Analyzing network log files using Big Data techniques

Víctor Plaza-Martín, Carlos J. Pérez-González , Marcos Colebrook, José L. Roda-García, Teno González-Dos-Santos, José C. González-González

Abstract The IT Department of the Universidad de La Laguna (ULL, Tenerife, Spain) provides service to 26 buildings with more than 1,000 network devices (wireless and wired), and access to more than 10,000 devices (computers, tablets, smartphones, etc.) which generate around 200MB/day of data that is stored mainly in the DHCP log, the Apache HTTP log, and the WiFi log files. Within this context, the chapter addresses the design and development of an application that uses Big Data techniques to analyze those log files in order to track information on the device (date, time, MAC address, and georeferenced position), as well as the number and type of network accesses for each building. In a near future, this application will help the IT Department to analyze all these logs in real time.

V. Plaza-Martín
Grado de Ingeniería Informática
Universidad de La Laguna, Tenerife, Spain
e-mail: plazamartin.victor@gmail.com

C.J. Pérez-González
Depto. de Matemáticas, Investigación Operativa y Computación
Universidad de La Laguna, Tenerife, Spain
e-mail: cpgonzal@ull.edu.es

M. Colebrook (✉)
Departamento de Ingeniería Informática y de Sistemas
Universidad de La Laguna, Tenerife, Spain
e-mail: mcolesan@ull.edu.es

J.L. Roda-García
Departamento de Ingeniería Informática y de Sistemas
Universidad de La Laguna, Tenerife, Spain
e-mail: jlroda@ull.edu.es

T. González-Dos-Santos
Grado de Ingeniería Informática
Universidad de La Laguna, Tenerife, Spain
e-mail: tenoglez@gmail.com

J.C. González-González
Servicio de Telecomunicaciones y Tecnologías de la Información
Universidad de La Laguna, Tenerife, Spain
e-mail: jgonzal@ull.edu.es

1 Introduction

By the time you read this chapter, over 16 TB of data have been generated in the world every second, as shown in [1]. This represents that in 2016, global IP traffic will reach 1.1 zettabytes (ZB) per year, or 88.4 exabytes (EB, nearly one billion gigabytes) per month, or nearly 3 EB per day, which is certainly a huge amount of data.

According to [2], in 2013 four zettabytes (1 ZB = 10^9 TB = 10^{21} bytes) of data were created by digital devices. In 2017, it is expected that the number of connected devices will reach three times the number of people on earth.

Hence, the main topic of the book, namely Big Data, is justified based on the current technological situation in which data is generated at a higher speed than can actually be processed, and large companies are facing the problem of deleting data due to the impossibility to store it, thus losing useful information.

In fact, one of the main sources of data are the log files, which record either events that occur in an operating system or other software runs, or messages between different users of a communication software.

Thus, an opportunity arose to work collaboratively with the Information Technology and Communication Department (STIC in Spanish) at the Universidad de La Laguna. The STIC provides service to 26 buildings with more than 1,000 network devices (wireless and wired), and renders access to more than 10,000 devices (computers, tablets, smartphones, etc.), which generate around 200 MB/day of data that is stored mainly in the DHCP (Dynamic Host Configuration Protocol) log, the Apache HTTP log, and the WiFi log files.

The key problem was the need to monitor the access made from a single device, or the user's activities all around the campus. In this sense, they wanted to infer usage patterns throughout the day in order to strengthen the WiFi network at certain points.

Within this context, the chapter addresses the design and development of an application that uses Big Data techniques to analyze those log files in order to track information on the device (date, time, MAC address, and georeferenced position), as well as the number and type of network accesses for each building. In a near future, we believe that this application will help the STIC to analyze all these logs in real time.

Although there are several applications focused on log management in the market (see Table 5.1), the STIC wanted a custom tailored application since their log files were not in the standard format, as we explain in the next sections. For a detailed comparison of open-source log management solutions, the reader is referred to [3].

Table 5.1 List of log management and analysis tools.

Tool	**Description**	**Type of license**
ELK Stack	Set of applications and utilities (Logstash, Elasticsearch, Kibana) to create a powerful search and analytics platform	Free
Graylog2	Log management system with server and a web interface	Free
Logentries	Real-time log management and analytics	Free/Commercial
Loggly	Cloud-based log management service	Free/Commercial
Logscape	Allows searching, visualizing and analyzing log files from a central dashboard	Free/Commercial
Splunk	Industry-leading platform that automatically indexes log data	Commercial
Logtrust	Turns machine data into business insights	Commercial

The remainder of the chapter is organized as follows. In Sect. 2, we provide the definition of Big Data, its influence and relevance nowadays, as well as a description of the Hadoop framework. Sect. 3 presents the problem description using TOGAF. The project development and the working methodology are described in Sect. 4. In Sect. 5, we present and discuss the results for each developed task, along with some charts depicting the Hadoop cluster indicators. Finally, the conclusions are provided in Sect. 6.

2 Big Data state-of-the-art

As we stated in the introduction, every day nearly 3 EB (1 EB = 10^6 TB) of data is generated [1]. IBM [4] pointed out that the main sources for this entire data arise from the following:

- Web data and social media
  - Web content
  - Twitter feeds
  - Facebook postings
  - Clickstreams (data from user navigation)
- Data generated from M2M (machine-to-machine) communication
  - GPS signals
  - RFID readings
  - Intelligent readers
- Great data transactions
  - Government
  - Business

- Biometrics
  - Facial recognition
  - Genetics
- Data generated directly from human beings
  - Emails
  - Voice recordings
  - Records of all kinds

Another interesting approach is given by [5], who associated the data types (structured or unstructured) to the data source (internal or external), asserting that the unstructured external data is the largest area of opportunity for the enterprise (see Fig 5.1).

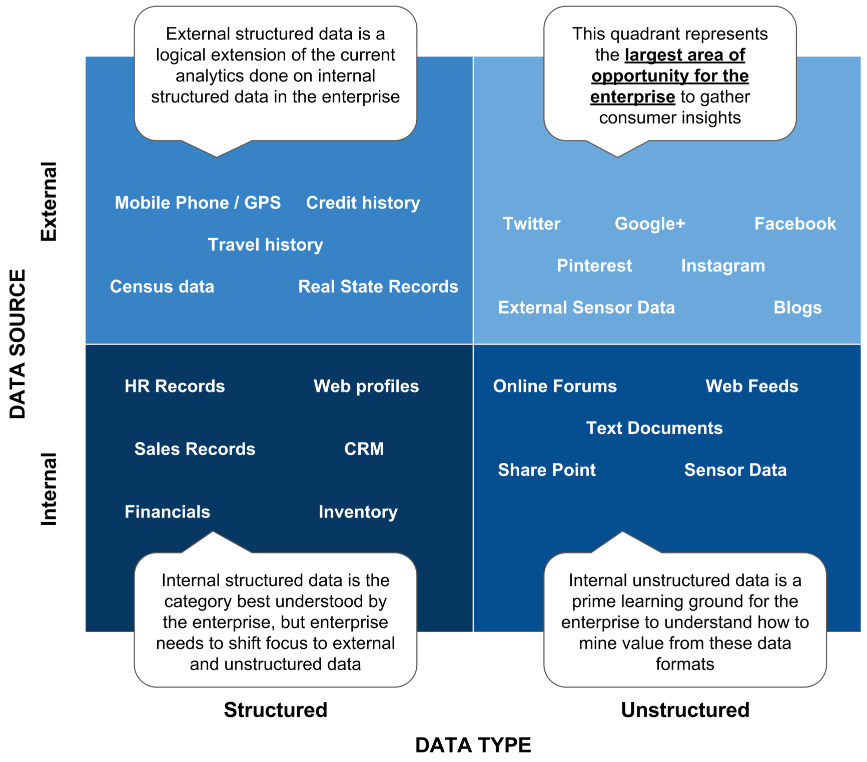
External structured data is a logical extension of the current analytics done on internal structured data in the enterprise
This quadrant represents the **largest area of opportunity for the enterprise** to gather consumer insights
Mobile Phone / GPS
Credit history
Travel history
Census data
Real State Records
Twitter
Google+
Facebook
Pinterest
Instagram
External Sensor Data
Blogs
HR Records
Web profiles
Sales Records
CRM
Financials
Inventory
Online Forums
Web Feeds
Text Documents
Share Point
Sensor Data
Internal structured data is the category best understood by the enterprise, but enterprise needs to shift focus to external and unstructured data
Internal unstructured data is a prime learning ground for the enterprise to understand how to mine value from these data formats
External
Internal
DATA SOURCE
Structured
Unstructured
DATA TYPE

Fig. 5.1 Data Types vs. Data Sources (adapted from [5])

Bloem et al. [6] summarized these sources by relating the increasing data variety and complexity with its volume (in bytes), from the ERP systems to the current Big Data scenario, going through the CRM and Web systems (see Fig. 5.2). As a consequence, Big Data is defined as a variable equal to:

$$Transactions + Interactions + Observations$$

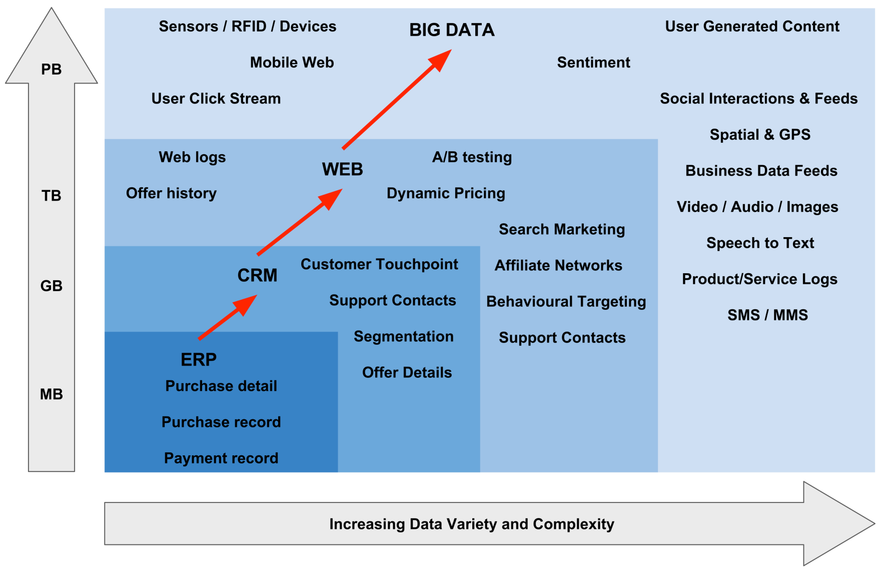

Fig. 5.2 From ERP to Big Data (adapted from [6])

Nevertheless, the definition of Big Data that is the most widely accepted was stated by Doug Laney, industry analyst from Gartner, in 2001 when he established the three V's of the Big Data [7]: Volume, Velocity and Variety.

Volume

The volume refers to the amount (size) of the datasets generated and the difficulty that companies and organizations have to process them. The forecasts are that in 2020, 25 billion devices will be connected to the Internet, making data grow exponentially, multiplying by 10 the amount in a period of just 5 years.

Velocity

The speed is often misunderstood as a real-time analysis, but it also refers to the rate of change (or flow) in the data, linking data coming at different speeds, and also to the activity peaks.

Variety

This is perhaps the most interesting characteristic to analyze, since data can arise from all areas of daily life, including multiple repositories, domains or types. However, most of this data already belongs to organizations, but they make no use of it, becoming what Doug Laney called *dark data*. This data, despite not being useful by itself, can be analyzed and processed to generate useful and valuable information, and hence, opening new business opportunities. Depending on the type of data, we can subdivide it into four main groups:

- Structured

- Semi-structured
- Unstructured
- Complex structured

Besides, the terms **variability** and **complexity** also arise as possible features concerning Big Data, since data might be generated inconsistently, with huge peaks from multiple sources that makes very difficult to relate them.

The following Fig. 5.2 adapted from [8] graphically summarizes the three characteristic V's of Big Data.

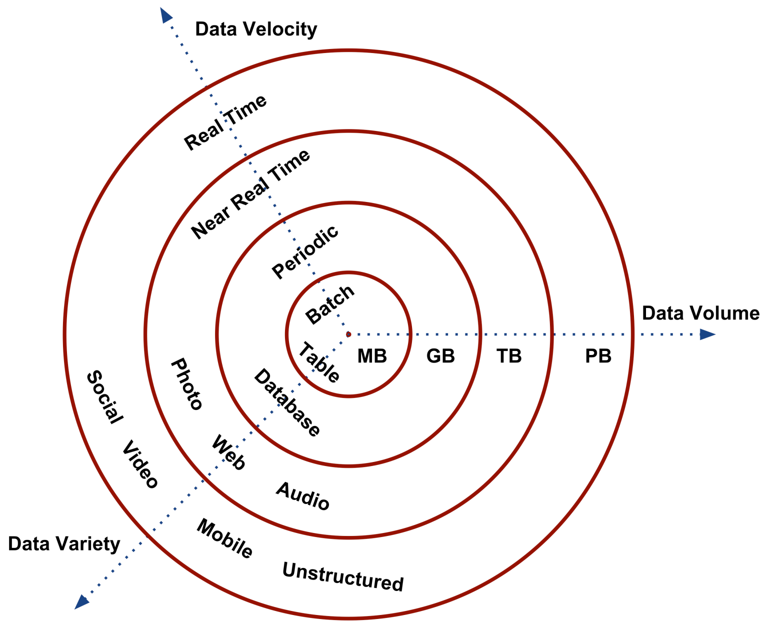

Fig. 5.3 The three V's that define Big Data (adapted from Soubra, 2012)

Yiu [9] also contributed with another definition of Big Data: *"datasets that are too awkward to work with using traditional, hands-on database management tools*. Besides, he also defined the term Big Data Analytics as *the process of examining and interrogating big data assets to derive insights of value for decision making*."

For the TechAmerica Foundation [10], Big Data is a term that describes "*large volumes of high velocity, complex and variable data that require advanced techniques and technologies to enable the capture, storage, distribution, management, and analysis of the information*."

One of the latest definitions comes from the National Institute of Standards and Technology [11], who considers Big Data as "*consisting of extensive datasets –*

primarily in the characteristics of volume, variety, velocity, and/or variability – that require a scalable architecture for efficient storage, manipulation, and analysis."

The main idea behind all these definitions of Big Data is to analyze the data sets to gain insight and valuable information for the final decision making. In this sense, Davenport and Harris [12] and SAP [13] suggest that businesses should change as soon as possible from a "Sense & Respond" to a "Predict & Act" strategy, in which Big Data plays an important role in all the steps towards the main goal (see Fig. 5.4).

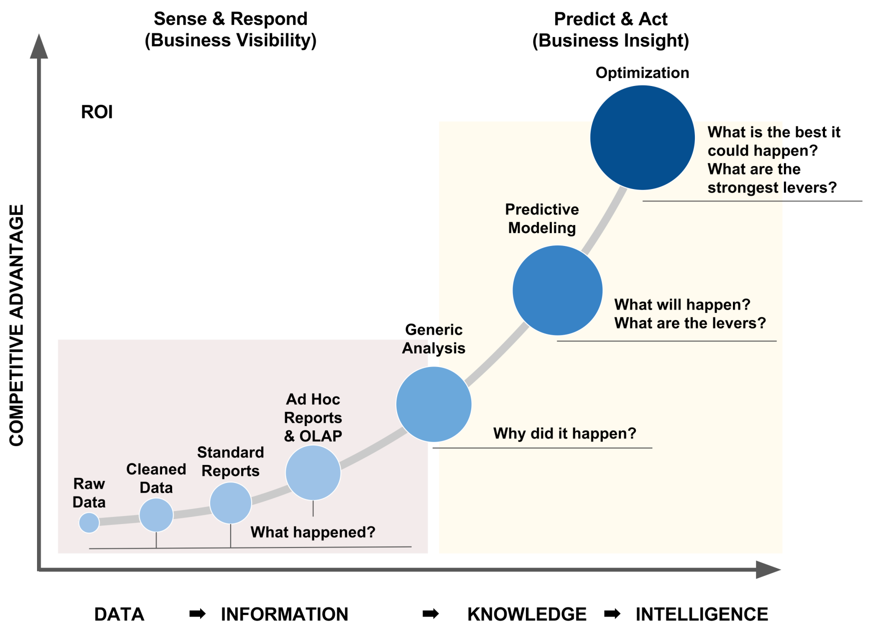

Fig. 5.4 From "Sense & Respond" to "Predict & Act" (adapted from SAP, 2011)

Finally, and before presenting the Hadoop framework in the next section, just to remark that the Telecommunications Standardization Sector of the International Telecommunication Union (ITU-T) has recently released "Recommendation Y.3600", which provides requirements, capabilities and use cases of cloud computing based Big Data as well as its system context [14]. This is the first attempt to make an international standard on Big Data, and we believe that it will truly provide solid foundations for the future design and development of applications on this subject.

2.1 The Hadoop framework

For some time ago, the term Big Data has been directly involved with the use of the Hadoop framework because it allowed analyzing unstructured data easily and quickly. The Hadoop MapReduce framework [15] is an free software solution that supports distributed applications, and is currently the most widely used solution for leading companies such as Yahoo, Ebay, Facebook, IBM, etc., since it allows working with thousands of nodes and petabytes of data.

Hadoop is usually referred as an ecosystem, due to the huge number of projects, services, libraries, APIs, etc., that gathers around. However, its architecture is mainly composed of these elements:

- Hadoop Common: the set of Java libraries and tools required by the different modules to access the filesystem and to start the cluster daemons.
- HDFS (Hadoop Distributed File System): basically a distributed file system, scalable and portable, characterized by its high throughput access to application data and the suitability for applications that have large data sets. An HDFS cluster consists of:
  - A single NameNode, that is a master server that manages the file system namespace and controls the file access by the clients.
  - In addition, there are a number of DataNodes, usually one per physical node in the cluster, which manages the storage attached to them.
- Hadoop YARN: the framework to split up the functionalities of resource management and job scheduling/monitoring into separate daemons. YARN frees the MapReduce engine from cluster resource management tasks, streamlining data processing and the execution of the tasks. This data-computation framework consists of:
  - The ResourceManager: ultimate authority that arbitrates resources among all the applications in the system.
  - The NodeManager: framework agent who is responsible for containers, monitoring their resource usage (CPU, memory, disk, network) and reporting to the ResourceManager/Scheduler.
- MapReduce: a JobTracker where client applications send jobs.
- A set of parallel applications that enhance the Hadoop ecosystem, such as Pig, HBase, Hive, Hue, etc.

To develop this project we used Hadoop 2.0, which is a great update to the architecture of Hadoop 1.0, as shown in Fig 5.5.

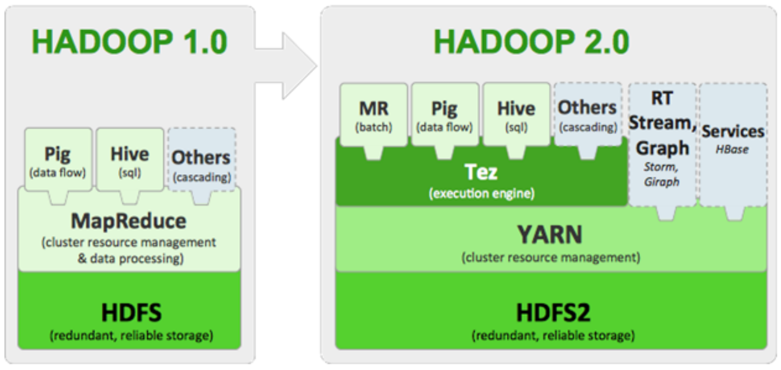
HADOOP 1.0
Pig (data flow)
Hive (sql)
Others (cascading)
MapReduce (cluster resource management & data processing)
HDFS (redundant, reliable storage)
HADOOP 2.0
MR (batch)
Pig (data flow)
Hive (sql)
Others (cascading)
RT Stream, Graph Storm, Giraph
Services HBase
Tez (execution engine)
YARN (cluster resource management)
HDFS2 (redundant, reliable storage)

Fig. 5.5 Architectural update in Hadoop 2.0

It is noteworthy to remark that the HDFS system is specially designed to handle large files. Besides, it is responsible for splitting and distributing the contents of large data files into blocks or chunks of 64 MB (by default), generally across multiple machines (nodes). Its reliability is based on the replication of data blocks across three different hosts (by default). Hence, the hosts can communicate among them and rebalance the flow of data, move data and maintain a high replication within the cluster to ensure the proper performance of the application.

The inner working scheme in Hadoop MapReduce is a sequential process in which the JobTracker partitions large data sets stored in the HDFS system into smaller blocks of 64 MB in the NameNode (master node). Then, these blocks are distributed to the DataNodes (slave nodes) to perform the reading and mapping to generate the output as a key-value tuple that is stored in a temporary destination. At this stage, the Shuffler sorts those tuples by the key and subsequently all elements with the same key in the same node remain together. In the last step, the DataNodes gather all the tuples with other partial results, to generate the final output file. Even though the MapReduce process is completed, one last step is usually added to visualize the data, such as using the statistical software R and the Shiny framework.

For a deeper knowledge using the Hadoop framework for addressing big data challenges, the reader is referred to Hu et al. [16]. Once we have introduced the working framework, we are now ready to describe the problem in the next section

3 Problem description

As we stated in the introduction, the STIC is in charge of the management and maintenance of the university's network, which comprises more than 100 telematical services accessible from 26 buildings, along with more than 1,000 network devices (wired and WiFi).

Previous to this project, the STIC had an application developed with Pentaho Business Intelligence Community [17]. The main goal of this tool was to combine the information obtained from various sources, mainly the access log from the Apache server, the DHCP log file, and the log files generated by the WiFi device drivers, in order to determine the total number and type of accesses for each building.

Although the application performed relatively well without the chance to apply any filter, the running time spent to get the results for the network's least used time slot for a single day was nearly three days of execution. Thus, we settle the possibility of carrying out a new process using Big Data tools to significantly reduce the running times.

Therefore, and before developing the new solution, we first modeled the log system using architectural framework tools.

3.1 Modeling the WiFi log system

Current modern ICT (Information and Communications Technology)-intensive institutions need to use business-modeling techniques that relate the business goals with the technology evolution. In the last twenty years, new ways to interconnect company internal systems using new enterprise architecture frameworks and technologies have appeared.

Regarding this matter, The Open Group Architecture Framework (TOGAF) is one of the most important enterprise architecture frameworks developed since the 1990s. In its latest version 9.1, TOGAF [18] provides a method and a set of good practices that allow a good relationship between the overall systems of an enterprise and its stakeholders.

Businesses evolve rapidly to consider new stakeholders and technical requirements. Institution goals and emerging technologies must be aligned accordingly, but daily operational tasks make this issue a hard job to deal with, whereas daily changes make business management a complex activity. The continuous transformation from the current system to the required one is maintained under the TOGAF methods and good practices. Two of the most important concepts in TOGAF are the views and the viewpoints, respectively, which allow the enterprise architect to focus in different parts of the complete business. As stated in TOGAF documentation (TOGAF 2011), we can use the framework to model all the business or we can concentrate in different parts of it.

In this chapter, we model the WiFi log information system of the ICT Department (STIC) of the Universidad de La Laguna. We have used two different views to define the main goals: the layered structure of the system and the application solution view.

For this task, we have used Archimate (TOGAF 2011), which is an enterprise modeling language that captures the complexity of different domains and its relations within the organization. Archimate is a modeling tool that will increase the representation and comprehension of the enterprise modeled systems. Hence, the enterprise architect models different parts of the institution focusing on the elements he/she wants to emphasize considering many other aspects of the system. Besides, by using Archimate views the architect is able to analyze a specific problem bearing in mind the complete system.

In Archimate, the service concept plays a central role and can be defined as a unit of functionality that an entity offers to other entities or environment. One of the most important viewpoint diagrams is the Layered viewpoint [19]. The main goal of the Layered viewpoint is to provide an overview in one diagram. This view consists of three layers: Business layer, Application layer and the Technology layer.

The Business layer offers products and services to external customers that run business processes performed by business actors. The Application layer gives support to the business layer with application services that are performed by applications. And finally, the Technology layer offers infrastructure services accomplished by computers and communication hardware and software. Relationships between layers are formed by the use relation, which shows how the higher levels use the services of the lower layers. A second type of link is the realization relation that shows which element in the lower layer realizes comparable elements in higher layers.

3.2 Solution achieved

We analyzed the WiFi log system and, although there exist many relationships between different business domains (login, academic, human resources, etc.) we have centered our efforts in the main goals that STIC's staff defined:

- Track all network communication elements (routers, switches, servers, etc.)
- Fast support to network failures
- Track specific MAC or IP elements in the network
- Detect communication bottlenecks
- Geolocated network status visualization
- Provide better network services

3.2.1 Layered viewpoint

Through the Layered viewpoint diagram, we propose a viewpoint based on layers or views, namely a BAI scheme (Business - Application - Infrastructure). Thus, we can make an approach from different perspectives or viewpoints based on our interests and subsequently, obtaining both a global representation of the solution developed, as well as the current application by only changing the view.

Starting with the infrastructure layer, we can see in Fig. 5.6 that consists of two basic elements. The first one is a set of computers located in the Computer Science School connected via LAN, which provide hardware support in our Big Data solution. On this infrastructure we have installed Hadoop 2.6.0 for Linux [15], running both as NameNode and DataNode. One of the computer plays the role of master of the cluster while the rest of nodes have the DataNode settings and play the role of slaves.

On the other hand we have a Windows based computer holding an Apache HTTP server, along with R [20] as the statistical analysis software.

Hadoop provides a number of services such as Streaming, HDFS, YARN and JobTracker, which will all be used by the application component MapReduce, whereas R provides RStudio [21] and ggplot services to Shiny [22].

As stated above, at the application level we have two basic components. The first one is the MapReduce component, which comprises the Mapper, Shuffler and Reducer services running sequentially in a pipeline mode. This application component generates a CSV output file that serves as input to the other application component (Shiny), and hence, establishing a collaboration between these two components.

At the business level, we can see that the user (STIC's staff) is presented with three basic actions: the monitoring of the tasks and the cluster (both through the user interface provided by YARN), and viewing the results and managing the queries through a small dashboard.

3.2.2 Application Behavior Viewpoint

If we now change our view and perform an analysis at the application level focusing on the MapReduce component (see Fig. 5.7), we can see that is composed by two other MapReduce components: one for analyzing the DHCP logs, whereas the other one integrates this information with the analysis of the Apache logs.

The MapReduce process responsible for the analysis of the DHCP logs starts cleaning the input data using a regular expression. Thus, we obtain the set of useful data and, in turn, we filter those log lines that are not needed for the analysis. All this is accomplished by using generators and iterators in Python, so that a certain element is not generated until you actually need it. These programming techniques imply less computational cost and memory usage. Once we have filtered this information, we proceed to generate the data structure with that task, in this case a dictionary within a dictionary, so that the IP is the key of the generic dictionary and its value a dictionary where the key is the time and the MAC is the value. This allows us to clearly identify the use of an IP address at any time and what device is making use of it.

The Shuffler sorts all the information generated on each of the cluster nodes within the mapping process, so as to ensure that all entries with the same key are computed in the same DataNode, and therefore the integrity of the solution is assured.

Now the Reduce process begins to synthesize all the DHCP information to a minimal set, storing a record of each time an IP assigned to MAC changed, instead of the whole entry. Consequently, the output volume is reduced by 58%, even though it contains the same useful information. This aspect is key since the MapReduce process responsible for processing the Apache Log files will use this file later.

The second MapReduce process is similar to the first one in its concept, because it is based on cleaning the input data through regular expressions to get the smallest unit of information required. Later on, we check the dictionary generated as a result of the above process for the DHCP analysis, and we add both data if needed.

Thus, we relate each useful access to its corresponding MAC address, adding georeferencing information generated by the STIC's staff based on the architecture of their WiFi and wired network.

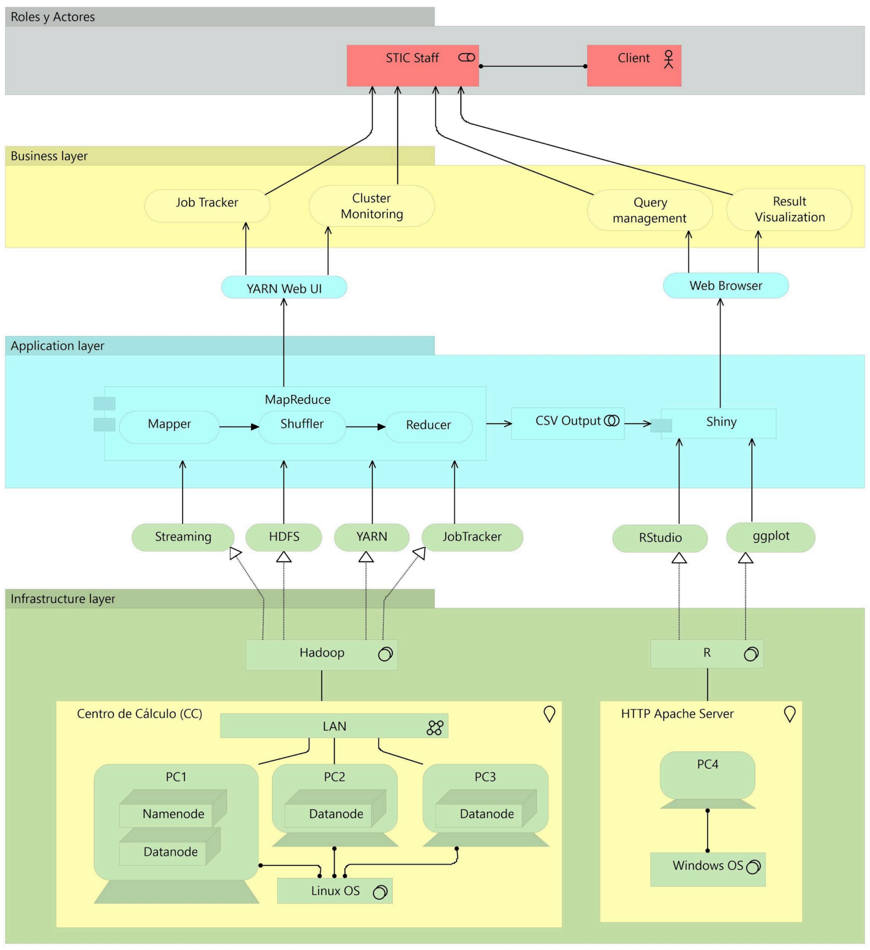

Fig. 5.6 The Layered Viewpoint

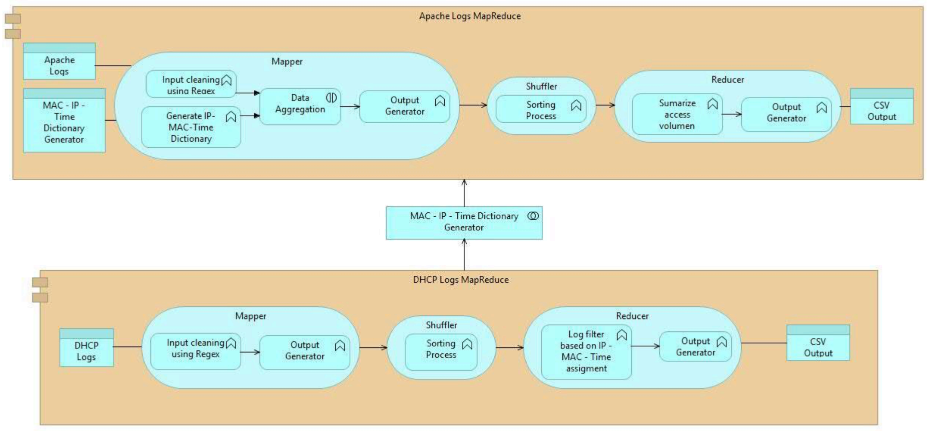

Fig. 5.7 The Application View

4 Project development

After several meetings with the STIC's staff, some restrictions were set, which shaped the structure of this project, emphasizing among them the use of Python as the programming language in order to ease the maintenance of the solution proposed.

Bearing always in mind that performance was the top priority, different alternatives were analyzed and compared in order to ensure the best results were obtained in such diverse aspects as reading/writing speed from both files and the HDFS system, processing time, functional requirements, etc.

Hence, and using as a baseline the comparative generated by Laserson [23], we concluded that the best option was to develop the solution through the Streaming library that has performance ratios and a range of features similar to Java.

4.1 Working methodology

One of the biggest challenges when working with large amounts of data is to verify the accuracy of the results obtained, and this was a critical aspect for the STIC's staff.

Accordingly, a working methodology was designed in which an incremental development of the solution would allow controlling the results from a large enough input dataset.

Therefore, the development process was split into milestones (or tasks), each of them with a defined, well-documented objective, and a series of data files as inputs for analysis. Thus, it was concluded that the working planning would be as follows (the ‘V’ before the number stands for the version):

V0.5 Analysis of a log file folder

The system with a single node will compute the data in a directory, traverse all log files inside, and produce a CSV file as the output with the names of the files found and the number of log lines detected in each one.

V1.0 Analysis of a single log file with a single node

For one single log file of a single virtual server from the campus, get the number of accesses grouped by hour.

V1.5 Analysis of a single log file with three nodes

For one single log file of a single server from the farm of virtual servers, get the number of accesses grouped by hour, splitting the computation into three nodes.

V2.0 Analysis of several log files one-day analysis on multiple nodes

For the set of log files in a single day of multiple servers in the campus, obtain the number of accesses grouped by hour, for each server and for each time slot.

V2.5 Analysis of several log files one-day analysis on multiple nodes with date and time restrictions

For the set of log files from a single day of multiple servers in the campus, obtain the number of accesses grouped by hour, for each server and for each time slot, and filter the restrictions from-day/from-time and to-day/to-time specified in a special file.

V3.0 DHCP log analysis

We must determine the MAC devices that connect to the network at any time. The system will read the log files from the DHCP servers and generate a new output file with the MAC addresses associated to each IP address at each time instant.

V4.0 Linking unstructured data and correlating elements in different time instants

For a certain access log line to the virtual campus server, we have to determine the MAC address associated with the IP address that made that precise access. The output may now include the tuple: Date, Time, IP, MAC and Access Object.

V4.5 Deploying V4.0 for a period specified in the input filters

This task must link all the above sections and get the tuple of V4.0 for all log files for a given period of time in the input filter.

V5.0 Analyzing WiFi connections

Analyze the log files of the different WiFi device controllers to obtain the tuples: Date, Time, MAC, WiFi Access Point.

V5.5 Analyzing WiFi connections and georeferenced access

Analyze the georeferenced files for the WiFi access points, and link to the previous section the coordinates of the detected accesses.

V6.0 Linking all together

Link all previous sections to obtain the following tuple for a specific time period defined in the input filter: Date, Time, Virtual Campus Access, IP, MAC, Access Point, GPS coordinates.

5 Results

The main data sources are the semi-structured access log files that are automatically generated each time a user's device access the university's network. Currently, the amount of data generated daily as a result of network monitoring exceeds 200 MB per day.

This is mainly due to the extensive network, which provides access to approximately 10,000 devices and renders services to more than 26 buildings, with a total of more than 1,000 network devices (wired and WiFi).

Besides, the access log files lack sensitive information and hence they need not be anonymized. Indeed, the files have the characteristic morphology of being a concatenation of an unsorted standard Apache log along with data added by the log management system of the STIC. The fields are separated by spaces, and the symbol "-" denotes the absence of data. An example of a log line in this type of files is shown in Fig 5.8.

```
Mar 16 12:04:32 udvweb1.stic.ull.es apache2_access: 85.59.43.201 - -
[16/Mar/2014:12:04:32 +0000] "GET /1314/theme/image.
php/institucional/core/1387354149/t/hide HTTP/1.1" 200 1496 "http:
//campusvirtual.ull.es/1314/course/view.php?id=637" "Mozilla/5.0 (Windows
NT 6.1; WOW64; rv:27.0) Gecko/20100101 Firefox/27.0"
```

Fig. 5.8 Sample line in an access log file

Before presenting the results obtained over a given set of server logs from 2014, first we must process these log files by running a Hadoop MapReduce program stored in the HDFS system within a cluster of nodes.

5.1 Cluster configuration

A cluster of 21 data nodes (and 1 node manager) with 4 GB of RAM in each node was deployed with Hadoop YARN version 2.7. When submitting MapReduce procedures in Hadoop, we are interested in measuring the cluster performance and benchmarking the job executions. In this sense, Hadoop provides a high valuable Java API for debugging problems in MapReduce jobs, analyzing the use of memory and CPU time, and tracking the filesystem operations.

Among the most important resources within this Hadoop API are the counters, which are a wide selection of indicators and statistics reported by the individual tasks associated to job submissions. The counters in Hadoop are used to track the job progress of a MapReduce algorithm, and to study the events occurred during the job execution. White [24] describes five categories or groups of counters in Hadoop (see Table 5.2) :

Table 5.2 Counter groups in Hadoop

Group	Java enum
Map Reduce task counters	org.apache.hadoop.mapreduce.TaskCounter
FileSystem counters	org.apache.hadoop.mapreduce.FileSystemCounter
Job counters	org.apache.hadoop.mapreduce.JobCounter
FileInputFormat counters	org.apache.hadoop.mapreduce.lib.input.FileInputFormatCounter
FileOutputFormat counters	org.apache.hadoop.mapreduce.lib.output.FileOutputFormatCounter

The group of task counters is updated as a task progresses, whereas the group of job counters are updated as a job progresses. Some of the built-in MapReduce counters are detailed in the following Table 5.3:

Table 5.3 MapReduce task counters

Counter	Description
MAP_INPUT_RECORDS	The number of input records consumed by all the maps in the job.
MAP_OUTPUT_RECORDS	The number of map output records produced by all the maps in the job.
MAP_OUTPUT_BYTES	The number of bytes of uncompressed output produced by all the maps in the job.
PHYSICAL_MEMORY_BYTES	The physical memory being used by a task in bytes, as reported by /proc/meminfo.
CPU_MILLISECONDS	The cumulative CPU time for a task in milliseconds, as reported by /proc/cpuinfo.
GC_TIME_MILLIS	The elapsed time for garbage collection in tasks in milliseconds (reported by GarbageCollectorMXBean.getCollectionTime())

Likewise, several built-in filesystem counters can be outlined as well:

Table 5.4 Filesystem counters

Counter	Description
BYTES_READ	The number of bytes read by the filesystem by map and reduce tasks. There is a counter for each filesystem (local filesystem, HDFS, etc.)
BYTES_WRITTEN	The number of bytes written by the filesystem by map and reduce tasks.
READ_OPS	The number of read operations (i.e. open, file status) by the filesystem by map and reduce tasks
WRITE_OPS	The number of write operations (i.e. create, append) by the filesystem by map and reduce tasks.

The job counters measure statistics at the job level, and their values are not changed while a task is running. Some significant counters in this group are shown in the following Table 5.5:

Table 5.5 Job counters

Counter	Description
TOTAL_LAUNCHED_MAPS	The number of map tasks that were launched.
TOTAL_LAUNCHED_REDUCES	The number of reduce tasks that were launched.
NUM_FAILED_MAPS	The number of map tasks that failed.
NUM_FAILED_REDUCES	The number of reduce tasks that failed.
NUM_KILLED_MAPS	The number of map tasks that were killed.
NUM_KILLED_REDUCES	The number of reduce tasks that were killed.
MILLIS_MAPS	The total time taken running map tasks, in milliseconds.
MILLIS_REDUCES	The total time taken running reduce tasks, in milliseconds.
MB_MILLIS_MAPS	The total time taken running map tasks multiplied by RAM allocated, in miliseconds*megabyte
MB_MILLIS_REDUCES	The total time taken running reduce tasks multiplied by RAM allocated, in miliseconds*megabyte

5.2 Sample results for each task

Next, we show a sample of the MapReduce output for each stage of the project on the input data, namely, the access log files, the DHCP log files, and a set of filter files or static information such as location coordinates of the IP addresses within the campus.

V0.5 Analysis of a log file folder

Given a set of access log files, we obtain the following Table 5.6:

Table 5.6 Output of task V0.5

Filename	Number of Lines
access.log-20140317	318599
access.log-20140318	527553
access.log-20140319	513896
access.log-20140320	505612
Total number of lines	1865660

V1.0 Analysis of a single log file with a single node

The fragment of the generated output is shown in Table 5.7.

Table 5.7 Output of task V1.0

Month	Day	Hour	IP	Number of Accesses
Mar	17	10	10.212.2.230	843
Mar	17	10	10.212.2.94	68
Mar	17	10	10.212.3.189	59
Mar	17	10	10.212.3.212	1357
Mar	17	10	10.212.3.233	277

V1.5 Analysis of a single log file with three nodes

The output generated contains the same format as seen in Table 5.7, but the input now comes from three files, one for each node, in order to verify that the aggregation between node output files works properly.

V2.0 Analysis of several log files one-day analysis on multiple nodes

The output shown in Table 5.8 counts the number of accesses per IP/date/server, and then computes a total result by date/IP.

Table 5.8 Output of task V2.0

Month	Day	Hour	Ip	udvweb1 log-20140318	udvweb2 log-20140318	udvweb3 log-20140318	udvweb4 log-20140318	Total
Mar	17	8	10.219.3.69	0	0	19	0	19
Mar	17	8	10.219.8.238	6	914	0	0	920
Mar	17	8	10.219.8.239	0	1522	0	10	1532
Mar	17	8	10.219.8.242	176	1	0	0	177
Mar	17	8	10.219.8.243	0	0	0	32	32

V2.5 Analysis of several log files one-day analysis on multiple nodes with date and time restrictions

The output of this task is identical to Table 5.8, but in this case we use a custom filter file with timeslots to restrict the time of the accesses. This filter file has the following structure:

```
From Mar 17 10:00:00

Until Mar 17 17:00:00
```

V3.0 DHCP log analysis

In this case, we filter according to file containing IP ranges associated with different logical origins within the network management. This filter file has the following contents:

```
192.168.x.x Danger1

172.16.x.x Danger2

10.x.x.x Internal

193.145.96.x PublicInternal

193.145.97.x PublicInternal

193.145.120.x Administration

193.145.125.x PublicInternal

*.ull.es InternalWithDNS

Others External
```

Table 5.9 shows a sample of the output generated in this task.

Table 5.9 Output of task V3.0

Month	Day	Hour	Admin.	External	Internal	Internal With DNS	Public Internal	Danger1	Danger2	Total
Mar	17	6	0	4686	103	128	0	0	0	4917
Mar	17	7	0	20003	3495	384	0	0	0	23882
Mar	17	8	0	44578	64334	384	0	0	0	109296
Mar	17	9	0	67733	91041	384	0	0	0	159158

V4.0 Linking unstructured data and correlating elements in different time instants

The goal is to analyze the DHCP log file in order to have a baseline to gather all the data previously analyzed from the V5.5 task on (see Table 5.10).

Table 5.10 Output of task V4.0

Month	Day	Hour	Server	Access Type	IP address	MAC address
Mar	18	11:34:32	udvweb1.stic.ull.es	apache2_access	10.225.2.207	00:17:31:c4:30:71

V4.5 Deploying V4.0 for a period specified in the input filters

This output is similar to task V2.5 but over another data set, whereas the structure of the output is the one obtained in task V4.0 but considering some time filters.

V5.0 Analyzing WiFi connections

The output is similar to task V4.0 but for WiFi connections (see Table 5.11).

Table 5.11 Output of task V5.0

Date	Hour	MAC	Access Point
01/03/2016	8	88:9f:fa:93:35:7c	10.174.3.28

V5.5 Analyzing WiFi connections and georeferenced access

Given a set of files containing information related to the logical design of the WiFi network, we generate an output with the format shown in Table 5.12. This output format will help to link all previously developed tasks with the goal described in task V6.0.

Table 5.12 Output of task V5.5

Building Code	Building Name	Network Type	Netmask	Building GPS Coordinates
BA2	Bellas Artes Nuevo	Red Azul	10.101.0.0/24	28.461210,-16.276267

V6.0 Linking all together

The output of this task (shown in Table 5.13) was from the beginning the main goal of the project, that is, to aggregate access data with DHCP information and georeferenced positions in order to track a particular device, and to analyze both the network traffic flow and the network load.

Table 5.13 Output of task V6.0

Date	Access Point	Device MAC	Building	Building GPS Coordinates
01/03/2016 8:38	10.174.3.28	88:9f:fa:93:35:7c	PE Periodismo Red WIFI	28.469314,-16.301921
01/03/2016 8:56	10.158.0.207	20:64:32:4c:44:94	IA IUBO+Agricolas Red WIFI	28.481374,-16.319417

5.3 Dashboards using R charts and data from counters

The tasks described in Sect. 4 were executed on four instance problems to analyze the server logs of several months in 2014. In particular, the following months periods were processed: one month (August), two months (August to September), three months (August to October) and, finally, four months (August to November).

The job counters of the tasks execution were collected and the information was gathered into several JSON files, in order to properly show it in a dashboard developed using R [20]. This dashboard will allow us to study the efficiency of our MapReduce scripts. For example, the next charts shown in Fig. 5.9 and Fig. 5.10 represent the number of tasks and the memory usage by data node (simply averaging PHYSICAL_MEMORY_BYTES by the number of data nodes) for each problem instance, respectively.

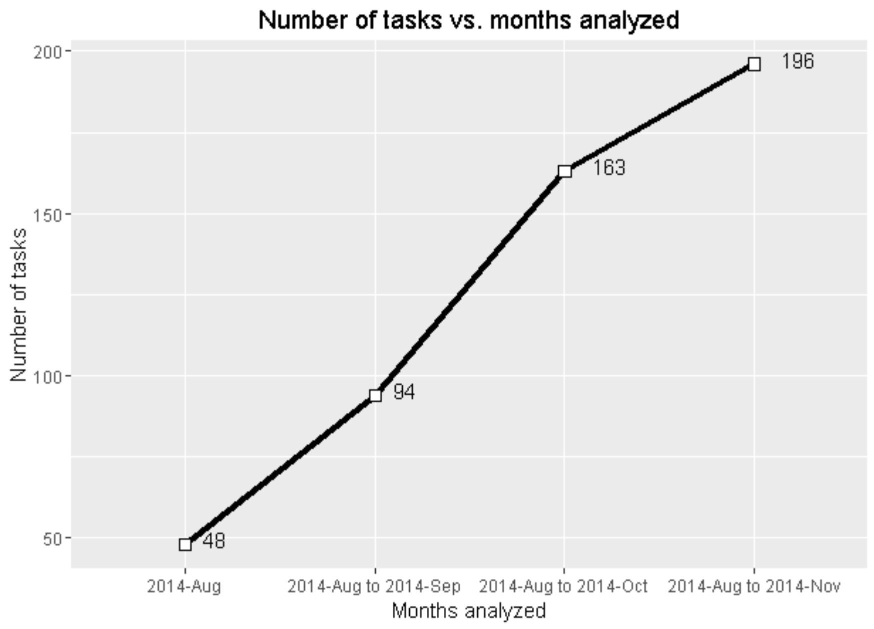

Fig. 5.9 Number of tasks vs. months analyzed

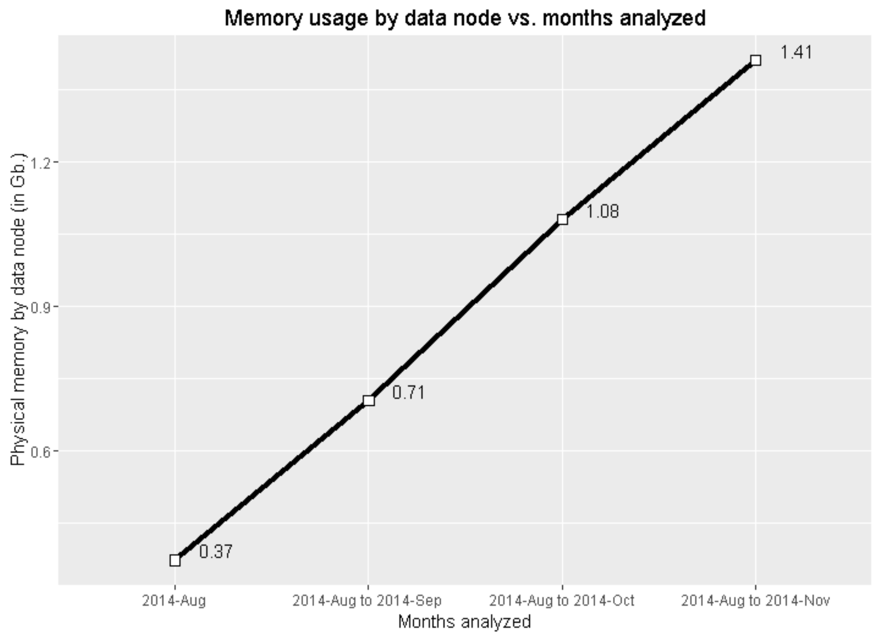

Fig. 5.10 Memory usage by data node vs. months analyzed

The tasks (sum of TOTAL_LAUNCHED_MAPS and TOTAL_LAUNCHED_REDUCES) in each job execution can be studied taking into account the success or failure states (tasks that were killed or failed). This is shown in Fig. 5.11.

The number of HDFS read and write operations can also be represented (see Fig. 5.12) in order to study the data workflow in each problem. This issue is very important since it could reduce the number of failures.

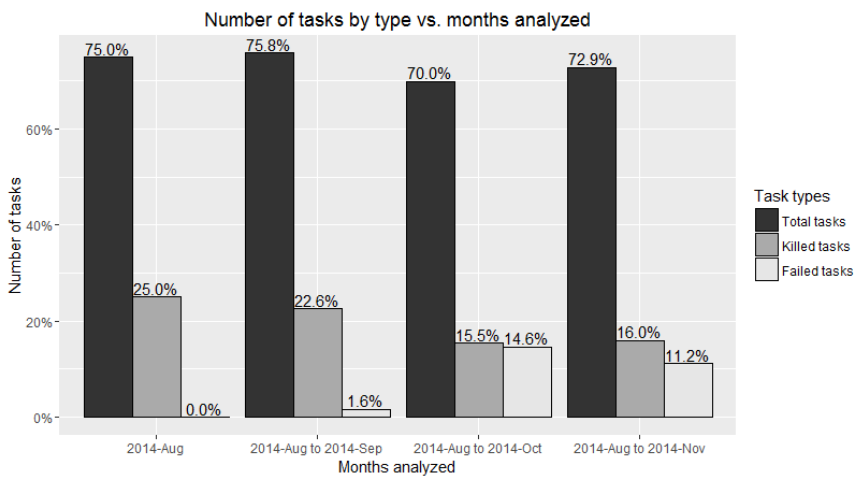

Fig. 5.11 Number of tasks by type vs. months analyzed

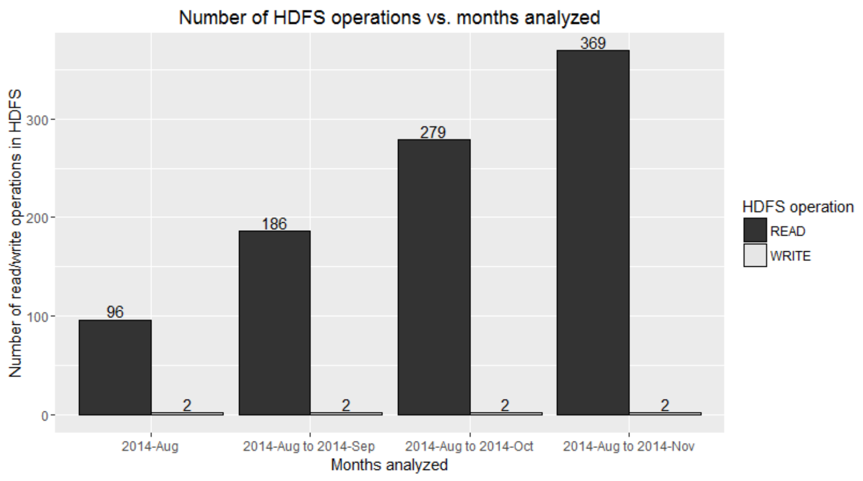

Fig. 5.12 Number of HDFS operations vs. months analyzed

The comparison of counters can be very useful to detect and debug other problems in job executions. The following charts represent the size of data read operation in HDFS (computed from BYTES_READ, see Fig. 5.13) and the memory usage by data node with respect the number of total tasks (Fig. 5.14).

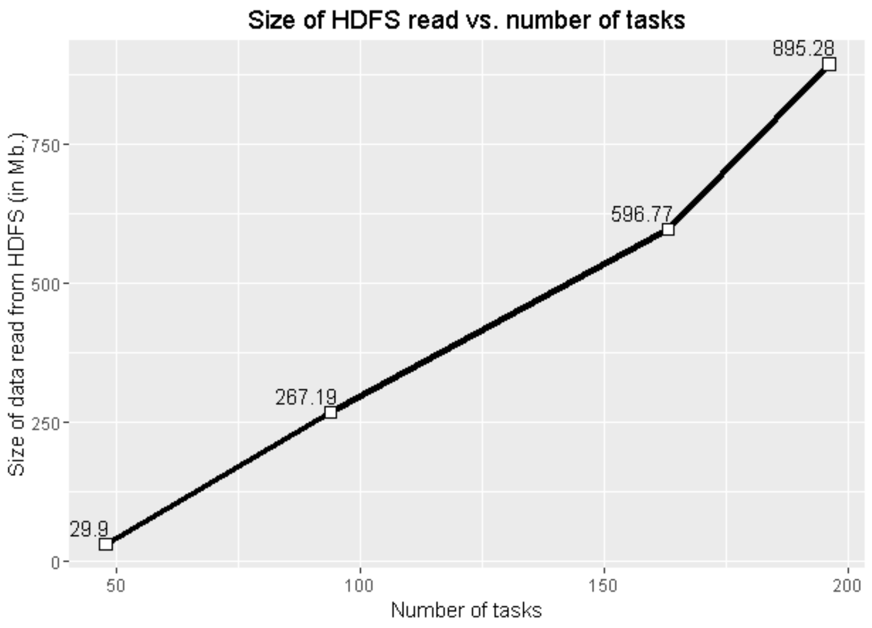

Fig. 5.13 Size of HDFS reads vs. number of tasks

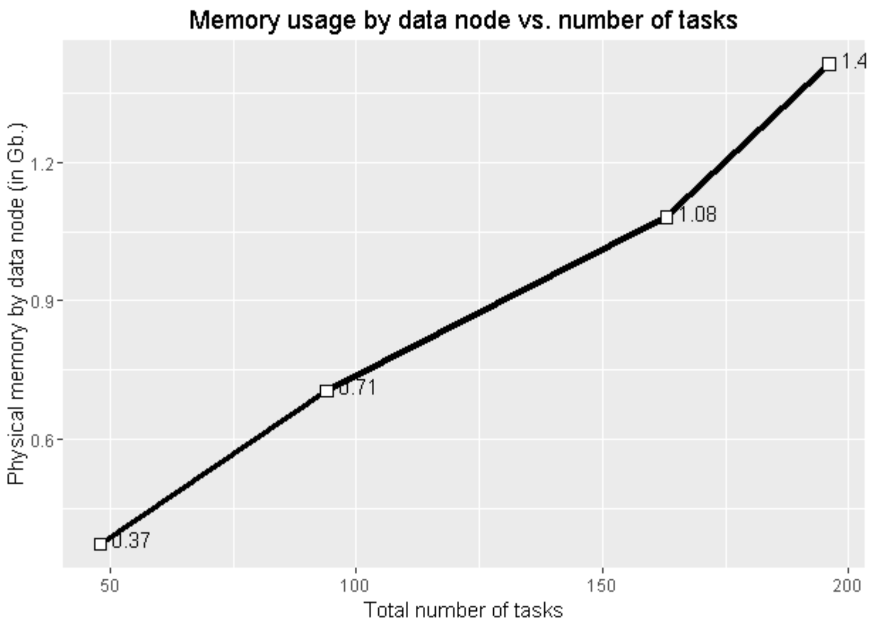

Fig. 5.14 Memory usage by data node vs. number of tasks

In order to compute the CPU time, we multiply the counter CPU_MILLISECONDS by the data size read from the HDFS (that differs considerably between each problem instance). The chart in Fig. 5.15 illustrates the time of computation with respect to the amount of data that is processed.

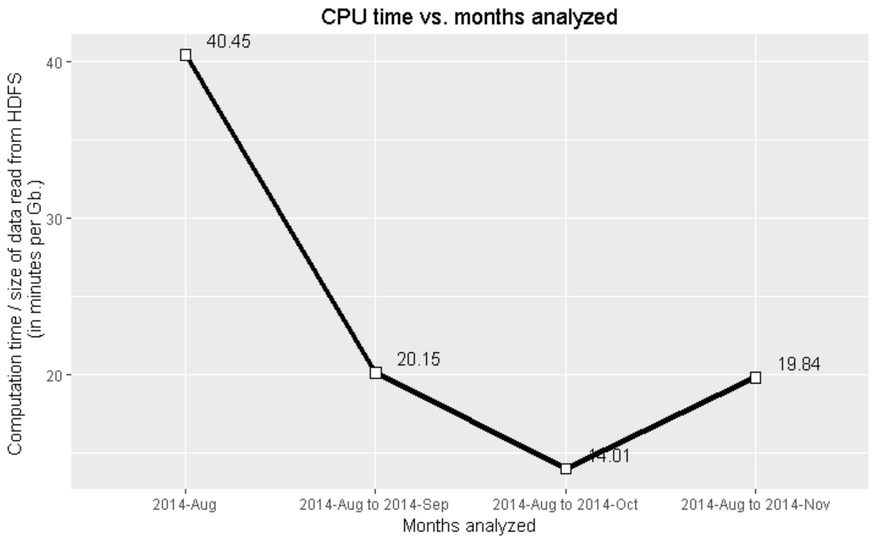

Fig. 5.15 CPU time vs. months analyzed

The counters can be used also to define more complex indicators as a measure of memory consumed in a job execution. In this case, the memory allocated and occupied by tasks is given by:

- Allocated_Memory = (MB_MILLIS_MAPS + MB_MILLIS_REDUCES)
- Occupied_Memory = millis_tot * memory_mb / total_tasks

where:

- millis_tot = MILLI_MAPS + MILLI_REDUCES
- memory_mb = PHYSICAL_MEMORY_BYTES / (1024*1024)
- total_tasks=TOTAL_LAUNCHED_MAPS+ TOTAL_LAUNCHED_REDUCES

Therefore, the percentage of memory allocation used can be computed as PERCENT_MEM_ALLOC = 100 * Occupied_Memory / Allocated_Memory, and the corresponding chart shows in Fig 5.16 the memory usage for each problem instance.

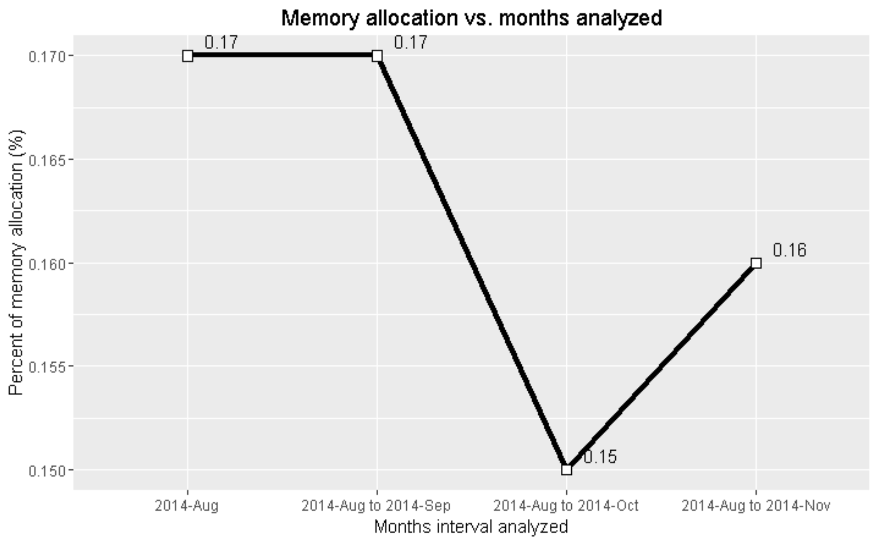

Fig. 5.16 Memory allocation vs. months analyzed

Another indicator of interest could be the percentage of Garbage Collector (GC) time. When a Java application has excessive heap utilization, the corresponding JVM can run a full garbage collection that blocks other works and uses large amounts of CPU. The percentage of time spent in GC is computed as GC_TIME_MILLIS / (MILLIS_MAPS + MILLIS_REDUCES). The following chart in Fig. 5.17 represents this time for each problem instance, and can be considered as a control indicator to prevent this problem.

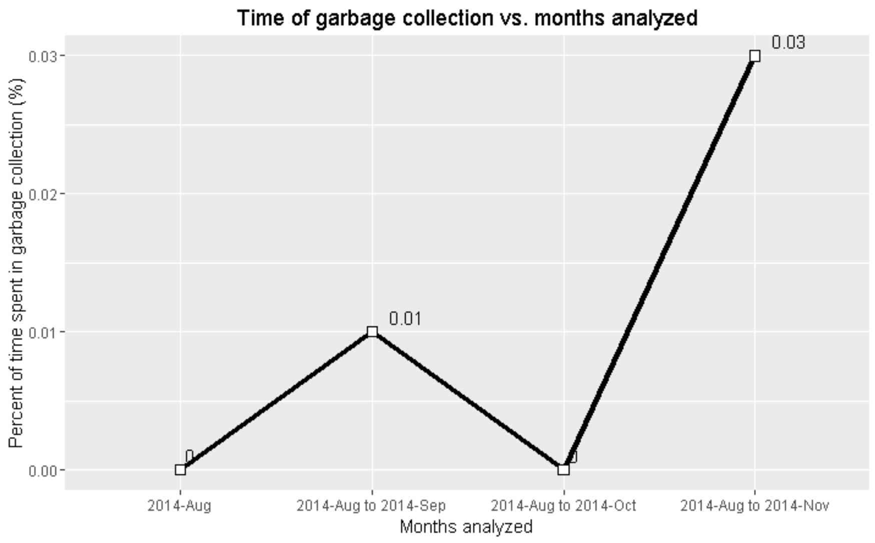

Fig. 5.17 Time of Garbage Collection vs. months analyzed

5.4 Graphical results

The summarized data obtained after processing can be represented by grouping the different servers where the logs come from, as shown in the next Fig. 5.18.

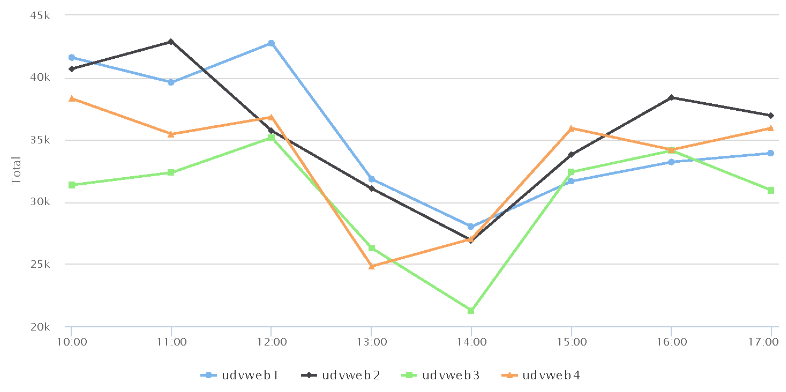

Fig. 5.18 Total number of accesses for different servers along a time slot

The information can also be completed with georeferenced data to show the spatial distribution of web accesses through the different WiFi access points in several centers belonging to different campus of the university, as depicted in Fig 5.19.

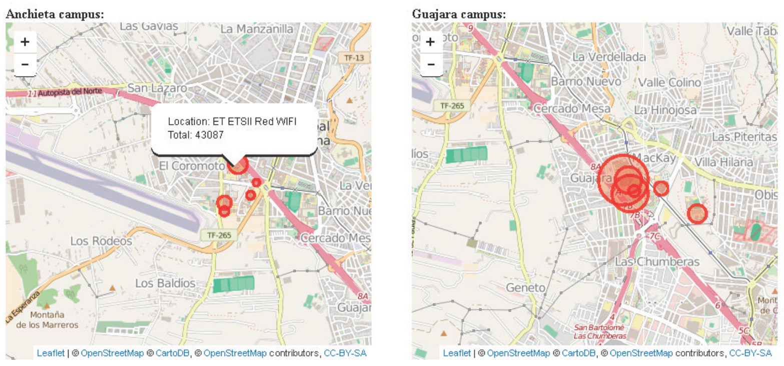
Anchieta campus:
Guajara campus:
Location: ET ETSII Red WIFI
Total: 43087
Leaflet | © OpenStreetMap © CartoDB, © OpenStreetMap contributors, CC-BY-SA

Fig. 5.19 Spatial distribution of the web accesses through the WiFi access points

6 Conclusions

The IT Department of the Universidad de Laguna (STIC) provides service to 26 buildings with more than 1,000 network devices, and renders access to more than 10,000 user devices, which generate around 200 MB/day of log data. With such a huge infrastructure, it is highly desirable to provide new tools to explore this semi-structured data to get insights for the decision making.

In this chapter we have addressed the design and development of an application that uses Big Data techniques to analyze those log files in order to track information on user devices, as well as the number and type of network accesses for each building. Indeed, we have obtained several interesting statistical measures regarding the frequency and type of accesses.

Besides, the collaboration with STIC has tested an iterative and incremental working methodology that has been very useful to obtain quite interesting results to improve both network indicators and analysis metrics.

Accordingly, TOGAF and Archimate become necessary tools when we want to analyze, communicate and maintain complex systems. In this work we present two of the most important viewpoints used in Archimate. The Layered viewpoint gives the developer team a graphical view of the complete WiFi logs system, from the business functions to the infrastructure technology used. An intermediate layer, the Application layer, shows the software components needed to achieve the actors' goals. The second viewpoint, the Application Behavior viewpoint describes in detail the internal behavior of our MapReduce application.

The final result of processing massive log files has proved to be extremely useful to provide very valuable information in a short time. The charts shown in the last sections of this chapter enable to make a clear analysis of our cluster's performance when the different jobs are submitted. In particular, the information depicted in the different figures eases the detection of errors and the control of the success level of all the associated tasks of the different jobs.

Furthermore, we can obtain the total number of accesses for different servers along a time slot and the spatial distribution of the web accesses through the WiFi access points. This is particularly important to audit the service quality in order to define new policies for the system design and the way the users connect to the network.

All these features, along with some more improvements that could allow the analysis of log files in real time, will be studied and developed in future research.

Acknowledgments This work is partially supported by the European Commision, Agreement no. 621012, "Share PSI 2.0: Shared Standards for Open Data and Public Sector Information", ICT Policy Support Programme as part of the Competitiveness and Innovation Framework Programme. By the Spanish Ministry of Education and Science, Research Project MTM2013-43396-P, National Plan of Scientific Research, Technological Development and Innovation. And by the Cabildo de Tenerife, through the Open-Big-Smart Data Project. The authors wish to thank Adrián Muñoz-Barrera, Luis A. Rubio-Rodríguez and Pedro González-Yanes for their support and assistance both in the configuration and deployment of the Hadoop cluster and in the development of the solution.

References

[1] The Zettabyte Era. http://www.cisco.com/en/US/solutions/collateral/ns341/ns525/ns537/ns705/ns827/VNI_Hyperconnectivity_WP.html. Accessed May 2016.

[2] Intel (2014) What Happens in an Internet Minute? http://www.intel.es/content/www/es/es/communications/internet-minute-infographic.html. Accessed May 2016.

[3] Vaarandi R, Niziński P (2013) A Comparative Analysis of Open-Source Log Management Solutions for Security Monitoring and Network Forensics. CCDCOE – NATO Cooperative Cyber Defence. http://ccdcoe.org/multimedia/comparative-analysis-open-source-log-management-solutions-security-monitoring-and-network.html. Accessed May 2016.

[4] What is Big Data? (in Spanish). http://www.ibm.com/developerworks/ssa/local/im/que-es-big-data. Accessed May 2016.

[5] Nair R, Narayanan A (2012) Benefitting from Big Data: Leveraging Unstructured Data Capabilities for Competitive Advantage. Booz & Company. http://www.strategyand.pwc.com/media/file/Strategyand_Benefiting-from-Big-Data.pdf. Accessed May 2016.

[6] Bloem J, van Doorn M, Duivestein S, van Manen T, van Ommeren E (2012) Creating clarity with Big Data. SOGETI. http://blog.vint.sogeti.com/wp-content/uploads/2012/07/VINT-Sogeti-on-Big-Data-1-of-4-Creating-Clarity.pdf. Accessed May 2016.

[7] Laney D (2012) Deja VVVu: Others Claiming Gartner's Construct for Big Data. http://blogs.gartner.com/doug-laney/deja-vvvue-others-claiming-gartners-volume-velocity-variety-construct-for-big-data. Accessed May 2016.

[8] Soubra D (2012) The 3 Vs that define BigData. Data Science Central. http://www.datasciencecentral.com/forum/topics/the-3vs-that-define-big-data. Accessed May 2016.

[9] Yiu C (2012) The Big Data Opportunity. Policy Exchange. http://www.policyexchange.org.uk/images/publications/the%20big%20data%20opportunity.pdf. Accesssed May 2016.

[10] TechAmerica Foundation (2012) Demystifying Big Data: A Practical Guide To Transforming The Business of Government. Available at: https://www-304.ibm.com/industries/publicsector/fileserve?contentid=239170. Accessed May 2016.

[11] NIST (2015) Big Data Interoperability Framework: Volume 1, Definitions. http://dx.doi.org/10.6028/NIST.SP.1500-1 . Accessed May 2016.

[12] Davenport T, Harris J (2007) Competing on Analytics. Harvard Business School Press. Boston, MA.

[13] SAP (2011) Making Business Run Better with In-Memory Computing & Predictive Analytics. http://scn.sap.com/docs/DOC-5024. Accessed May 2016.

[14] ITU-T (2015) Big Data – Cloud computing based requirements and capabilities. http://handle.itu.int/11.1002/1000/12584. Accessed May 2016.

[15] Apache Hadoop. http://hadoop.apache.org. Accessed May 2016.

[16] Hu H, Wen Y, Chua TS, Li X (2014) Toward Scalable Systems for Big Data Analytics: A Technology Tutorial. IEEE Access 2:652-687.

[17] Pentaho. http://www.pentaho.com . Accessed May 2016.

[18] The Open Group Architecture Framework (TOGAF) Version 9.1. The Open Group. http://www.opengroup.org/togaf. Accessed May 2016.

[19] Lankhorst MM (2004) Enterprise architecture modelling—the issue of integration. Advanced Engineering Informatics 18(4):205–216.

[20] The R Project for Statistical Computing. https://www.r-project.org. Accessed May 2016.

[21] RStudio. https://www.rstudio.com. Accessed May 2016.

[22] Shiny. http://shiny.rstudio.com. Accessed May 2016.

[23] Laserson U (2013) A Guide to Python Frameworks for Hadoop. http://blog.cloudera.com/blog/2013/01/a-guide-to-python-frameworks-for-hadoop. Accessed May 2016.
[24] White T (2015) Hadoop: The Definitive Guide. O'Reilly Media.