación *WordCount* con Hadoop.
- Instalación del framework Hadoop.
- Configuración de las conexiones entre los nodos y comprobación que se realizó correctamente.
- Verificación utilizando la interfaz gráfica Hadoop.

Entre algunas otras de las configuraciones necesarias para concluir con un sistema que permita realizar procesamiento de datos de manera distribuida.

## Comentarios finales

*Resultados*

Se realizaron diversas pruebas para comprobar el funcionamiento y verificar la eficiencia del procesamiento distribuido. Estas pruebas consistieron en formar grupos de archivos de texto de distintos tamaños para ser procesados por el clúster y tomar como punto de comparativa el tiempo que le tomaba al sistema realizar la tarea definida. Para tener una mejor referencia, se realizaron las mismas pruebas en dos distintos escenarios, el primero realizado en una configuración de un solo nodo, y el segundo escenario con un par de nodos para realizar los procesos. Los archivos que se utilizaron fueron descargados de un dataset llamado Gutenberg Dataset (Ramiah, 2017) el cual es una colección que consta de 3,036 libros de 142 autores distintos. Todos los archivos descargados para realizar las pruebas fueron archivos de texto plano con extensión txt. En la Tabla 1 se muestran los resultados obtenidos después de realizar las pruebas.

| Prueba | Tamaño (MB) | Tiempo (seg) | | Decremento |
|---|---|---|---|---|
| | | 1 Nodo | 2 Nodos | |

| | | | | |
|---|---|---|---|---|
| PRUEBA 1 | 12.2 | 55 | 56 | **1.82%** |
| PRUEBA 2 | 50.5 | 88 | 62 | -29.55% |
| PRUEBA 3 | 248.2 | 504 | 259 | -48.61% |
| PRUEBA 4 | 1224 | 7571 | 3712 | -50.97% |
| PRUEBA 5 | 4296 | 30467 | 12955 | -57.48% |

Tabla 1. Resultados de las pruebas.

*Discusiones*

Al finalizar las pruebas se pudo observar que, al procesar 50.5 megabytes de datos o menos, el tiempo de cómputo no refleja una gran diferencia al procesar la información en un solo equipo respecto al utilizar dos. El cambio se pudo apreciar cuando se procesaron datos superiores al tamaño antes mencionado, ya que el tiempo del procesamiento utilizando dos nodos se redujo en un 48.61% en la prueba 3 y hasta un 57.48% en la prueba 5. Esto, en comparación con el tiempo que tomó hacerlo con un solo nodo. Las ventajas que tiene un clúster es que su configuración se puede modificar. Por ejemplo, se pueden agregar cuantos nodos esclavos se tengan disponibles, teniendo en cuenta que la escalabilidad del sistema distribuido se puede realizar de manera horizontal.

*Conclusiones*

En este trabajo se implementó un prototipo de clúster para realizar procesamiento de datos en un ambiente *Big Data.* Se evaluó su desempeño en términos de tiempo al procesar archivos de texto de distintos tamaños. Se compararon los tiempos de ejecución de cada proceso utilizando un solo nodo esclavo y dos nodos esclavos. Se creó una guía para la implementación de un prototipo de clúster, que, realizando cómputo distribuido, puede ayudar a procesar información de manera más eficiente resultando en una reducción significativa del tiempo de cómputo. Como conclusión general se puede decir que el cómputo distribuido es una herramienta que puede ser utilizada en áreas donde se tenga la necesidad de procesar grandes cantidades de datos, ya que puede hacer que estos cobren relevancia al poder analizarlos y procesarlos en un tiempo considerable.

**Agradecimientos**

Los autores desean agradecer el apoyo financiero del PRODEP (Proyecto UACJ-PTC-423).

**Referencias**

Bobbs, R., Brown, B., Bughin, J., Chui, M., Hung, B. A., Manyika, J., & Roxburgh, C. (Mayo 2011). *Big Data: The next frontier for innovation, competition, and productivity.* McKinsey & Company.

Cuevas, J. (21 de 01 de 2013). *JC Magazine.* Obtenido de http://www.jcmagazine.com/emc-nuevo-estudio-sobre-el-universo-digital/

Fuentes Pino, Ó. (2018). *computing*. Obtenido de http://www.computing.es/mercado-ti/opinion/1104344046401/muchos-datos-pocos-profesionales.1.html

Joyanes Aguilar, L. (2013). *Big Data: Análisis de grandes volúmenes de datos en organizaciones.* México: Alfaomega.

Mondaca, J., Palomino, N., & Rodríguez, P. (2017). *El uso de datos masivos y sus técnicas analíticas para el diseño e implementación de políticas públicas en Latinoamérica y el Caribe.* Banco Interamericano de desarrollo.

Pérez Marqués, M. (2015). *Big Data: Tecnicas, herramientas y aplicaciones.* Madrid: Alfaomega.

Ramiah, S. (03 de Enero de 2017). *bytequest*. Obtenido de http://bytequest.net/index.php/2017/01/03/freely-available-large-datasets-to-try-out-hadoop/