Universitat de Lleida

# SISTEMES INTENSIUS DE PROCESSAMENT DE DADES

## Final Report
## Map Reduce Tasks

**Mariano Garralda Barrio**
**Oscar Ujaque Perez**

**ESCOLA POLITÈCNICA SUPERIOR, UdL**
**LLEIDA, JULY 2016**

## 1.Project Structure

The project delivered is composed by several folders. In each folder we can find:

- **finalreport**: it contains a brief report explaining the tasks done in the project and an image which explains the design of the project.

- **inputdata:** it contains the input file to analyze in json format.

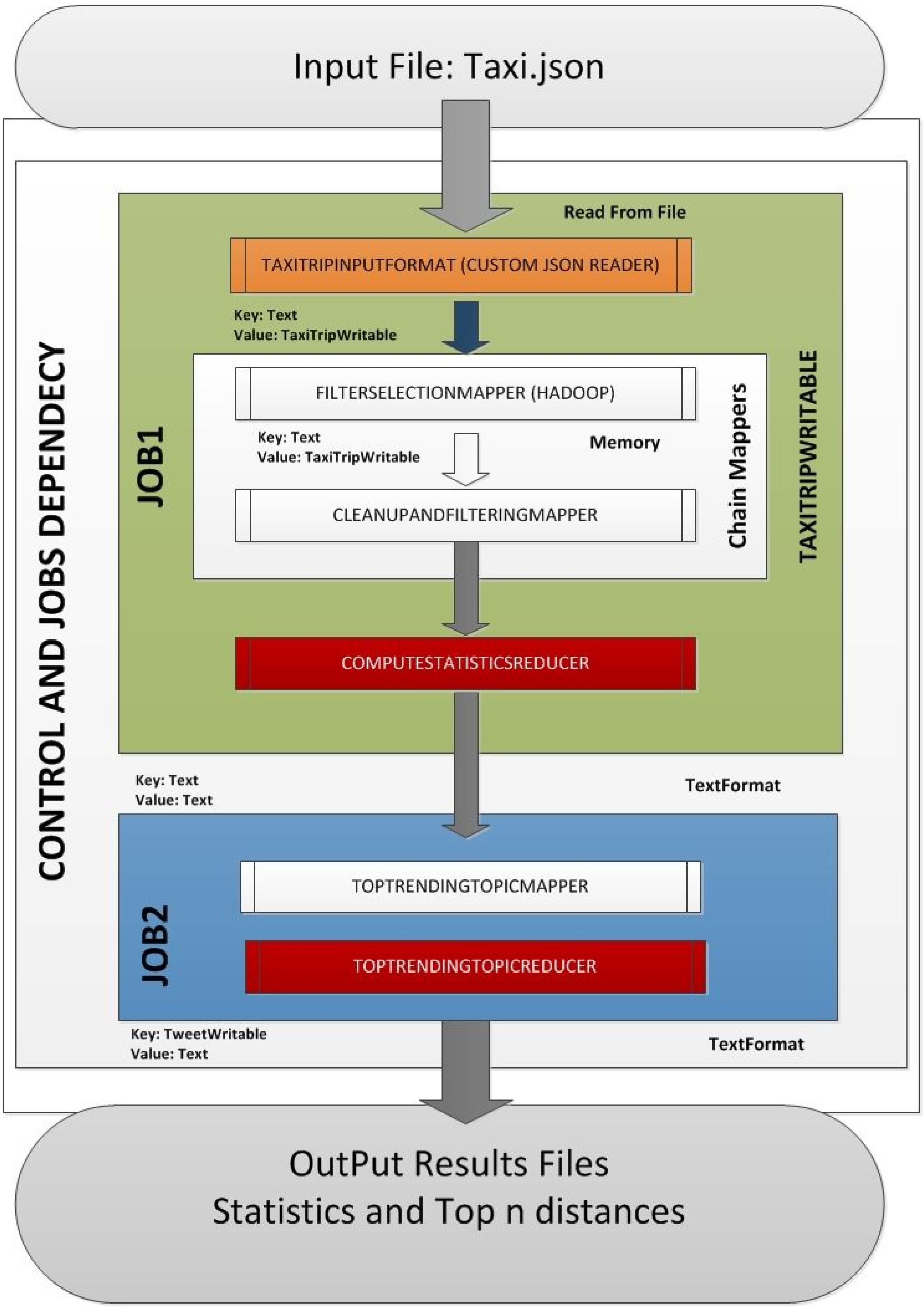
- **results**: if contains the output thrown by the execution of the program.

- **src/eps/mareduce:** it contains all the source code.

## 2. Project Design

As we can see in the below image the project is composed  by two jobs and next steps:

1. In the first job, and firstly, it starts by reading the json file with TaxiTripInputFormat class.
2. Once it is read it uses two mappers joined by a ChainMapper.
    a. The first mapper “CleanUpAndFilteringMapper” stores all the values of the input.json in a TaxiTripWritable class. It also computes the distance, the time and the velocity of each trip of each input.
    b. The second mapper, TopDistancesMapper, writes in context the N greatest values of distances of each trip.
3. The reducer of these mappers computes some statistics. In these statistics we can find the following fields for each TaxiDriver:
    i. Total Distance
    ii. Max Distance
    iii. Distance average
    iv. Max velocity
    v. Velocity average
    vi. Max Trip time
    vii. Trip time Average
    viii. Number of trips
4. The second job computes the Top N max distances travelled.
    a. The mapper only writes the top N best distances travelled in context using NullWritable method.
    b. The reduce also writes the top N best distances travelled in context also using NullWritable method. The results are ordered by distance.
5. Finally, the results are stored in results folder.


Input File: Taxi.json
CONTROL AND JOBS DEPENDECY
JOB1
Read From File
TAXITRIPINPUTFORMAT (CUSTOM JSON READER)
Key: Text
Value: TaxiTripWritable
FILTERSELECTIONMAPPER (HADOOP)
Key: Text
Value: TaxiTripWritable
Memory
CLEANUPANDFILTERINGMAPPER
Chain Mappers
TAXITRIPWRITABLE
COMPUTESTATISTICSREDUCER
Key: Text
Value: Text
TextFormat
JOB2
TOPTRENDINGTOPICMAPPER
TOPTRENDINGTOPICREDUCER
Key: TweetWritable
Value: Text
TextFormat
OutPut Results Files
Statistics and Top n distances

## 3. Advanced structures used

Several advanced structures have been used in the project:

- Custom Input File Format: For reading the json file.

- Custom Writables:
    - TaxiTripWritable: For storing the json data after reading
    - GpsPositionWritable: For storing the gps coordinates after reading.
    - ArrayWritable<GpsPositioWritable>:

- Chain Mappers: For chaining two mappers in first job.

- Mappers from hadoop: FieldSelectionMappers: For treating the input data.

- Job Control and dependencies: For running two jobs.

## 4. Commands:

To execute the program you have to type the following command:

```
$  yarn  jar  eps.mapreduce.MainTaskActivities  <Input>   <Output>   <TopN>
```