# Energy consumption in an SOA that combines Spark, MapReduce and HDFS

**Mint Mohamed Lemin Zeinebou, Noura AKNIN,Salihi Mohamed Lemin, Moustapha Mohamed Lemin**
LIROSA Laboratory Information Technology and Modeling Systems Research Unit Abdelmalek Essaadi University, Morocco
Faculty of Science Nouakchott University Nouakchott, Mauritania

**Article Info**

***Article history:***

Received Sept 28,2019
Revised Oct xx, 2019
Accepted Oct xx, 2019

***Keyword:***

Big data
DataNode
DVFS
HadoopHDFS
HadoopMapReduce
NameNode
Namenode_Secondaire
SOA
Spark

**ABSTRACT**

During any operation of management of large volumes of data, we must use many frameworks together which appears efficient but consumes more economic energies and material. In this article, we have made a comparison between two service-oriented architectures. Each one has, as purpose, to gather a set of advanced frameworks to manage the largest amount of data collected but with the least possible energy consumption. The first architecture is based on Spark and Hadoop HDFS. The second one is based on Hadoop MapReduce and Hadoop HDFS. All of them use the Dynamic voltage and frequency scaling technique to maximize the energy consumed. After the calculation of this consumption for the two architectures, we concluded with a third architecture that uses less consumption than the first two architectures.

.

***Corresponding Author:***

Mint Mohamed Lemin Zeinebou
LIROSA Laboratory Information Technology and Modeling Systems Research Unit
Abdelmalek Essaadi University, Morocco
Email: zeinebou.l7aj@gmail.com

## 1. INTRODUCTION

The calculation of the energy consumption is a very important goal in order to determine everything that has been used up to the end of the requested work. In this article, we use frameworks to manage the maximum possible volumes of Big Data. We chose a technology to maximize the energy consumed, after, we applied this technique on service-oriented architectures to determine the architecture with the least energy.

The problem we had studied is that of limiting the capacity of most Frameworks in the field of Big Data. These frameworks can not, indeed, manage more petabyte of data whereas the current size of big data exceeds the zettabyte of data.

We started our research by studying each of these terms individually, for Big Data, we noticed the huge volume evolution of Big Data or The idea that a single Hadoop is insufficient to manage Big Data in this way. moment, in parallel we studied the components of SOA to discover that each service in the SOA kernel is able to handle part of a given processing and the possibility of splitting the data processing at the beginning of the core processing of this architecture and collect it at its end, then we found to replace this treatment with a big data analysis and each service with a big data management framework, which help us connect both SOA term and Big data. Then we compare the designs of these architecture via their energy consumption by using DVFS.

This research was based on several bricks that are: A. The Dynamic Voltage and Frequency Scaling (DVFS) is a commonly used technique that improves CPU utilization and power management by tuning frequency Of the CPU depending on the current load. The ideal DVFS mechanism can instantly change the voltage / frequency values. Since the 2.6.10 version of the Linux kernel, there are five different power schemes (governors) available for dynamically spreading the CPU frequency according to the CPU usage. Each governor favors either performance or energy efficiency [1].

B. Big Data: Begoli has conducted a brief overview of the state of the art in architectures and platforms for large-scale data analysis. The survey focused on the adoption of related technologies, platforms for knowledge discovery, and architectural taxonomies. Chen & al. Presented a thorough investigation on the Big Data. The topics covered by the survey cover related technologies, generation and acquisition of data, storage, applications and future prospects. Chen and Zhang also studied Big Data. Their work focused on opportunities and challenges, techniques and technologies, design principles and future research. Wu & al. provided a framework for the great exploration of Big Data mining. [15]. The authors proposed the HACE (Heterogeneous, Autonomous sources, Complex and Evolving relationships among data) theorem to characterize Big Data. The authors also presented a three-layer framework for big data processing, which is comprised of big data mining platform, semantics and application knowledge, and mining algorithms. Finally, Cuzzocrea et al. discussed Online Analytical Processing (OLAP) over big data, big data posting and privacy as part of big data research agenda [2]. For instance, there is the challenge of managing large amounts of data (large data), which is getting increasingly larger because of cheaper storage and evolution of digital data and information collection devices, such as cell phones, laptops, and sensors. For example, Facebook, a social networking site, is home to 40 billion photos, and Wal-Mart manages more than one million customer transactions every hour, feeding databases estimated at over 2.5 Petabytes. There are 4.6 billion mobile phone subscriptions worldwide and 1-2 billion people use the Internet [3]. C. The MapReduce programming model is implemented on our platform for simplified processing of large Web datasets with a parallel, distributed algorithm on the Hadoop cluster, The MapReduce implemented on Hadoop helps to shift processing jobs to other connected nodes if one fails, so, it's inherent in fault-tolerance. Compared with parallel relational database-management-systems (DBMS) which perform excellently in executing a variety of data-intensive query processing benchmark, the Hadoop ecosystem is more optimized for computationally intensive operations [4]. D.HDFS (Hadoop Distributed File System) constituted the part of storage of Hadoop, Both HDFS and MapReduce are based on a master–slave architecture .In HDFS, the master, referred to as the Name Node is responsible for maintaining metadata relating to the distribution of files (partitioned into blocks) across the cluster. These blocks are then distributed to slaves, referred to as Data Nodes. Each blocks replicated across three data nodes by default, to ensure fault tolerance. Block replication and robustness of the cluster in the case of Data Node failure is an inherent characteristic of HDFS. In MapReduce, the master known as the Job Tracker, is responsible for allocating MapReduce tasks to nodes within the cluster, ideally those nodes that actually store the data. These slave nodes are referred to as Task Trackers which execute the map tasks i.e. java code in parallel. Results are then sent to nodes that run reduce tasks (another piece of java code). Reduce tasks typically aggregate the results output from the distributed mappers and write the final results to HDFS. [5] E. Apache Spark is a Framework open source of treatments of Big Data. Spark has an advanced DAG execution engine that supports cyclic data flow and in-memory computing. It run programs up to 100x faster than Hadoop MapReduce in memory, or 10x faster on disk. Spark offers over 80 high-level operators that make it easy to build parallel apps, Spark powers a stack of libraries including SQL and Data Frames, MLlib for machine learning, GraphX, and Spark Streaming. It runs on Hadoop, Mesos, standalone, or in the cloud. It can access diverse data sources including HDFS, Cassandra, HBase, and S3. You can run Spark using its standalone cluster mode, on EC2, on Hadoop YARN, or on Apache Mesos. Access data in HDFS, Cassandra, HBase, Hive, Tachyon, and any Hadoop data source [6]. F. The service-oriented architecture (SOA) principles, manufacturing automation components are extended in their structures and functionalities by a web-service interface. This extension transform the component in a ''Service, i.e., Client and Server'', able to expose and or consume, to compose, to orchestrate web services inside the architecture. The mechatronics/production layout has then been transformed into a collaborative network of operators [7]. Interoperability can be improved using the Service Oriented Architecture (SOA) inside the smart transducers network. SOA can be implemented by means of Web service technology where each service is wrapped around particular IEEE 1451.0 smart transducer function [8]. The DVFS technology is used initially to maximize the energy consumed and provide the calculation of this consumption that was applied on the two architectures presented in [9] and [6] and we arrived to conclude that the second architecture can consume more energy than the first. This simulation is done in two phases, the calculation of consumption was done on two computers on which was installed Spark and Hadoop HDFS it tries to represent an approach to almost very similar to our first service oriented architecture in accordance with the same conditions it installs Hadoop MapReduce and Hadoop HDFS to represent our second architecture and after having obtained the results which represents respectively the energy consumed in the two architectures. Hence the great difference in consumption and as we wanted to manage more Petabytes data until reaching Zettabytes and also to manage all types of data variety possible the fact that there is a need of Spark because the latter is able to manage more variety than Hadoop MapReduce but all this with the least consumption. Thus born the idea to seek and propose a third service-oriented architecture that ensures us the speed of Spark, its capacity with the least energy consumption completed by Hadoop MapReduce.

## 2. THE PROPOSED

We have proposed three implementations of the SOA, the first is an SOA where the services are in the form of MapReduce plus an HDFS [9]. This concept allows companies to facilitate the processing of Big Data so that information is divided at the beginning of processing, and the work is conducted in real time through communication between departments, and that each can individually manage Petabyte of the data. data which allows us to manage the current size of Big Data according to the SOA.

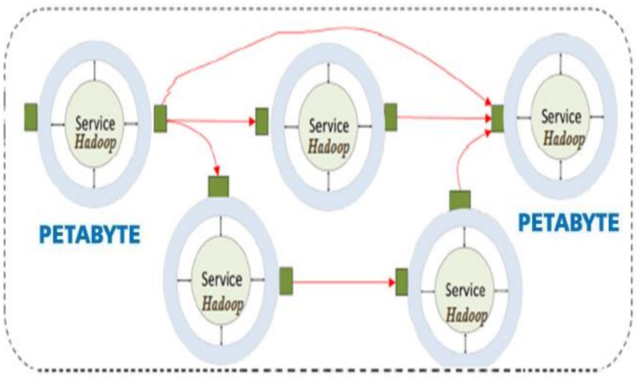


Figure 1 : Nucleus of our Service Oriented Architecture [9]

This image represents a design of the SOA kernel that is made up of services but in our design each service plays the role of a Hadoop, in this new architecture we can manage Petabyte of data several times which solves the problems related to volumes. Each service manages petabytes of data, the data is divided at the beginning of processing and collected at the end. So this design also allows us to manage big data by Hadoop regardless of the size of the data collected, so we proposed a new use of SOA to manage Big Data with the size of Zettabytes of data. Then After a simple comparison between MapReduce and Spark, we replaced the MapReduce by Spark to obtain a second new perception [6].

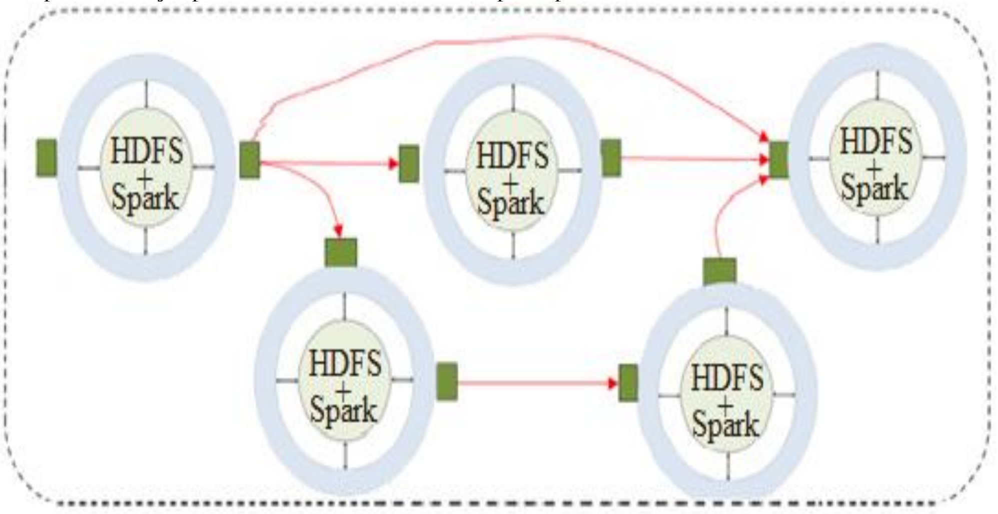


Figure 2: The core a new SOA for Spark and HDFS [6]

We suggest a new implementation of the service-oriented architecture, to bring together several Spark based on HDFS for data storage, for several other types of varieties the first architecture can not be considered for this we group together several Spark together to be able to handle Big Data regardless of its size, and its varieties. After proposing these two different scenarios for AOS, one based on Hadoop MapReduce and the other on Spark, we compared them via their power consumption using DVFS technology, we found a different new design of the previous two, namely a variety of services, one of the services is in the form of MapReduce and HDFS and the other in the form of Spark and HDFS. DVFS technology is used initially to maximize the energy consumed and offer the value of this consumption, which we applied on the two architectures presented before, and we came to conclude that the second architecture can consume more energy than the first one. . At first, we will show the need to exploit DVFS techniques to reduce Hadoop's energy consumption as well as for Spark. So, we also propose the use of DVFS to calculate the energy consumed by different SOA designs.

### 2.1. MOTIVATION

The slowness of response poses a major problem when we have to analyze large volumes of data as data warehouses from the exploitation of Big Data. We must, to reduce this slowness, choose the most rapid and effectives tools at the same time and put these tools together, in an architecture allowing: (i) managing the exchange between them and (ii) asking each tool to fulfill a contract to minimize delays in the management of Big Data and ensure the integrity and reliability of the analysis and of the exploitation of the data. In this article, we propose a new design of an implementation of the service oriented architecture in order to put Spark, Hadoop MapReduce and Hadoop HDFS all together to allow the processing and analysis of varieties of important data quickly and reliably with a view to reduce the consumption of energy during these operation. The DVFS technology shows that the energy consumption of Spark Hadoop HDFS may exceed that of Hadoop MapReduce Hadoop HDFS, as we want the architecture that has the lower consumption and at the same time there is a need of spark for a few types of varieties of data such as graphs and the batch source type or flow real-time as well as its speed in relation to the Hadoop MapReduce. We propose a new design that allows the minimization of the energy consumed this service oriented architecture have two type of services, one in the form of Hadoop MapReduce+ Hadoop HDFS, the other in the form of spark + Hadoop HDFS and as we use spark just in a few cases and also ensure the exchanges between the two type of service thanks to the characteristic of SOA which minimizes the consumption of energy used. For the delay of the work, we benefit firstly from the speed of Spark and the features offered by the SOA as the exchanges between the services and the autonomies of each service.

### 2.2. Related work

Our goal was to manage the Big Data with the most efficient Framework in this area, at the beginning, we solved the problems related to he volumes of data by regrouping several Hadoop (Hadoop MapReduce + Hadoop HDFS) in the form of services in a service oriented architecture where each is able to manage Petabytes until arriving to the Zettabytes, the current size of Big Data, during this design it can conclude that Hadoop HDFS is the most efficient to store these large volumes of data and that Hadoop MapReduce allows to process the collected data in two phases . The fact that the first comparison between Spark, which processes and analyzes data in a single phase, allows us to replace Hadoop MapReduce by Spark which represents our second result. The latest service-oriented architecture faster than the first is also able to handle more varieties and in a shorter time; But in this article we realize that the second architecture consumes more energy than the first. The purpose that we seek being to make an evolution of this architecture to make the consumption less; but still retaining all the advantages of our second and first architecture.

### 2.3. Contribution

The advanced frameworks consume considerable energy, the DVFS technique has shown that the composite architecture of Spark + Hadoop HDFS can consume more energy than that composed by Hadoop MapReduce + Hadoop HDFS among the possible reasons that Hadoop MapReduce compatible with Hadoop HDFS but the effect that Spark can access the different Datanode where Hadoop HDFS stores the data will increase energy consumption and since Spark is based on HDFS for storage so this latter offers several possibilities especially for analyzes and data processing from special sources, while keeping both: the compatibility of Hadoop MapReduce with Hadoop HDFS and the enormous capacity of spark to manage other varieties as well as its Speed.

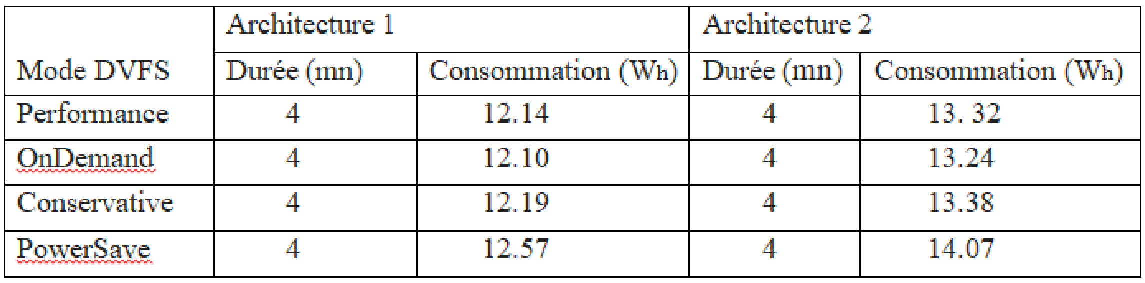
## 3. RESULTS AND ANALYSIS

The domain of Big Data is extremely large especially if want to analyze and process the data with the most effective current Framework and things become more important if we must collect several Framework together to do not exceed the limits of each one and to be able to manage all the data qualities to collect but in this new approach a comparison was made by a simulation to reduce the energy consumption during the management of Big Data, all in a service-oriented architecture.

This simulation is done in two phases, We try to represent an approach very similar to our first service oriented architecture with the same conditions we install Hadoop MapReduce and Hadoop HDFS we do the same thing to represent our second architecture and after having obtained the results, which respectively represent the energies consumed in the two architectures. To achieve this result energy consumption calculations are required in all four modes of DVFS which showed the possibility for Spark plus HDFS to exceed the consumption of MapReduce plus HDFS, and as we wanted to handle more petabytes of data. Until you reach zettabytes and also manage all the possible varieties of data the fact that you need Spark because it is able to handle more variety than Hadoop MapReduce, but all with the least consumption. the fact that Spark's power consumption with HDFS when analyzing and storing data is more than the power consumption of MapReduce with HDFS since HDFS is fully compatible with MapReduce. But for Spark to access various data sources, including HDFS, consumption is increasing.

It is ensured that the two service-oriented architectures which represent successively several Hadoop (MapReduce + HDFS) and several Spark + HDFS are able to manage Petabytes data many times, because we need to manage Zettabytes, but the energy consumption of Spark plus HDFS is more important than the energy consumption of MapReduce plus HDFS. At the same time, we want to keep the Spark favors, so we want to reduce the energy consumption of the Spark + HDFS architecture. To obtain this result, we calculated the energy consumption in the four modes of DVFS which has shown the possibility for Spark plus HDFS to exceed the consumption of MapReduce plus HDFS, this simulation is shown in the figure below (Figure.3) realized in the four modes of DFVS which are the OnDemand mode and conservative modes which have runtimes similar to the Performance mode because it does not imply slowing down of execution by increasing the frequency of the CPUs directly when necessary, and thus the OnDemand mode is the one that gives the best result of energy consumption because it is more reactive in frequency decreases than the Conservative mode, the PowerSave mode which gives the lowest consumptions at a time [10].

Where the idea of collecting Hadoop (MapReduce plus HDFS) on a service and Spark plus HDFS on another service and designing a new architecture to reduce energy consumption and manage all varieties.



| Mode DVFS | Architecture 1 | | Architecture 2 | |
|---|---|---|---|---|
| | Durée (mn) | Consommation (Wh) | Durée (mn) | Consommation (Wh) |
| Performance | 4 | 12.14 | 4 | 13. 32 |
| OnDemand | 4 | 12.10 | 4 | 13.24 |
| Conservative | 4 | 12.19 | 4 | 13.38 |
| PowerSave | 4 | 12.57 | 4 | 14.07 |

Figure 3 : Comparison of the results of energy consumption between the first architecture MapReduce plus HDFS and the second architecture Spark plus HDFS in the four modes of DVFS

After the simulation carried out using the DVFS technique, this technology, which maximizes consumption, shows that our first architecture had the capacity to consume less than the second one, but the latter gives several opportunities that should not be overlooked. We want to reduce the energy consumption with respect to our last result. To this end we combined MapReduce and Spark based on HDFS for data storage, this new architecture having two types of services, each service in our design must be composed of MapReduce + HDFS or another service composed of a Spark + HDFS, then this new implementation of the service-oriented architecture reduces the consumption of energy, money and time. We take advantage of the speed of Spark and, at the same time, that we retain the compatibilities of MapReduce with HDFS (Figure 4).

The energy consumption in the second architecture is higher than that of the first one because Spark needs to access the data contained in the HDFS, which consumes more than MapReduce when processing the data stored on the HDFS. Then, we find this design that mixes the speed of Spark and its ability to process data of different varieties like the real-time flow , all this, in order to consume the least possible energy.

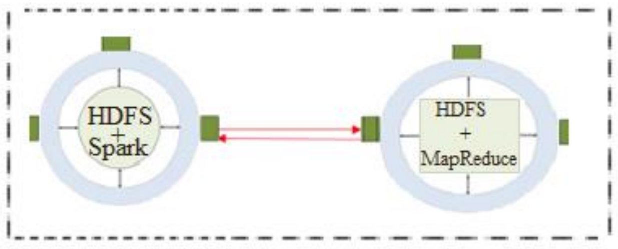


Figure 4: The core of a new SOA for Spark, MapReduce and HDFS

On the new architecture that combines Spark + HDFS and MapReduce + HDFS, the same simulation was carried out with the same DVFS technology and also in the four modes and the result shown in the figure below (Figure 5), that this new architecture consumes less than the second [3] keeping the advantages of the first two.

| Mode DVFS | Architecture 3 | |
|---|---|---|
| | Duration (mn) | Consumption (Wh) |
| Performance | 4 | 12.01 |
| OnDemand | 4 | 11.95 |
| Conservative | 4 | 11.98 |
| PowerSave | 4 | 12.41 |

Figure 3 : The energy consumption for the architecture MapReduce HDFS and Spark HDFS in the four modes of DVFS.

## 4. CONCLUSION

The development of distributed applications has, thanks to the introduction of service-oriented architecture, gained very important momentum. This architecture breaks down the functionality of the information system into several functions that are services and manages all intersections between these services. One of the implementations of this architecture was in the form of several Hadoops gathered in the environment to evolve Hadoop and make it compatible with the Big Data size of today and tomorrow. The other implementation was to replace each MapReduce by Spark and to mix the two types of service of the first two architectures, to use and to repeat each service in case of need. This third implementation regroups all the advantages of its previous ones and gives an intermediate consumption between the two consumptions more than the first one but less than the last. This study allowed carrying out a simulation by the DVFS technology. It also highlighted the large energy consumption. We want to use more the characteristics of the service-oriented architecture to reduce the energy consumption while adding a possibility to calculate this consumption from the architecture and without the need to use another simulation technology.

## ACKNOWLEDGEMENTS

I was accompanied by many people who contributed to the progress of this work. I wish to express to them my deep gratitude and gratitude, thank you so much for all of you .

## REFERENCES

[1] S. Ibrahim, T. D. Phan, A. Carpen-Amarie, H. E. Chihoub, D. Moise, and G. Antoniu, “Governing energy consumption in Hadoop through CPU frequency scaling: An analysis,” Futur. Gener. Comput. Syst., vol. 54, pp. 219–232, 2014.

[2] P. Pääkkönen and D. Pakkala, “Reference Architecture and Classification of Technologies, Products and Services for Big Data Systems,” Big Data Res., vol. 2, no. 4, pp. 166–186, 2015.

[3] H. Demirkan and D. Delen, “Leveraging the capabilities of service-oriented decision support systems: Putting analytics and big data in cloud,” Decis. Support Syst., vol. 55, no. 1, pp. 412–421, 2013.

[4] S. Gao, L. Li, W. Li, K. Janowicz, and Y. Zhang, “Constructing gazetteers from volunteered Big Geo-Data based on Hadoop,” Comput. Environ. Urban Syst., pp. 1–15, 2014.

[5] A. O’Driscoll, V. Belogrudov, J. Carroll, K. Kropp, P. Walsh, P. Ghazal, and R. D. Sleator, “HBLAST: Parallelised sequence similarity - A Hadoop MapReducable basic local alignment search tool.,” J. Biomed. Inform., vol. 54, pp. 58–64, 2015.

[6] M. Mohamed, L. Zeinebou, N. Aknin, and S. M. Lemin, “A SOA implementation to combine Spark and HDFS,” vol. 8, no. 2, pp. 1068–1071, 2017.

[7] K. Nagorny, A. Walter, and U. Schmidtmann, “Computers in Industry A service- and multi-agent-oriented manufacturing automation architecture An IEC 62264 level 2 compliant implementation,” Comput. Ind., vol. 63, no. 8, pp. 813–823, 2012.

[8] N. Bezanic and I. Popovic, “Service-oriented implementation model for smart transducers network,” Comput. Stand. Interfaces, vol. 38, pp. 78–83, 2015.

[9] M. Mohamed, L. Zeinebou, N. Aknin, and S. M. Lemin, “Implementation of the Service-Oriented Architecture to manage the Big Data with Hadoop,” vol. 7, no. 10, pp. 1615–1618, 2016.

[10] D. Costa, “Open Archive TOULOUSE Archive Ouverte ( OATAO ) Simulation énergétique de tâches distribuées avec changements dynamiques de fréquence,” vol. 2013, no. January, 2013.