# Mahaut

Mahout es una biblioteca de aprendizaje automático con herramientas para clústeres, clasificación y sistemas de recomendación, incluidas herramientas diseñadas para calcular elementos más similares o crear recomendaciones de elementos para los usuarios. Mahout emplea el marco de trabajo de Hadoop para distribuir cálculos en un clúster y ahora incluye los métodos de distribución de trabajo adicionales, incluidos Spark.

Mahout se apoya en [Hadoop](http://hadoop.apache.org/), una vez que los datos grandes se almacenan en el Sistema de archivos distribuidos de Hadoop (HDFS), Mahout proporciona las herramientas de ciencia de datos para encontrar automáticamente patrones significativos en esos grandes conjuntos de datos. El proyecto Apache Mahout tiene como objetivo hacer que sea más rápido y cómodo convertir datos grandes en información grande.

La lista de [algoritmos soportados](https://cwiki.apache.org/confluence/display/MAHOUT/Algorithms) por esta librería hasta el momento incluye:

* Clasificación: regresión logística (SGD), bayes, random forest.
* Clustering: k-medias, LDA.
* Búsqueda de conjuntos frecuentes y coocurrencias de términos.
* Factorización de matrices sparse para reducción de dimensionalidad: SVD, SSVD.
* Filtrado colaborativo.
* Algoritmos evolutivos (funcionalidad obsoleta/descontinuada).

## Ventajas e inconvenientes

Uno de los mayores puntos fuertes de Mahout es su uso para sistemas de recomendación mediante técnicas de filtrado colaborativo.

Entre los inconvenientes de esta librería cabe destacar que Mahout no incluye ningún algoritmo de regresión lineal.

Por otro lado, gran parte de los algoritmos de clasificación o clustering que implementa están orientados a análisis textual. Esto es consecuencia del origen de Mahout como subproyecto del motor de búsqueda [Apache Lucene](http://lucene.apache.org/).

Comparada con otras herramientas de software libre con funcionalidades similares, Mahout probablemente es la opción más robusta, orientada a soluciones empresariales. En este sentido, otro de sus principales puntos fuertes es el poder ejecutarse fácilmente sobre plataformas Hadoop en cloud como la que ofrece Amazon EMR. Frente a otras herramientas como **“**[**GraphLab”**](http://graphlab.org/) o **“**[**Vowpal Wabbit**](http://hunch.net/~vw/)**”**, que ofrecen tal vez un enfoque más experimental, centrado en la eficiencia y algoritmia estado-del-arte, a costa de una menor tolerancia a fallos y/o una mayor complejidad en el despliegue para entornos de producción. El mayor nivel de adopción y comunidad de usuarios de Mahout también supone una ventaja práctica frente al resto de alternativas open-source comparables.

# R

R es un software para el análisis estadístico de datos considerado como uno de los más interesantes programas para este propósito. R proporciona una amplia variedad de técnicas estadísticas (modelado lineal y no lineal, pruebas estadísticas clásicas, análisis de series de tiempo, clasificación, agrupamiento) y gráficos, y es altamente extensible. El lenguaje S suele ser el vehículo de elección para la investigación en metodología estadística, y R proporciona una ruta de código abierto para la participación en esa actividad.

Uno de los puntos fuertes de R es la facilidad con la que se pueden producir parcelas de calidad de publicación bien diseñadas, que incluyen símbolos matemáticos y fórmulas cuando es necesario. Se ha tenido mucho cuidado con los valores predeterminados para las opciones menores de diseño en los gráficos, pero el usuario conserva el control total.

El mayor inconveniente que podría presentar es el hecho de funcionar mediante comandos, lo que para algunos usuarios puede resultar complejo. Para solventar esta dificultad existe un paquete llamado R Commander que permite utilizar R sin tener que escribir los comandos, es decir, con la sola utilización del ratón. Una vez realizado el registro para esta actividad, tendrá acceso a materiales que permiten aprender a: descargar e instalar el programa, manipular ficheros de datos, hacer representaciones gráficas y realizar análisis estadísticos de mayor o menor complejidad.

En el siguiente cuadro se muestra las funcionalidades de R en comparación con otros programas de uso estadístico.



Cuadro comparativo de R

# Apache ZooKeeper

ZooKeeper es un proyecto de Apache de código abierto que proporciona infraestructura centralizada y servicios que permiten la sincronización a través de un clúster Apache Hadoop.

ZooKeeper mantiene los objetos comunes necesarios en grandes entornos de clúster. Los ejemplos de estos objetos incluyen información de configuración, espacio de nombres jerárquico, etc.

Las aplicaciones aprovechan estos servicios para coordinar el procesamiento distribuido en grandes clusters.

Entre las aplicaciónes que se puede usar como alternativa a Zookeper estan [Consul](https://www.consul.io/intro/vs/zookeeper.html) y [doozerd](https://github.com/ha/doozerd)

## Funcionamiento

Imagine un clúster Hadoop que abarca 500 o más servidores básicos. Hay una necesidad de gestión centralizada de todo el clúster en términos de nombre. Servicios de grupo y sincronización, gestión de configuración y más. Otros proyectos de código abierto que aprovechan los clústeres Hadoop requieren servicios de clúster cruzado. La incrustación de ZooKeeper alivia la construcción de servicios de sincronización desde cero. La interacción con ZooKeeper ocurre a través del tiempo de la interfaz Java o C.

Para las aplicaciones, ZooKeeper proporciona una infraestructura para la sincronización entre nodos. Hace esto manteniendo la información del tipo de estado en la memoria en los servidores de ZooKeeper.  Dentro de ZooKeeper, una aplicación puede crear lo que se denomina znode (un archivo que persiste en la memoria en los servidores de ZooKeeper). El znode puede ser actualizado por cualquier nodo en el clúster, y cualquier nodo en el clúster puede registrarse para estar informado de los cambios en ese znodo.

# Apache Hive

El software de almacenamiento de datos Apache Hive facilita la lectura, escritura y administración de grandes conjuntos de datos que residen en el almacenamiento distribuido mediante SQL. La estructura se puede proyectar en datos que ya están almacenados. Se proporciona una herramienta de línea de comandos y un controlador JDBC para conectar a los usuarios a Hive. Berkely DB y Level DB pueden ser utilizadas como alternativa para Apache Hive.

## Funcionamiento

Hive utiliza servidores de consultas persistentes con almacenamiento en memoria caché para evitar la latencia Hadoop orientada a lotes y proporcionar tiempos de respuesta rápidos de consulta inferiores a segundos contra volúmenes de datos más pequeños.

Las tablas en Hive son similares a las tablas en una base de datos relacional, y las unidades de datos están organizadas en una taxonomía de unidades más grandes a más granulares. Las bases de datos se componen de tablas, que se componen de particiones.

Dentro de una base de datos particular, los datos en las tablas están serializados y cada tabla tiene un directorio correspondiente del Sistema de Archivos Distribuidos Hadoop (HDFS). Cada tabla se puede subdividir en particiones que determinan cómo se distribuyen los datos dentro de los subdirectorios del directorio de la tabla

El acceso externo a las tablas Hive se admite a través de un componente llamado HiveServer2. Con un controlador ODBC, muchas aplicaciones basadas en SQL pueden interactuar con HiveServer2 y tratar los datos de Hadoop como una base de datos. Las aplicaciones aprovechan esta capacidad para una variedad de casos de uso, que incluyen:

* Minería de datos.
* Análisis Ad-hoc.

# Apache Pig

Apache Pig es una plataforma para analizar grandes conjuntos de datos que consisten en un lenguaje de alto nivel para expresar programas de análisis de datos, junto con infraestructura para evaluar estos programas. La propiedad principal de los programas de Pig es que su estructura es susceptible de una paralelización sustancial, lo que a su vez les permite manejar conjuntos de datos muy grandes.

En la actualidad, la capa de infraestructura de Pig consiste en un compilador que produce secuencias de programas Map-Reduce, para los cuales ya existen implementaciones en paralelo a gran escala (por ejemplo, el subproyecto de Hadoop). La capa de idioma de Pig consiste actualmente en un lenguaje textual llamado Pig Latin, que tiene las siguientes propiedades clave:

* **Facilidad de programación.**Es trivial lograr la ejecución paralela de tareas de análisis de datos simples, "embarazosamente paralelas". Las tareas complejas formadas por múltiples transformaciones de datos interrelacionadas se codifican explícitamente como secuencias de flujo de datos, lo que facilita su escritura, comprensión y mantenimiento.
* **Oportunidades de optimización.**La forma en que se codifican las tareas permite que el sistema optimice su ejecución de forma automática, lo que permite al usuario centrarse en la semántica en lugar de la eficiencia.
* **Extensibilidad.**Los usuarios pueden crear sus propias funciones para hacer un procesamiento especial.

Entre las aplicaciones que se pueden utilizar como alternativa apara Apache Pig tenemos:

* Apache Spark.
* Tableau Desktop.
* Apache Drill.

# Apache Sqoop

Sqoop es una herramienta diseñada para importar fácilmente información desde bases de datos estructuradas (como SQL) y sistemas Hadoop relacionados (como Hive y HBase) en su clúster Hadoop. También puede utilizar Sqoop para extraer datos de Hadoop y exportarlos a bases de datos relacionales y almacenes de datos empresariales.

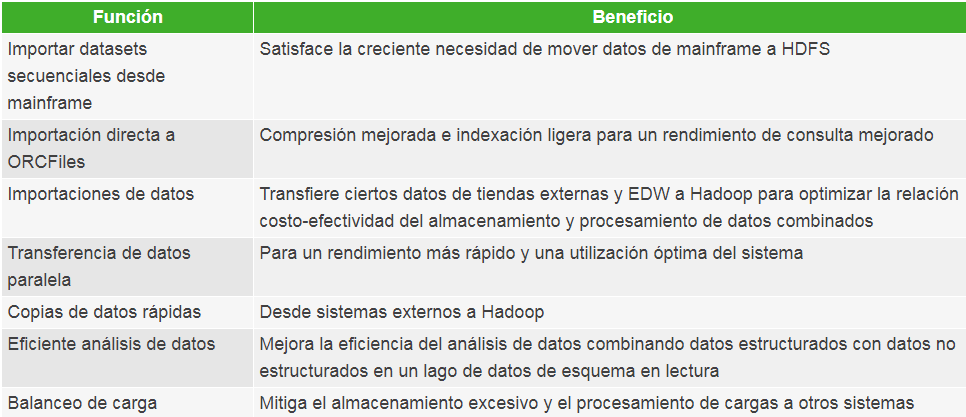
Sqoop ayuda a descargar ciertas tareas (como el procesamiento de ETL) de EDW a Hadoop para una ejecución eficiente a un costo mucho más bajo. Sqoop también se puede usar para extraer datos de Hadoop y exportarlos a almacenes de datos estructurados externos. Sqoop funciona con bases de datos relacionales como Teradata, Netezza, Oracle, MySQL, Postgres y HSQLDB.

Las herramientas que se pueden utilizar como alternativa para Apache Sqoop son:

* Apache Spark
* Apache drill
* Datameer

## QUÉ HACE SQOOP

Apache Sqoop hace lo siguiente para integrar el movimiento masivo de datos entre Hadoop y los almacenes de datos estructurados:



Funciones de Apache Sqoop

# Apache Flume

Flume es un servicio distribuido, confiable y disponible para recopilar, agregar y mover grandes cantidades de datos de eventos, incluidos, entre otros, datos de tráfico de red, datos generados en redes sociales, mensajes de correo electrónico y prácticamente cualquier fuente de datos posible de registro de manera eficiente. Tiene una arquitectura simple y flexible basada en flujos de datos de transmisión. Es robusto y tolerante a fallas con mecanismos de confiabilidad ajustables y muchos mecanismos de conmutación por error y recuperación. Utiliza un modelo de datos extensible simple que permite la aplicación analítica en línea.

Se puede utilizar logstash como alternativa para Apache Flume

Los conceptos que maneja Flume son:

**Evento**: Un payload de bytes con encabezados opcionales que representan la unidad de datos que Flume puede transportar desde su punto de origen hasta su destino final.

**Flujo:** Movimiento de eventos desde el punto de origen hasta su destino final.

**Cliente**: Implementación que opera en el punto de origen de los eventos y los entrega a un agente Flume. Por ejemplo, Log4J appender de Flume es un cliente.

**Agente**: Un proceso independiente que aloja los componentes Fume como las Fuentes (Source), canales y sumideros (Sink), tiene la capacidad de recibir, almacenar y reenviar eventos a su próximo destino.

**Fuente (Source)**: Implementación que puede consumir eventos entregados a él a través de un mecanismo (por ejemplo, una fuente de Avro se puede utilizar para recibir los eventos Avro de clientes u otros agentes en el flujo). Cuando una fuente recibe un evento, se lo entrega a uno o más canales.

**Canal** (Channel): es un almacenamiento temporal para eventos, donde los eventos se entregan al canal a través de fuentes que operan con del agente.

**Sumidero (Sink)**: Implementación que puede eliminar eventos de un canal y transmitirlos al siguiente agente en el flujo, o hasta el destino final del evento.

# MapReduce

MapReduce es un framework que proporciona un sistema de procesamiento de datos paralelo y distribuido. MapReduce está orientado a resolver problemas con conjuntos de datos de gran tamaño, por lo que utiliza el [sistema de archivos distribuido HDFS](http://solidqblogs.azurewebsites.net/big-data/hadoop-hdfs-almacenamiento-y-consulta-parte-2/).

El Framework MapReduce tiene una arquitectura maestro / esclavo. Cuenta con un servidor maestro o JobTracker y varios servidores esclavos o TaskTrackers, uno por cada nodo del clúster.

El JobTracker es el punto de interacción entre los usuarios y el framework MapReduce. Los usuarios envían trabajos MapReduce al JobTracker, que los pone en una cola de trabajos pendientes y los ejecuta en el orden de llegada. El JobTracker gestiona la asignación de tareas y delega las tareas a los TaskTrackers. Los TaskTrackers ejecutan tareas bajo la orden del JobTracker y también manejan el movimiento de datos entre la fase Map y Reduce.

Entre las alternativas para map reduce se tiene: Spark, Pig y Hive.

## Map

La función Map recibe como parámetros un par de (clave, valor) y devuelve una lista de pares. Esta función se encarga del mapeo y se aplica a cada elemento de la entrada de datos, por lo que se obtendrá una lista de pares por cada llamada a la función Map. Después se agrupan todos los pares con la misma clave de todas las listas, creando un grupo por cada una de las diferentes claves generadas. No hay requisito de que el tipo de datos para la entrada coincida con la salida y no es necesario que las claves de salida sean únicas.

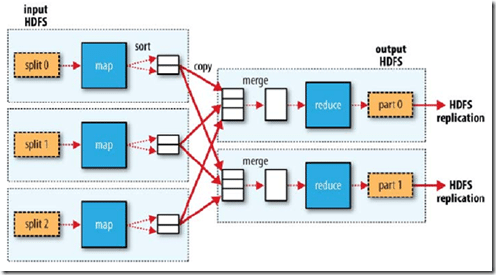
Map (clave1, valor1) –> lista (clave2, valor2)

## Reduce

La función Reduce se aplica en paralelo para cada grupo creado por la función Map().La función Reduce se llama una vez para cada clave única de la salida de la función Map. Junto con esta clave, se pasa una lista de todos los valores asociados con la clave para que pueda realizar alguna fusión para producir un conjunto más pequeño de los valores.

Reduce (clave2, lista (valor2)) –> lista (valor2)

Cuando se inicia la tarea reduce, la entrada se encuentra dispersa en varios archivos a través de los nodos en las tareas de Map. Los datos obtenidos de la fase Map se ordenan para que los pares clave-valor sean contiguos (fase de ordenación, *sort fase*), esto hace que la operación Reduce se simplifique ya que el archivo se lee secuencialmente. Si se ejecuta el modo distribuido estos necesitan ser primero copiados al filesystem local en la fase de copia. Una vez que todos los datos están disponibles a nivel local se adjuntan a una fase de adición, el archivo se fusiona de forma ordenado. Al final, la salida consistirá en un archivo de salida por tarea reduce ejecutada.



Funcionamiento de MapReduce

# Storm

Apache Storm es un sistema de computación en tiempo real distribuido de fuente abierta y gratuita. Storm facilita el procesamiento confiable de flujos ilimitados de datos, haciendo para el procesamiento en tiempo real lo que Hadoop hizo para el procesamiento por lotes. Storm es simple, se puede usar con cualquier lenguaje de programación.

Storm tiene muchos casos de uso: análisis en tiempo real, aprendizaje automático en línea, computación continua, RPC distribuido, ETL y más. Storm es rápido: un punto de referencia registró más de un millón de tuplas procesadas por segundo por nodo. Es escalable, tolerante a fallas, garantiza que sus datos serán procesados, y es fácil de configurar y operar.

## Modos de funcionamiento

Storm puede funcionar en dos modos: local y cluster. El modo local es muy útil para probar el código desarrollado en la topología de Storm ya que corre en una única JVM por lo que podemos hacer pruebas integradas de nuestro sistema, depurar código, etc. y así poder ajustar los parámetros de configuración.

El modo cluster de Storm como su propio nombre indica, ejecuta la topología en cluster, es decir distribuye y ejecuta nuestro código en las distintas máquinas. Es el considerado ‘modo producción’.

## Topología

Una topología en Storm es similar a un grafo. Cada nodo se encarga de procesar una determinada información y le pasa el testigo al siguiente nodo. Esto se configura previamente en la topología. La topología se compone de Spouts y Bolts.

Construimos la topología con la clase TopologyBuilder:

final TopologyBuilder builder = new TopologyBuilder();

## Spout

El componente Spout de Storm es el encargado de la ingesta de los datos en el sistema, por ejemplo si tenemos que leer un fichero de texto y contar las palabras, el componente que recibiría los streams del fichero sería el Spout.

Añadimos el Spout a la topología: builder.setSpout("mySpout", new MySpout());

## Bolt

El Bolt es encargado de consumir las tuplas que emite el Spout, las procesa en función de lo que dicte el algoritmo que programamos sobre los streams de entrada y puede emitirlos a otro Bolt. Es recomendable que cada Bolt realice una única tarea. Si necesitamos realizar varios cálculos o transformaciones sobre los datos que le llegan al Bolt, lo mejor es que se dividan en distintos Bolt para mejorar la eficiencia y la escalabilidad. Tanto el Spout como el Bolt emiten tuplas que serán enviadas a los Spouts que estén suscritos a ese determinado stream configurado en la topología.

Añadimos el Bolt a la topología:

builder.setBolt("myBolt", new MyBolt()).shuffleGrouping("mySpout");

## Stream grouping

Un aspecto importantísimo de Storm es la forma en que se van a compartir los datos entre los distintos componentes por lo que la elección de la topología es crucial, a esto se le denomina stream groupings. Como modelo de datos Storm utiliza tuplas que básicamente son listas de valores con un nombre específico. El valor asociado puede ser un objeto de cualquier tipo, por ello podemos pasar lo que queramos entre nodos implementando un serializador.

* **Shuffe grouping:** Storm decide de forma aleatoria la tarea a la que se va a enviar la tupla de manera que la distribución se realiza equitativamente entre todos los nodos
* **Fields grouping:** Se agrupan los streams por un determinado campo de manera que se distribuyen los valores que cumplen una determinada condición a la misma tarea. Similar a una hash.
* **All grouping**: El stream se pasa a todas las tareas del cluster haciendo multicast.
* **Global grouping**: El stream se envía al bolt con ID más bajo.
* **None grouping**: Bastante similar a shuffle grouping donde el orden no es importante.
* **Direct grouping**: La propia tarea es la encargada de decidir a qué bolt emitir la tupla indicando el ID de ese emisor. Esta forma dota de mayor lógica de distribución en los nodos para que puedan decidir hacia donde redirigir los streams.
* **Local grouping**: Se utiliza el mismo bolt si tiene una o más tareas en el mismo proceso.

# AVRO

Avro es un proyecto de código abierto que proporciona servicios de serialización de datos e intercambio de datos para Apache Hadoop. Estos servicios se pueden usar juntos o de forma independiente. Avro facilita el intercambio de datos grandes entre programas escritos en cualquier idioma. Con el servicio de serialización, los programas pueden serializar eficientemente datos en archivos o en mensajes. El almacenamiento de datos es compacto y eficiente. Avro almacena la definición de datos y los datos en un solo mensaje o archivo.

Avro almacena la definición de datos en formato JSON, lo que facilita su lectura e interpretación, los datos en sí se almacenan en formato binario, por lo que es compacto y eficiente. Los archivos Avro incluyen marcadores que se pueden usar para dividir grandes conjuntos de datos en subconjuntos adecuados para [el](https://www.ibm.com/analytics/hadoop/mapreduce)  procesamiento de [Apache MapReduce ™](https://www.ibm.com/analytics/hadoop/mapreduce) . Algunos servicios de intercambio de datos usan un generador de código para interpretar la definición de datos y producir código para acceder a los datos. Avro no requiere este paso, por lo que es ideal para lenguajes de scripting.

Una característica clave de Avro es el sólido soporte para esquemas de datos que cambian con el tiempo, a menudo llamados evolución de esquema. Avro maneja cambios de esquema como campos faltantes, campos agregados y campos modificados; como resultado, los programas antiguos pueden leer datos nuevos y los nuevos programas pueden leer datos antiguos. Avro incluye API para Java, Python, Ruby, C, C ++ y más. Los datos almacenados con Avro pueden transmitirse desde programas escritos en diferentes idiomas, incluso desde un lenguaje compilado como C a un lenguaje de scripts como Apache Pig.

## Esquemas

Avro se basa en esquema. Cuando se leen datos Avro, el esquema utilizado al escribirlo siempre está presente. Esto permite que cada dato se escriba sin gastos generales por valor, lo que hace que la serialización sea rápida y pequeña. Esto también facilita el uso con lenguajes dinámicos de scripting, ya que los datos, junto con su esquema, son completamente autodescriptivos.

Cuando los datos de Avro se almacenan en un archivo, su esquema se almacena con él, de modo que los archivos pueden ser procesados ​​posteriormente por cualquier programa. Si el programa que lee los datos espera un esquema diferente, esto se puede resolver fácilmente, ya que ambos esquemas están presentes.

Cuando se usa Avro en RPC, el cliente y el servidor intercambian esquemas en el enlace de conexión. (Esto se puede optimizar para que, en la mayoría de las llamadas, no se transmitan realmente esquemas.) Dado que tanto el cliente como el servidor tienen el esquema completo del otro, la correspondencia entre los mismos campos con nombre, campos faltantes, campos adicionales, etc. puede resolverse fácilmente.

Los esquemas Avro se definen con [JSON](http://www.json.org/) . Esto facilita la implementación en idiomas que ya tienen bibliotecas JSON.

## Comparación con otros sistemas

Avro proporciona una funcionalidad similar a sistemas como [Thrift](http://thrift.apache.org/) , [Protocol Buffers](http://code.google.com/p/protobuf/) , etc. Avro difiere de estos sistemas en los siguientes aspectos fundamentales.

* **Tipado dinámico**: Avro no requiere que se genere ese código. Los datos siempre van acompañados de un esquema que permite el procesamiento completo de esos datos sin generación de códigos, tipos de datos estáticos, etc. Esto facilita la construcción de sistemas e idiomas genéricos de procesamiento de datos.
* **Datos no etiquetados**: dado que el esquema está presente cuando se leen los datos, es necesario codificar considerablemente menos información de tipo con los datos, lo que da como resultado un tamaño de serialización más pequeño.
* **No hay ID de campo asignados manualmente**: cuando un esquema cambia, tanto el esquema antiguo como el nuevo siempre están presentes cuando se procesan los datos, por lo que las diferencias se pueden resolver simbólicamente, usando los nombres de los campos.

# Hbase Big Table

Una Bigtable es un hash-map de datos persistente, multidimensional, ordenado, poco denso y distribuido. El hash-map esta indexado por una Have de fila, una Have de columna y un timestamp, cada valor en el hash-map es una matriz de bytes no interpretados.

HBase utiliza un modelo de datos muy similar al de Bigtable. Los usuarios almacenan filas de datos en las tablas de la etiqueta. Una fila de datos tiene una Have ordenada y un número arbitrario de columnas. La densidad de la tabla es baja, por lo que las filas en la misma tabla pueden tener un número muy variable de columnas.

HBase y BigTable se basan en sistemas de archivos distribuidos de manera que el almacenamiento de archivos subyacente puede ser repartido entre un conjunto de máquinas independientes. HBase puede utilizar cualquier sistema de archivos distribuido de Hadoop (HDFS) o Amazon Simple Storage Service (S3), mientras que BigTable hace uso del sistema de archivos de Google (GFS). En el caso de los clusters Hadoop que monta Pragsis, se con!gura HBase para que use HDFS, su almacenamiento más "natural". Los datos se replican a través de una serie de nodos que participan de una manera análoga a cómo los datos están almacenados en los discos en un RAID del sistema. Esto proporciona una capa de protección contra fallos del sistema, como, por ejemplo, un nodo del cluster que deja de funcionar.

# HDFS

Es un sistema de archivos distribuidos diseñado para ejecutarse en hardware básico. Tiene muchas similitudes con los sistemas de archivos distribuidos existentes. Sin embargo, las diferencias con respecto a otros sistemas de archivos distribuidos son significativas. HDFS es altamente tolerante a fallas y está diseñado para implementarse en hardware de bajo costo. HDFS proporciona acceso de alto rendimiento a los datos de la aplicación y es adecuado para aplicaciones que tienen grandes conjuntos de datos. HDFS relaja algunos requisitos POSIX para permitir el acceso continuo a los datos del sistema de archivos.

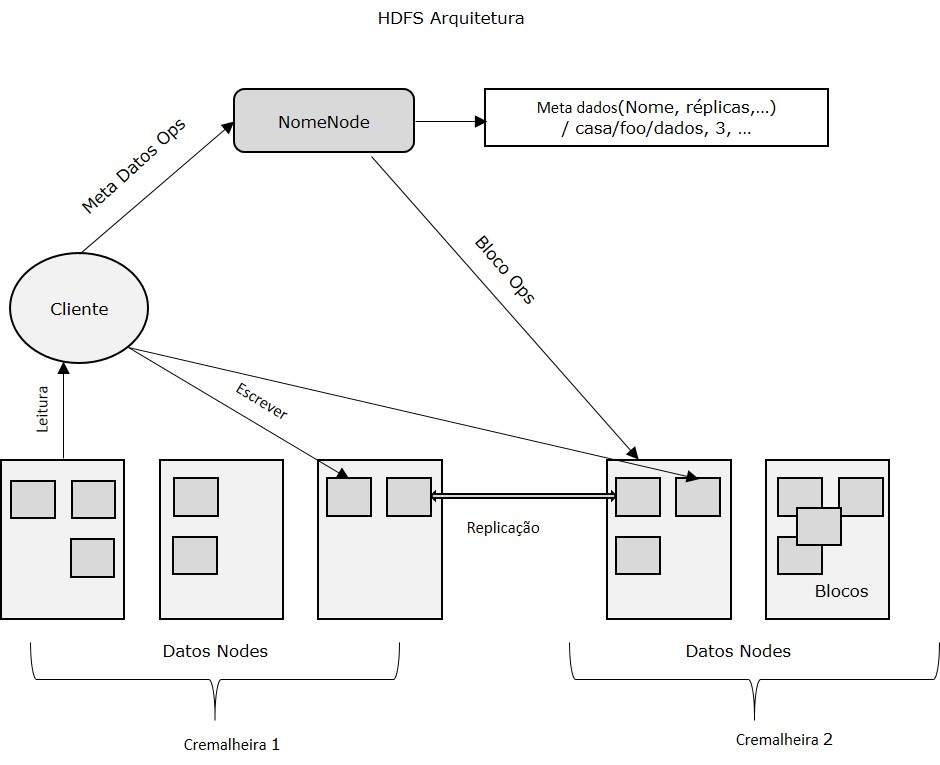
Los datos en un clúster Hadoop se dividen en partes más pequeñas llamadas bloques y, a continuación, se distribuyen en todo el clúster. Los bloques y copias de bloques, se almacenan en otros servidores en el clúster hadoop. Es decir, un archivo individual se almacena como bloques más pequeños que se replican entre varios servidores en el clúster.

Cada HDFS clúster tiene un número de DataNodes habiendo un DataNode para cada nodo en el clúster. Los DataNodes gestionan el almacenamiento que se adjunta a los nodos en los que se ejecutan. Cuando se divide un archivo en bloques, éstos se almacenan en un conjunto de DataNodes que se distribuyen en todo el clúster. Los DataNodes son responsables de servir las solicitudes de lectura y escritura de los clientes en el sistema de archivos, y también gestionan la creación, supresión y replicación de bloques.

Un clúster HDFS da soporte a NameNodes, un NameNode activo y un NameNode en espera, que es la configuración común para la alta disponibilidad. El NameNode regula el acceso a los archivos por parte de los clientes, y rastrea todos los archivos de datos en HDFS. El NameNode determina la correlación de bloques con DataNodes y gestiona operaciones como abrir, cerrar y renombrar archivos y directorios. Toda la información del NameNode se almacena en memoria, lo que permite tiempos de respuesta rápidos al añadir almacenamiento o leer solicitudes. El NameNode es el repositorio para todos los metadatos de HDFS, y los datos de usuario nunca fluyen a través del NameNode. Un despliegue típico de HDFS tiene un sistema dedicado que ejecuta solo el NameNode, porque el debido a almacena los metadatos en memoria. Si el sistema que ejecuta el NameNode falla, se pierden todos los metadatos del clúster, por lo que dicho equipo suele ser más robusto que el resto del clúster.

## HDFS Arquitectura

A continuación se muestra la arquitectura de un sistema de archivos Hadoop.



HDFS sigue el maestro-esclavo y arquitectura que tiene los siguientes elementos:

### Namenode

El namenode es el hardware básico que contiene el sistema operativo GNU/Linux y el software namenode. Es un software que puede ejecutarse en hardware. El sistema que tiene la namenode actúa como el servidor maestro y no las siguientes tareas:

* Administra el espacio de nombres del sistema de archivos.
* Además, ejecuta las operaciones del sistema de archivos como el cambio de nombre, cierre y apertura de archivos y directorios.

### Datanode

La datanode es un hardware de productos básicos con el sistema operativo GNU/Linux y software datanode. Para cada nodo (Commodity hardware/Sistema) de un clúster, habrá un datanode. Estos nodos gestionar el almacenamiento de datos de su sistema.

* Datanodes realizan operaciones de lectura y escritura de los sistemas de archivos, como por petición del cliente.

# Máquina Virtual

Una máquina virtual es un archivo de PC, que suele denominarse “imagen”, que se comporta igual que un equipo real. Dicho de otro modo, es crear un equipo dentro de un equipo. Se ejecuta en una ventana, igual que cualquier otro programa, y permite que el usuario final tenga la misma experiencia en una máquina virtual que tendría en el propio sistema operativo host. La máquina virtual se sitúa en un espacio aislado del resto del sistema, es decir, el software de la máquina virtual no puede interferir con el equipo en sí. Esto crea un entorno ideal para probar otros sistemas operativos, incluidas versiones beta, acceder a datos infectados por virus, crear copias de seguridad de sistemas operativos y ejecutar software o aplicaciones en sistemas operativos para los que no se habían creado inicialmente.

Se pueden ejecutar varias máquinas virtuales a la vez en un mismo equipo físico. Para servidores, los diversos sistemas operativos se ejecutan en paralelo con un fragmento de software denominado hipervisor para administrarlos, mientras que los equipos de escritorio suelen implementar un solo sistema operativo para ejecutar los demás sistemas operativos en ventanas de programa.

# Linux OS

Al igual que Windows XP, Windows 7, Windows 8 y Mac OS X, Linux es un sistema operativo. Un sistema operativo es un software que administra todos los recursos de hardware asociados con su computadora de escritorio o computadora portátil. Para decirlo simplemente, el sistema operativo maneja la comunicación entre su software y su hardware. Sin el sistema operativo (a menudo denominado "SO"), el software no funcionaría.

El sistema operativo se compone de varias piezas:

* **El Bootloader:** el software que administra el proceso de arranque de su computadora. Para la mayoría de los usuarios, esto simplemente será una pantalla emergente que aparecerá y finalmente desaparecerá para iniciarse en el sistema operativo.
* **El kernel:** esta es la única pieza del conjunto que en realidad se llama "Linux". El kernel es el núcleo del sistema y administra la CPU, la memoria y los dispositivos periféricos. El kernel es el nivel "más bajo" del sistema operativo.
* **Daemons:** estos son servicios en segundo plano (impresión, sonido, programación, etc.) que se inician durante el inicio o después de iniciar sesión en el escritorio.
* **The Shell:** Probablemente has escuchado mencionar la línea de comandos de Linux. Este es el shell - un proceso de comando que le permite controlar la computadora a través de comandos escritos en una interfaz de texto. Esto es lo que, en un momento, alejó más a las personas de Linux (suponiendo que tuvieran que aprender una estructura de línea de comando aparentemente arcaica para hacer que Linux funcione). Este ya no es el caso. Con Linux de escritorio moderno, no hay necesidad de tocar la línea de comando.
* **Servidor gráfico:** este es el subsistema que muestra los gráficos en su monitor. Se conoce comúnmente como servidor X o simplemente "X".
* **Entorno de escritorio:** esta es la pieza del rompecabezas con la que los usuarios realmente interactúan. Hay muchos entornos de escritorio para elegir (Unity, GNOME, Cinnamon, Enlightenment, KDE, XFCE, etc.).

# Ethernet Switch

Un conmutador Ethernet es un dispositivo de interconexión [LAN](http://www.erg.abdn.ac.uk/users/gorry/course/intro-pages/lan.html) que opera en la capa de enlace de datos (capa 2) del [modelo de referencia OSI](http://www.erg.abdn.ac.uk/users/gorry/course/intro-pages/osi.html) . Un conmutador es fundamentalmente similar a un [puente](http://www.erg.abdn.ac.uk/users/gorry/course/lan-pages/bridge.html) , pero generalmente admite una mayor cantidad de segmentos de LAN conectados y tiene una capacidad de gestión más rica.

Las LAN modernas han reemplazado cada vez más a los medios compartidos con un medio conmutado, instalando conmutadores y puentes Ethernet en lugar de concentradores y repetidores. Estos dividen de forma lógica el tráfico para viajar solo a través de los segmentos de red en la ruta entre el origen y el destino. Esto reduce el desperdicio de ancho de banda que resulta de enviar el paquete a partes de la red que no necesitan recibir los datos. También hay beneficios de una seguridad mejorada (los usuarios tienen menos capacidad de acceder a los datos de otros usuarios), una mejor administración (la capacidad de controlar quién recibe qué información (es decir, LAN virtuales) y limitar el impacto de los problemas de red) y el capacidad de operar algunos enlaces en dúplex completo (en lugar de medio dúplex requerido para acceso compartido).

# Fortran

El FORTRAN (FORmula TRANslator) es un lenguaje de programación muy potente. Es el primer lenguaje científico, siendo desarrollado a últimos de la década de los 50, pero es aun ampliamente utilizado en aplicaciones científicas y de ingeniería. La última versión de FORTRAN actualmente es la version FORTRAN 95, que reemplazó a la versión FORTRAN 90, que a su vez reemplazó al FORTRAN 77 a comienzos de la década de los 90. El FORTRAN 90 incorporá nuevas estructuras y características en su sintáxis, si bien aún es compatible con la anterior versión. Este curso es muy introductorio; para una exposición más detallada habrá que consultar algún manual de FORTRAN.

En los primeros tiempos de la informática cualquier cálculo que implicara la evaluación de fórmulas matemáticas había de hacerse mediante complicados programas, que traducían esas fórmulas al lenguaje del ordenador, muy primitivo. EL FORTRAN es un lenguaje de alto nivel, orientado a facilitar las cosas al usuario, y que permite escribir el programa de manera casi idéntica a como se escriben las fórmulas en un papel.

En FORTRAN las instrucciones que el usuario transmite al ordenador se encuentran en un fichero de texto (en código ASCII), que es necesario escribir (utilizando un editor de textos cualesquiera). El nombre del fichero sigue las convenciones del sistema operativo del ordenador. En DOS un programa FORTRAN suele tener la extensión '.FOR', por ejemplo, 'CALCULO.FOR'; en Unix es tradicional que la extensión sea '.f', por ejemplo, 'calculo.f'. Si la versión de FORTRAN es la 90, es conveniente utilizar la extension '.f90'. Los nombres de los programas son libres (salvo ciertas limitaciones de longitud y tipos especiales de caracteres que no pueden aparecer en el nombre del fichero).

Al contrario que el BASIC, que es un lenguaje interpretado, el FORTRAN es un lenguaje compilado.

## Escritura del programa

Las instrucciones del programa se escriben en el fichero dispuestas en líneas. En FORTRAN 90 el texto ha de tener como mucho 132 caracteres en cada línea y, al contrario que en FORTRAN 77, donde las columnas de la 1 a la 5 se reservan para que se puedan incluir etiquetas de señalización (con vistas a la posibilidad de que el flujo del programa vuelva a esa línea desde otra parte del programa) y la columna 6 se reserva para incluir un símbolo que le indique al compilador que esa línea es la continuación de la línea anterior, en FORTRAN 90 se pueden utilizar las columnas libremente. Una línea que continúa en la siguiente se indica mediante el símbolo &. Un ejemplo de programa sencillo es:

PROGRAM EJEMPLO   
REAL :: A, B, C   
A=1   
B=A+1   
PRINT \*, A,B   
C=A+B+1   
PRINT \*,'C es igual a=',C   
END

## Conceptos básicos del lenguaje

## VARIABLES  El primer concepto importante es el de variable. Una variable es una zona reservada en la memoria del ordenador, a la que se asigna un nombre. En esta zona podemos guardar un número, o bien una secuencia o cadena de caracteres. Las variables en las que se pueden almacenar números se llaman numéricas, y pueden tener como nombres una combinación de caracteres, si bien el primero no puede ser un número; el número de caracteres varía según el compilador.

### ENTRADAS Y SALIDAS

Las entradas de datos a un programa y las salidas de datos producidos por éste se realizan en FORTRAN por medio de las instrucciones READ y WRITE. Para escribir también se utiliza la instrucción PRINT. La sintáxis de las instrucciones READ y WRITE es:

READ (unidad, formato) lista de variables

WRITE (unidad, formato) lista de variables

# Lenguaje de programación C

El lenguaje de programación en C, es un lenguaje conocido como de alto nivel, una de las características del lenguaje de programación en C, es que es un lenguaje estructurado, lo que permite generar código claro y sencillo, ya que está basado en la modularidad.

El lenguaje de programación en C, está estructurado en tres partes fundamentales, las cuales son, una librería estándar, un programa compilador y un preprocesador.

La librería estándar en el lenguaje de programación C, no es más que librerías realizadas en código objeto y puede haber sido realizada en otro lenguaje diferente que el C. Estas librerías se deben colocar en un programa de lenguaje programación en C, en la instrucción conocida como INCLUDE.

El programa compilador en el lenguaje de programación en C, es el que tiene como función traducir las instrucciones del programa fuente en C al lenguaje conocido por las computadoras u ordenadores, el llamado lenguaje máquina. El programa compilador, depura y detecta los posibles errores en el lenguaje fuente, y es capaz de notificar el error ocurrido al programador, mediante un mensaje de texto.

# C++

**C++** es un [lenguaje de programación](https://es.wikipedia.org/wiki/Lenguaje_de_programaci%C3%B3n) diseñado a mediados de los años 80 por [Bjarne Stroustrup](https://es.wikipedia.org/wiki/Bjarne_Stroustrup). La intención de su creación fue el extender al lenguaje de programación [C](https://es.wikipedia.org/wiki/C_(lenguaje_de_programaci%C3%B3n)) mecanismos que permiten la manipulación de [objetos](https://es.wikipedia.org/wiki/Objeto_(programaci%C3%B3n)). En ese sentido, desde el punto de vista de los [lenguajes orientados a objetos](https://es.wikipedia.org/wiki/Programaci%C3%B3n_orientada_a_objetos), el C++ es un lenguaje híbrido.

Posteriormente se añadieron facilidades de [programación genérica](https://es.wikipedia.org/wiki/Programaci%C3%B3n_gen%C3%A9rica), que se sumaron a los paradigmas de [programación estructurada](https://es.wikipedia.org/wiki/Programaci%C3%B3n_estructurada) y [programación orientada a objetos](https://es.wikipedia.org/wiki/Programaci%C3%B3n_orientada_a_objetos). Por esto se suele decir que el C++ es un [lenguaje de programación multi-paradigma](https://es.wikipedia.org/wiki/Lenguaje_de_programaci%C3%B3n_multiparadigma). Actualmente existe un estándar, denominado ISO C++, al que se han adherido la mayoría de los fabricantes de compiladores más modernos. Existen también algunos intérpretes, tales como ROOT.

## Ejemplo de código utilizando C++

A continuación se cita un programa de ejemplo [Hola mundo](https://es.wikipedia.org/wiki/Hola_mundo) escrito en C++:

#include *<iostream>*

**using** **namespace** std;

int main()

{

cout << "Hola mundo" << endl;

**return** 0;

}

## Tipos de Datos

C++ tiene los siguientes [tipos](https://es.wikipedia.org/wiki/Tipo_de_datos) fundamentales:

* Caracteres: char (también es un entero), wchar\_t
* Enteros: short, int, long, long long
* Números en coma flotante: float, double, long double
* Booleanos: bool
* Vacío: void

# IDE

Un entorno de desarrollo integrado o entorno de desarrollo interactivo, en inglés Integrated Development Environment (IDE), es una [aplicación informática](https://es.wikipedia.org/wiki/Aplicaci%C3%B3n_inform%C3%A1tica) que proporciona servicios integrales para facilitarle al [desarrollador](https://es.wikipedia.org/wiki/Desarrollador_de_software) o [programador](https://es.wikipedia.org/wiki/Programador) el [desarrollo de software](https://es.wikipedia.org/wiki/Desarrollo_de_software).

Normalmente, un IDE consiste de un [editor de código fuente](https://es.wikipedia.org/wiki/Editor_de_c%C3%B3digo_fuente), herramientas de construcción automáticas y un [depurador](https://es.wikipedia.org/wiki/Depurador). La mayoría de los IDE tienen [auto-completado inteligente de código](https://es.wikipedia.org/wiki/IntelliSense) (IntelliSense). Algunos IDE contienen un [compilador](https://es.wikipedia.org/wiki/Compilador), un [intérprete](https://es.wikipedia.org/wiki/Int%C3%A9rprete_(inform%C3%A1tica)), o ambos, tales como [NetBeans](https://es.wikipedia.org/wiki/NetBeans) y [Eclipse](https://es.wikipedia.org/wiki/Eclipse_(software)); otros no, tales como [SharpDevelop](https://es.wikipedia.org/wiki/SharpDevelop) y [Lazarus](https://es.wikipedia.org/wiki/Lazarus).

El límite entre un IDE y otras partes del entorno de desarrollo de software más amplio no está bien definido. Muchas veces, a los efectos de simplificar la construcción de la [interfaz gráfica de usuario](https://es.wikipedia.org/wiki/Interfaz_gr%C3%A1fica_de_usuario) (GUI, por sus siglas en inglés) se integran un sistema controlador de versión y varias herramientas. Muchos IDE modernos también cuentan con un navegador de clases, un buscador de objetos y un diagrama de jerarquía de clases, para su uso con el [desarrollo de software orientado a objetos](https://es.wikipedia.org/wiki/Programaci%C3%B3n_orientada_a_objetos).

Algunos IDE soportan múltiples lenguajes, tales como [GNU Emacs](https://es.wikipedia.org/wiki/GNU_Emacs) basados en [C](https://es.wikipedia.org/wiki/C_(lenguaje_de_programaci%C3%B3n)) y [Emacs Lisp](https://es.wikipedia.org/wiki/Emacs_Lisp), y [Eclipse](https://es.wikipedia.org/wiki/Eclipse_(software)), IntelliJ IDEA, MyEclipse o [NetBeans](https://es.wikipedia.org/wiki/NetBeans), todos basados en [Java](https://es.wikipedia.org/wiki/Java_(lenguaje_de_programaci%C3%B3n)), o [MonoDevelop](https://es.wikipedia.org/wiki/MonoDevelop), basados en C#.

Normalmente, el soporte para lenguajes alternativos regularmente es proveído por un plug-in, permitiéndoles ser instalados en el mismo IDE, al mismo tiempo.[4](https://es.wikipedia.org/wiki/Entorno_de_desarrollo_integrado#cite_note-4)​ [Eclipse](https://es.wikipedia.org/wiki/Eclipse_(software)), y [Netbeans](https://es.wikipedia.org/wiki/Netbeans) tienen plugins para [C](https://es.wikipedia.org/wiki/C_(lenguaje_de_programaci%C3%B3n))/[C++](https://es.wikipedia.org/wiki/C%2B%2B), [Ada](https://es.wikipedia.org/wiki/Ada_(lenguaje_de_programaci%C3%B3n)), (por ejemplo AdaGIDE), [Perl](https://es.wikipedia.org/wiki/Perl), [Python](https://es.wikipedia.org/wiki/Python_(lenguaje_de_programaci%C3%B3n)), [Ruby](https://es.wikipedia.org/wiki/Ruby_(lenguaje_de_programaci%C3%B3n)), y [PHP](https://es.wikipedia.org/wiki/PHP), los cuales son seleccionados entre extensión de archivos, ambientes o ajustes de proyectos.

## Programación visual

La [programación visual](https://es.wikipedia.org/wiki/Programaci%C3%B3n_visual) es un marco de usuario en la que generalmente se requiere una IDE. Los IDE visuales les permiten a los usuarios crear nuevas aplicaciones de programación en movimiento, bloques de construcción, o nodos de código para crear diagramas de flujo o diagramas de estructura que luego son compilados o interpretados. Estos diagramas de flujo muchas veces se basan en el [lenguaje de modelado unificado](https://es.wikipedia.org/wiki/Lenguaje_de_modelado_unificado).

Esta interfaz ha sido popularizada con los [Lego Mindstorms](https://es.wikipedia.org/wiki/Lego_Mindstorms), y se ha mantenido activa por un número de compañías deseando capitalizar el poder de los buscadores personalizados como los fundados en [Mozilla](https://es.wikipedia.org/wiki/Mozilla_Application_Suite). [KTechlab](https://es.wikipedia.org/wiki/KTechlab) apoya el flowcode y es un IDE de código abierto (opensource) y un simulador para desarrollar software para micro-controladores.

La programación visual también es responsable del poder de la distribución de software ([LabVIEW](https://es.wikipedia.org/wiki/LabVIEW) y software EICASLAB). Un primitivo sistema visual de programación, Max, fue modelado a partir de un sintetizador de diseño análogo siendo desarrollado para utilizar el desempeño de la música en tiempo real desde los [años 1980](https://es.wikipedia.org/wiki/A%C3%B1os_1980). Otro ejemplo primitivo fue Prograph, un programa a base de flujo de datos, originalmente desarrollado para la [Macintosh](https://es.wikipedia.org/wiki/Macintosh). El ambiente de programación gráfica "Grape" es usado para programar qfix robot kits.