Mphasis
Unleash the Next

Lightening-Speed Data Processing with Big Data

Whitepaper by:

Srini Challa
Associate Vice President, Mphasis

Anand Babu
Module Lead, Mphasis

What is Data Processing?

Data processing is the collection, manipulation and transformation of data items to produce meaningful information. It can also be described as converting data or information from one format to another, usually from the format of a source system into the required format of a new destination system. The usual process involves converting documents, but data conversions sometimes involve the conversion of a program from one computer language to another to enable the program to run on a different platform. The usual reason for this data transformation is the adoption of a new system that's different from the previous one.

Why to Process?

Good question. Why should one need to manipulate or transform the data from its original state? The reasons could be different for different purposes. We shall group the reasons into various categories as follows.

Readability

- Take an example where you want to collect the feedback survey about your company and present to your top management. The web service typically gives the survey results in XML or JSON format. You need to transform them into a tabular format and then present it to the management at required granularity.
- One may say that we can use No SQL databases for this. Yes, agreed. However, in that option also you need to transform the data from files (JSON or XML) to a No SQL Database.
- Typical file format conversions falls under this category. E.g. Unstructured to structured or Semi-structured to structured.

Consolidation

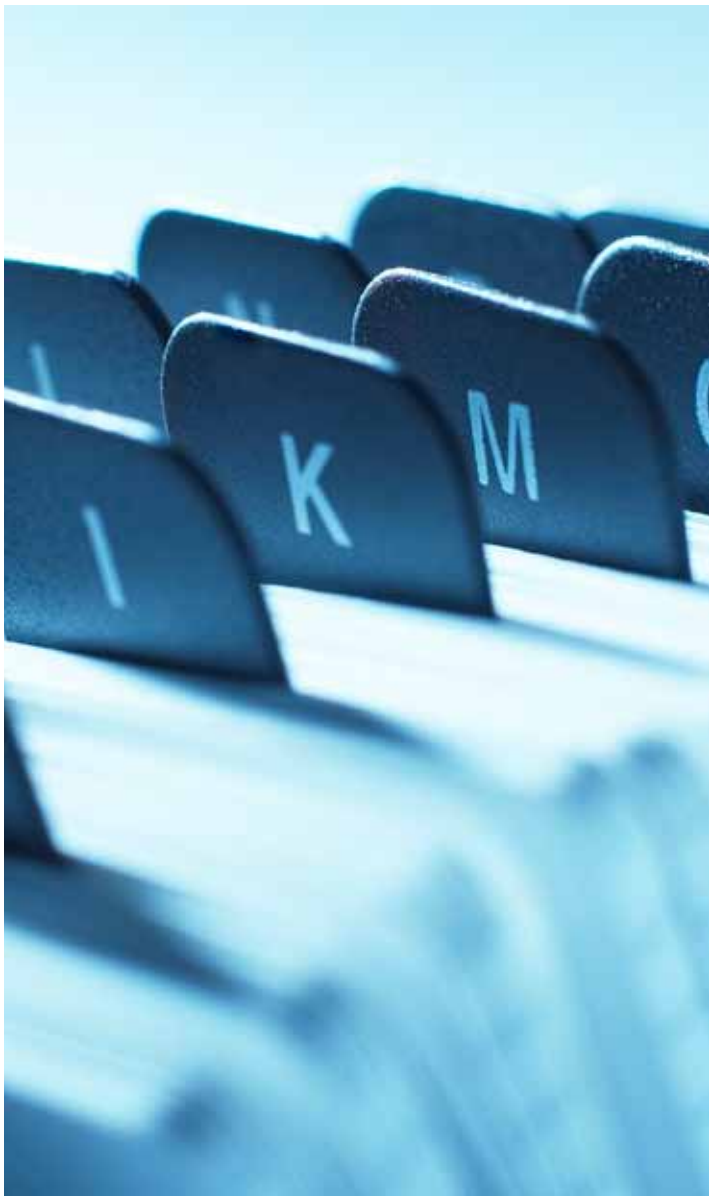
- Take an example where you want to collect the data from various departments of the company and analyze the data at the enterprise level. In this case, all data is in tabular format only. But they might have been stored in different grains, keys and data models. Hence, you need to transform the data into a common platform and consolidate them by common keys or ids.
- Another complex case could be consolidating enterprise internal data (tabular format) with external data (customer survey feedbacks, tweets … etc), which is semi-structured.

Performance

- Typically, Data Warehouse stores the data in aggregated formats, multi-dimensional cubes and in various other formats to improve the query performance and decrease the response time to the user.

Business/technical reasons

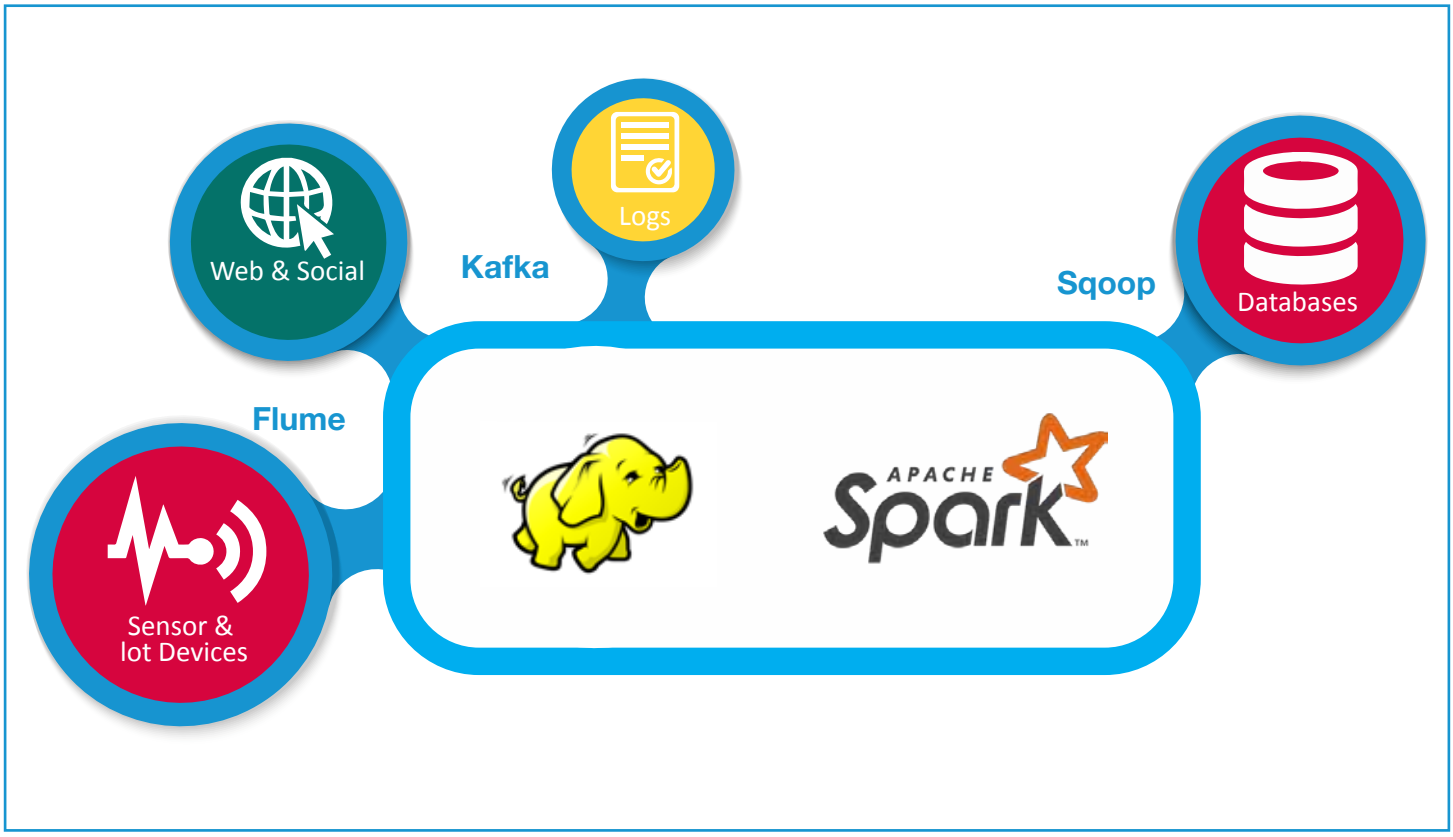
- These are other categories where the data transformation is needed for a given tool under the hood or for any business reasons.



Why Big Data?

The Big Data Technologies can store massive data (more than petabytes) and can process it faster with high performance (thanks to its distributed architecture). Though the Big Data Technologies started from web and internet processing, it has already proved its efficiency in processing various kinds of data (Variety) at greater performance (Velocity). Rest of this white paper describes the best practices and framework of implementing Data Transformations using Big Data Technologies (Hadoop and Spark).

Big Data Framework for Data Processing



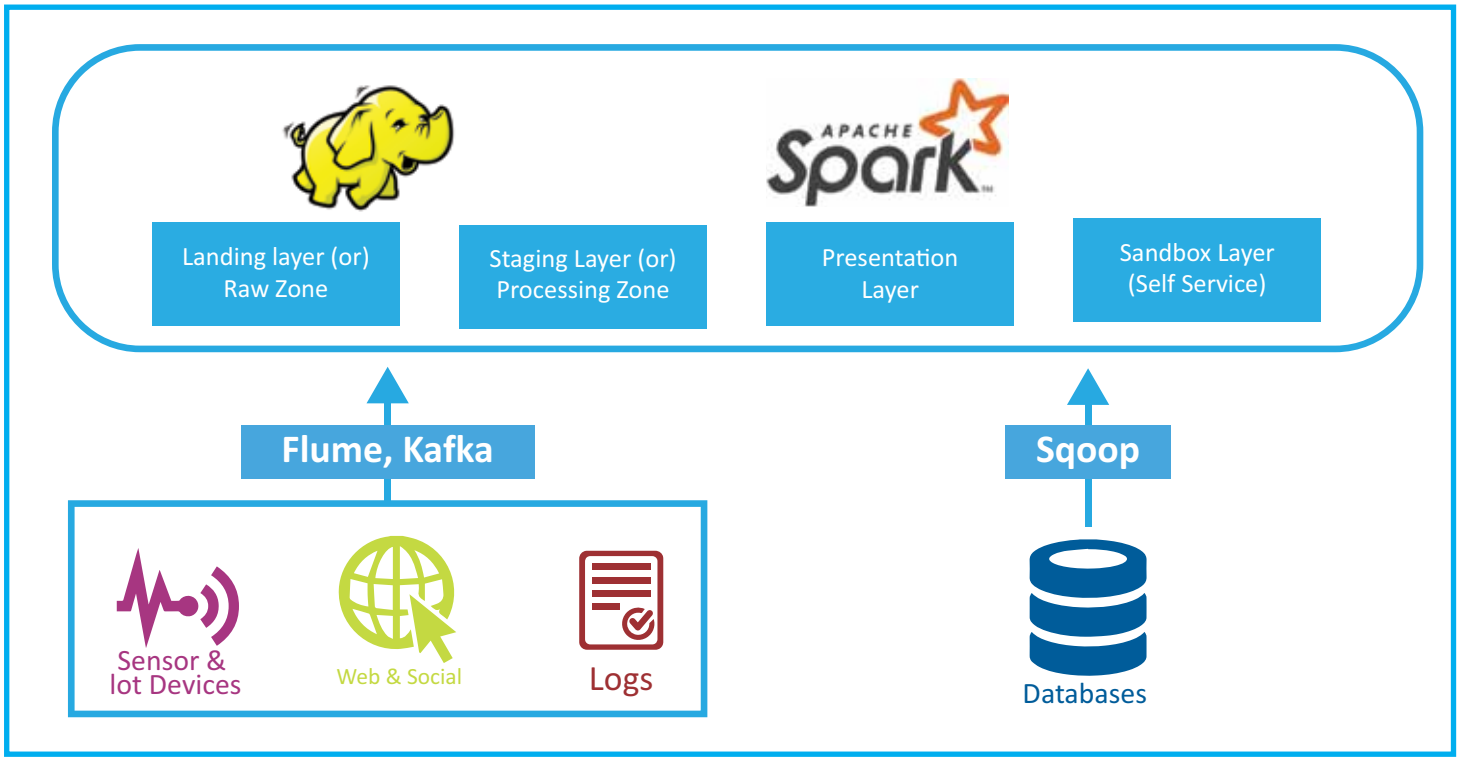
Data Ingestion

The first step is to bring your data to the Big Data Eco System. That is called data ingestion. There are several ingestion tools available in the market (open source) that are developed for different kinds of source systems. E.g. Sqoop can bring the data from various databases and load into HDFS (Hadoop Distributed File System). Flume and Kafka can stream the unstructured or semi-structured data into the HDFS at real time.

Ingestion tools typically don't transform the data but store it as-is into HDFS. Hence the raw data is preserved.

Layered Architecture

Let us take a deep dive on what happens after ingesting data into Hadoop Eco System. A layered architecture enables smooth transformation of the data to the format required by the final destination, with easy debugging and lineage.



Landing Layer

This is also referred as raw zone. The ingested data is stored in here with 'as-is' raw format, without applying any transformations. Usually the file names are suffixed/prefixed by the timestamps and also the folders are structured by dates to easily identify the files that are received in a particular date/time. This is useful for debugging and also makes identifying the delta (or unprocessed data) easier for the next process. Following are the different tasks performed at the landing layer.

- Ingestion completeness check
- File integrity check
- Checksums
- File format and delimiter check

Staging Layer

The actual data transformation starts from here. Business rules are applied, various data sources are integrated and finally data is brought to the suitable grain/format of the presentation layer. Following are the different tasks performed at the staging layer.

1. Cleansing
   - Data quality check
   - Data profiling

2. Transformations
   - Data parsing
   - Data conversion
   - Format conversion
   - Data type conversion
   - Data enrichment
   - File Format
   - Joins, filters, lookups

3. Controlling
   - Data configuration
   - Notifications

4. Post Transformations
   - Reconciliation
   - Partition
   - Aggregation
   - Encryption

Spark can play a vital role in transforming the data between the layers at lightning fast. It distributes and loads the data “in-memory” across the nodes of the cluster, also called partitions. It applies transformation logic on the partitions (minimizing disk i/o). The driver program coordinates all partitions.

Spark also gives adequate control to the programmer in terms of which data item needs to be copied/duplicated across the partitions (broadcast variables), ability to control the persistence (e.g. Spark can keep an entire data frame in memory throughout the execution) and lazy evaluation. For example, take a program that produces a series of map reduce jobs internally. In a typical map reduce world, each MR writes/distributes the (intermediate) data across the disks of the cluster. The Map Reduce jobs handshake the intermediate data via disks. Whereas Spark uses “in-memory” processing technology thus minimizing the disk i/o. Data bricks proved that Spark job can be 100x times faster than a map reduce job.

Presentation Layer

This is the final layer for the business users and external applications to consume the data and present to the external world. Typically, the self-serviced business user queries, data scientists, data visualization tools, API and Business Intelligence tools point to the presentation layer and consume data.

Sandbox Layer

The business users and data scientists typically have Read Only (RO) access to the presentation layer. They can only query the data and can’t save intermediate results and queries for further reuse. The Sandbox layer is given to these users with Read Write (RW) access so that they can store intermediate data, create views, temp tables and queries. Usually, the presentation and sandbox layers are hosted on same technology so that users can easily join the tables between both the layers and reference the presentation layer easily. A strong Data Governance is required in order to make the sandbox successful and ensure that it is used for the defined purpose.

The Technology Stack

The following table lists various Big Data tools and technologies that are used for the concepts referenced in this article. This is not a complete exhaustive list of tools, however can be used as a reference.

Technology	Functionality	Details
Sqoop	Data Ingestion	Apache Sqoop is a tool designed for efficiently transferring bulk data between Apache Hadoop and structured data stores such as relational databases.
Flume	Data Ingestion	Apache Flume is a tool designed for streaming the unstructured data into the Hadoop Eco System. It can read massive data from web logs, channels in real time.
Kafka	Data Ingestion Message Broker	Kafka is designed to allow a single cluster to serve as the central data backbone for a large organization. Data streams are partitioned and spread over a cluster of machines to allow data streams larger than the capability of any single machine and to allow clusters of coordinated consumers.
Hive	Data Store Data Processing Data Analysis	The Apache Hive (TM) data warehouse software facilitates querying and managing large datasets residing in distributed storage.
Impala	Data Store	Cloudera Impala is Cloudera's open source massively parallel processing SQL query engine for data stored in a computer cluster running Apache Hadoop
No SQL Databases	Data Store	No SQL Databases can store unstructured data on distributed cluster.
HBase	Data Store	Apache HBase is an open-source, distributed, versioned, column-oriented data store modeled after Google's Bigtable.
Spark	Eco System Framework	Apache Spark is a fast and general engine for large-scale data processing. It offers high-level APIs in Java, Scala and Python as well as a rich set of libraries, including stream processing, machine learning, and graph analytics.
Hadoop	Eco System Framework	The Apache Hadoop software library is a framework that allows for the distributed processing of large data sets across clusters of computers using simple programming models.
Oozie	Workflow Management	Oozie is a workflow scheduler system to manage Apache Hadoop jobs. Oozie is integrated with the rest of the Hadoop stack supporting several types of Hadoop jobs out of the box
Falcon	Workflow Management Metadata Management	Apache Falcon is a data processing and management solution for Hadoop designed for data motion, coordination of data pipelines, lifecycle management, and data discovery. Falcon enables end consumers to quickly onboard their data and its associated processing and management tasks on Hadoop clusters.
Pig	Data Processing Data Analysis	Apache Pig is a platform for analyzing large data sets that consists of a high-level language for expressing data analysis programs, coupled with infrastructure for evaluating these programs. The salient property of Pig programs is that their structure is amenable to substantial parallelization, which in turns enables them to handle very large data sets.

Srini Challa

Associate Vice President – Mphasis Analytics Practice

Srini Challa has 16 years of experience in designing and developing analytics solutions using Big Data and DW BI methodologies. During this journey, he has played multiple roles such as Data Architect, BI Architect, ETL Architect, DW BI Solutions architect and Big Data Solutions Architect. Srini is currently working for Mphasis Analytics Practice, leading the Big Data COE and responsible for NEXT Labs delivery. Srini is a Cloudera certified Spark & Hadoop developer (CCA-175) and Hortonworks certified developer (HDPCD).

Anand Babu

Module Lead – Mphasis Analytics Practice

Anand has 7+ years of experience in designing and developing software applications using Big Data technologies, such as Hadoop and Spark. He is experienced in Telecom and BFSI Business areas.

Anand has been working for Mphasis as a module lead for six months now. He handles data transformation and presentation layer in his current projects.

About Mphasis

Mphasis is a global Technology Services and Solutions company specializing in the areas of Digital and Governance, Risk & Compliance. Our solution focus and superior human capital propels our partnership with large enterprise customers in their Digital Transformation journeys and with global financial institutions in the conception and execution of their Governance, Risk and Compliance Strategies. We focus on next generation technologies for differentiated solutions delivering optimized operations for clients.

For more information, contact: marketinginfo@mphasis.com

USA
460 Park Avenue South
Suite #1101
New York, NY 10016, USA
Tel.: +1 212 686 6655

UK
88 Wood Street
London EC2V 7RS, UK
Tel.: +44 20 8528 1000

INDIA
Bagmane World Technology Center
Marathahalli Ring Road
DoddanakundhiVillage, Mahadevapura
Bangalore 560 048, India
Tel.: +91 80 3352 5000

www.mphasis.com

VAS 01/12/16 US LETTER BASIL 4118