Scientific Research Publishing

Journal of Computer and Communications, 2023, 11, 79-102
https://www.scirp.org/journal/jcc
ISSN Online: 2327-5227
ISSN Print: 2327-5219

Optimization of Mobile Network Radio Coverage by Automating Radio Parameter Updates Using Parsing

Patrick Dany Bavoua Kenfack, Alphonse Binele Abana, Emmanuel Tonye, Nadège Laure Bemehemie, William Tchofo Tchouleko

Department of Electrical and Telecommunications Engineering, National Advanced School of Engineering of Yaoundé, University of Yaoundé I, Yaoundé, Cameroon
Email: danybavoua@gmail.com

How to cite this paper: Kenfack, P.D.B., Abana, A.B., Tonye, E., Bemehemie, N.L. and Tchouleko, W.T. (2023) Optimization of Mobile Network Radio Coverage by Automating Radio Parameter Updates Using Parsing. *Journal of Computer and Communications*, **11**, 79-102.
https://doi.org/10.4236/jcc.2023.114005

Received: February 9, 2023
Accepted: April 25, 2023
Published: April 28, 2023

Abstract

The present work aims is to propose a solution for automating updates (MAJ) of the radio parameters of the ATOLL database from the OSS NetAct using Parsing. Indeed, this solution will be operated by the RAN (Radio Access Network) service of mobile operators, which ensures the planning and optimization of network coverage. The overall objective of this study is to make synchronous physical data of the sites deployed in the field with the ATOLL database which contains all the data of the coverage of the mobile networks of the operators. We have made an application that automates, updates with the following functionalities: import of radio parameters with the parsing method we have defined, visualization of data and its export to the Template of the ATOLL database. The results of the tests and validations of our application developed for a 4G network have made it possible to have a solution that performs updates with a constraint on the size of data to be imported. Our solution is a reliable resource for updating the databases containing the radio parameters of the network at all mobile operators, subject to a limitation in terms of the volume of data to be imported.

Keywords

Radio Parameters, Parsing, ATOLL Database, OSS NetAct, ETL

1. Introduction

Following the technological evolution and the digitalization of the world which currently impose new lifestyles which are: telework, telemedicine, teleconferencing, telesurveillance, e-commerce, e-learning, etc., we are witnessing new

profound and sometimes irreversible transformations in habits. In this upheaval, telecommunications appear as the nerve of digital. As a result, the infrastructures of mobile telephone operators should be efficient and capable of continuously ensuring a good quality of service for an ever-increasing demand.

And data is the most important thing in the network for mobile operators. For example, the mobile operator Orange Cameroon collects, stores, processes, analyzes data on a daily basis and makes manual updates of the radio parameters of its network. These updates are tedious and require a lot of attention and rigor in order to avoid the slightest error that would impact the analyses constantly carried out on the network.

IBM and ORACLE offer external management tools for updating databases such as "DATALINK" [1], inspired by the ETL concept [2], a Middleware-type technology, which has currently evolved considerably and meets the needs related to the rise of Cloud, SaaS and Big Data [3]. These tools are very expensive.

The RAN service continuously optimizes and decongests its network to guarantee better coverage. He should therefore keep his database up to date. It is with this in mind that our research work is aimed at a solution for automating the updates of the ATOLL radio parameters (database containing all the network data) from the OSS NetAct.

Our objective is therefore to set up an application whose functionalities will be among others:

- Automatically manage the import of new DUMPs and analyze them continuously.
- Group the parameters by type of technology (2G; 3G; 4G), by sites, by transmitters, by cells and make automatic updates.
- Give the possibility to the user to visualize and check the accuracy of the network parameters or to readjust them if necessary.
- Export data on file on demand and insert them into the ATOLL database.

2. Context of Our Study

The system for updating the database that contains the parameters of the radio access networks for most mobile phone operators is done manually, in this case with some operators where the method consists of "parsing" and processing data manual from OSS NetAct. Here, Macros are used to "parse" the collected data. Then another processing method is used; this consists of using Excel functions. This system has limitations:

- Time for treatment is long;
- Unreliable treatment method

In order to solve the problem posed, there exist in the literature some works carried out in the field and some tools and methods for automating the process.

According to [4] who worked on the Automation of the capacity management of the RAN subsystem of the mobile networks which consisted in setting up a tool which makes it possible to automate the management of the capacity of the

equipment and the interfaces of the network 2G and 3G mobile operator access. Indeed, this tool makes it possible to size the capacity of the interfaces and equipment of the RAN subsystem, to monitor the KPIs and to optimize the capacity of the RAN subsystem. The problem with this tool is due to the fact that this tool only manages 2G and 3G networks. Moreover, it is specific to an operator but the approach approached is very interesting.

According to [5], who worked on the Design of a tool to assist in the automatic management of the quality of service of UMTS/4G LTE radio access networks, it was a question of setting up a tool which makes it possible to automate the assessment of the QoS of UMTS/LTE access networks [5].

Indeed, the work made it possible to collect information concerning the KPIs, to highlight the real traffic, and to make forecasts. The problem with this tool is that it does not manage voice traffic forecasting in 2G networks followed by TRU sizing.

3. Data Automation Techniques and Tools

3.1. Techniques: The Xml Parser

An XML parser (or parser) is a computer program capable of analyzing an XML document in order to extract its content for better use. XML for Extensible Markup Language (extensible markup language) is a computer language that allows the exchange of data between two heterogeneous environments. An XML document is a tree composed of nodes which can be elements or attributes.

The ancestor of XML is the SGML language which was introduced in 1986 by C. Goldfarb. SGML was designed for large-scale technical documentation. And in 1991, T. Berners-Lee defined the HTML language for the WEB. Thus the XML language has become ubiquitous in the life of many companies at the database level (IBM; ORACLE; MICROSOFT; Nokia Siemens Networks) and presents as:

- **Advantages**
  - Incredible reading speed for small file sizes,
  - Very fast data access for small file sizes,
  - Great compatibility with different system languages for reading, analysis, processing and conversion (pdf, xls, csv, html).
- **Disadvantages**
  - The excessively long time, for large files to process, reading, accessibility to content, indexing, and sorting of data according to their number.

So for a better use of Xml files, you have to parse them. There are two main families of parsers:

- Non-validating parsers, which simply check that the XML document is well-formed.
- Validating parsers, which verify that the XML document is validated.

The parser is therefore used to "cut" the data file into a set of words in order to extract the hierarchical structure from it. It is associated with a processing

module which has a very important role: it is the XML Processor.

There are two types, each associated with each type of parser namely:

- SAX (Simple API for XML) processors that are event-driven: provide an event-driven interface for traversing an XML document. This API indeed sends back to the application which manipulates the XML document "events" (tag opening, tag closing, textual content, etc.) which are generally associated with the family of non-validating parsers.
- Hierarchy-oriented DOM (Document Object Model) processors: Unlike SAX processors, they use a hierarchical approach. They allow easy navigation in a document but require the complete loading into memory of its tree structure. They therefore store a DOM tree and are associated with the family of validating parsers. The types of parsers are shown in **Figure 1**.

3.2. Tools of the ETL Process

According to the Office Québécois de la Langue Française (OQLF), updating is "an operation which consists of adapting the hardware or software so that it complies with the most recent developments or modifying the data to that they reflect the most current information".

This updating of data or knowledge is essential before each decision-making process. It would be even more so if it were automated.

It is quite easy to do manual updates but limited in terms of capacity and time, for a large volume of data. This method is complex because it presents many challenges such as: very complex administration, technical support and code reuse.

The process of updating radio parameters is very complex and requires a lot of engineering for its design.

Hence the need for the engineering of automatic update functionalities [7], described by tools of the ETL process which are generally more scalable, less costly over time, more efficient and tend to meet the requirements of the concepts of business intelligence [8].

Indeed, the direct or indirect changes of the physical parameters of the deployed sites imply that the DB should be kept up to date in order to maintain consistency between the data stored in the DB and the actual data in the field. So

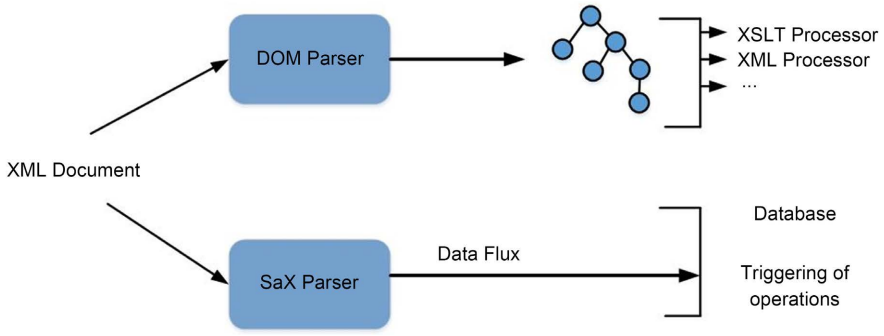

Figure 1. Parser's type [6].

for such a decision-making process, the engineering of automatic updates features uses methods that are exhaustively described in the literature and with good practical applications demonstrated by ETL concepts [2].

Typically The ETL process (see Figure 2) is a system consisting of a sequence of operations aimed at extracting data from all sources and loading them into various Data Warehouses or Data Lakes [9]. It allows, for a good automated management of the data, to ensure a good coherence to improve the integration of the data of several sources towards only one.

There are two families of these processes:

- **The ETL (Extract, Transform, Load) process** which is a traditional integration approach and collects information from remote sources, transforms it into defined formats and styles, and then loads it into databases as shown in Figure 2.
- **The ELT (Extract, Load, Transform)** process which is a more scalable approach that also extracts data from one or more remote sources, but then loads it into the target data warehouse with no change in format as shown Figure 3.

In an ELT process, data transformation takes place within the target database. Here the process is much faster than in ETL.

While using ETL/ELT tools, we describe:

- **The extraction of data** which is carried out from a set of heterogeneous data sources, structured or not (relational databases, web pages, text files) as shown in Figure 4.
- **Data transformation** which defines the rules and models so that they are compatible with the structure of the target database. This process includes: interpreting (reading), cleaning (removing unused data) and contextualizing the data as we see in Figure 5.

Figure 2. ETL process [10].

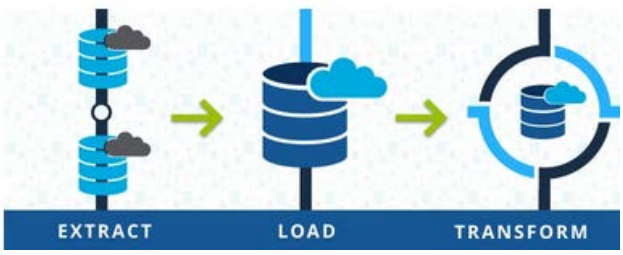

Figure 3. ELT process [10].

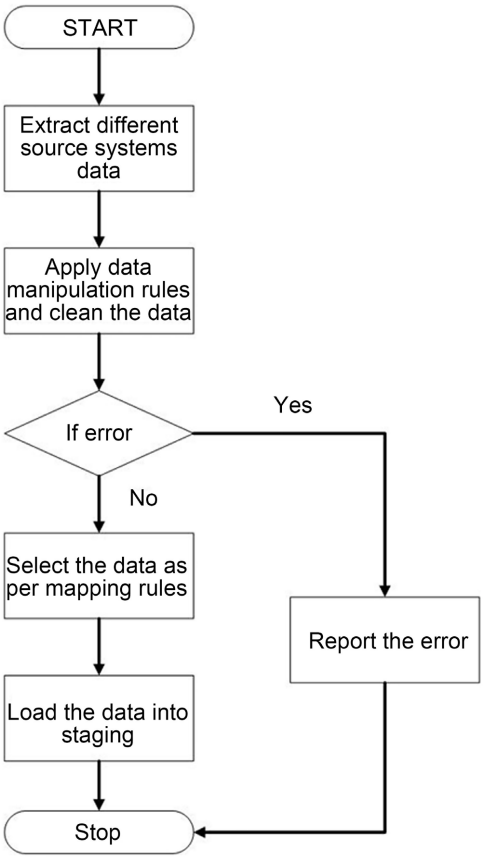

Figure 4. Extraction flow diagram [11].

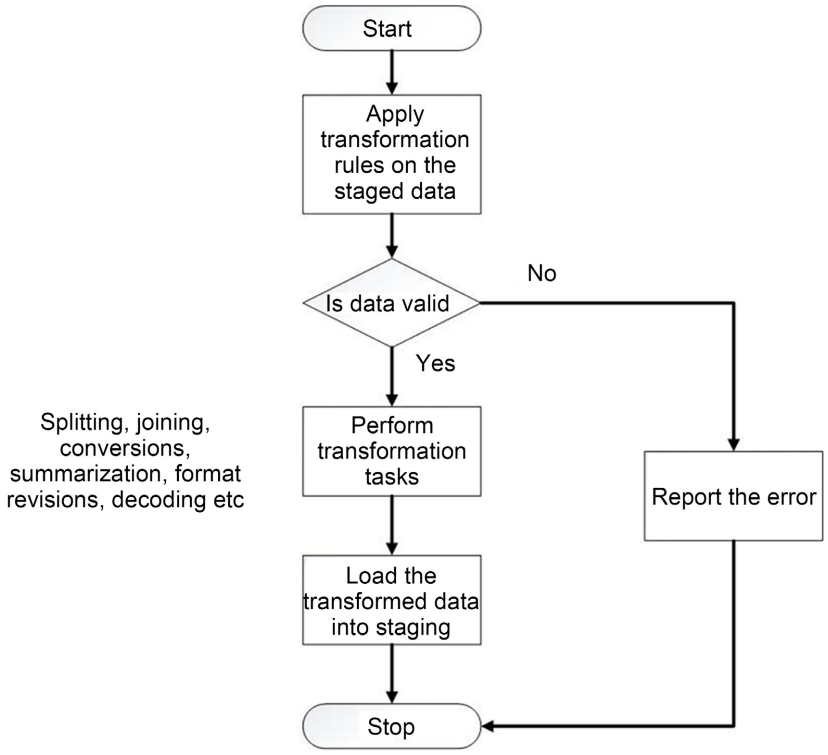

Figure 5. Transformation flow chart [11].

- **Data loading** which consists of loading and integrating the extracted and processed data into the database in such a way as to guarantee the consistency and uniqueness of the data as shown in Figure 6.

Figure 7 presents the summary of the ETL process.

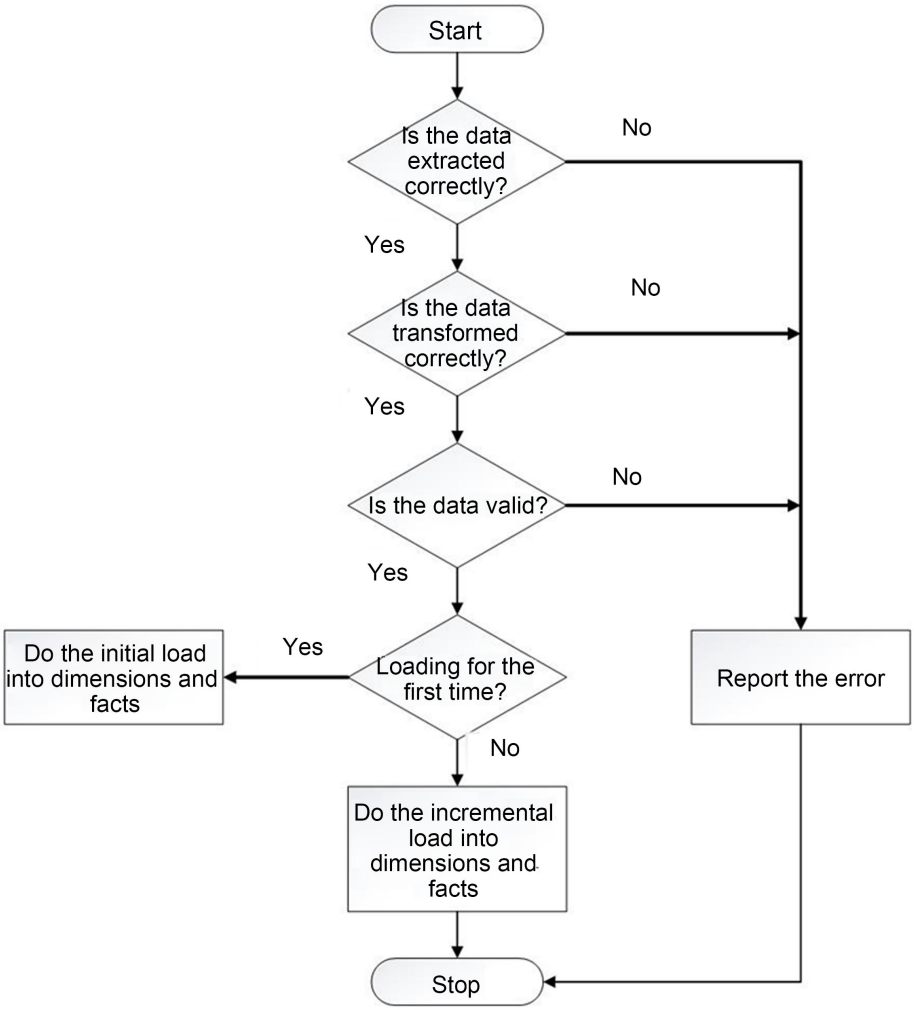

Figure 6. Load flow diagram [11].

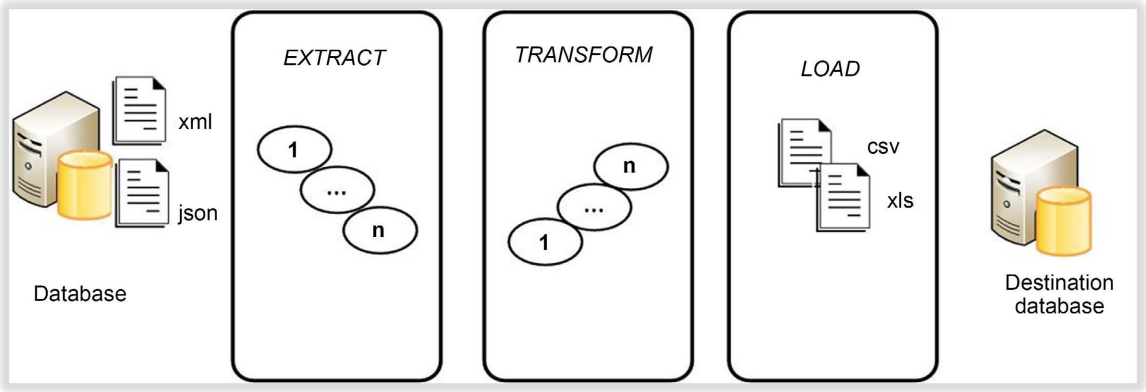

Figure 7. ETL process [12].

4. Methods and Tools

4.1. Insight

Thus, for the realization of our tool for automating updates of radio parameters from NetAct, we considered the need to design our own parser of XML files in the Python environment, a multiplatform and multiparadigm interpreted programming language which integrates other programming languages like Java and C.

However, the developed parser will borrow the analysis methods of validating parsers which build a hierarchical tree structure by creating a series of objects in memory representing a complete tree structure of the document to be parsed.

Here the method is:

- First iterate through the entire file creating the root of the tree each time;
- Fix on the latter its various objects, nodes, attributes;
- Load in memory in the form of a tree before parsing the parameters and extracting the content in .xls format.

4.2. Type of Processing

According to the current context which wants us to design a tool for automating the updates of the radio parameters of the ATOLL base from the OSS NetAct, to date there is no tool for "parsing". The method of updating the ATOLL database consists of a set of search and concatenation formulas on the Excel spreadsheet, therefore essentially manual.

In order to meet the stated objectives, file processing begins with:

A) data collection at the OSS NetAct level.

The latter provides Dump.xml files whose content is not accessible.

This is why we apply the parsing method chosen for this purpose: a script written in the python programming language that imports a set of libraries, such as:

- **Pandas** which is a library written in python programming language allowing data manipulation and analysis and has tools to read and write structured data in memory to different formats: CSV files, text files, Microsoft Excel spreadsheet file, database.
  - **Numpy** which is a library acting on multidimensional arrays, intended for:
- Has scientific analysis;
- A linear algebra;
- A Matrix calculation.
  - **Xlsxwriter** which is a Python module to write files in XLSX format. It supports features like formatting, images, charts, page layout, auto filters, conditional formatting and many more.

Thus the processing gives us the files (see **Figure 8**) which contain all the modules of the radio parameters necessary for the updates of the ATOLL base.

This file is a Dump of the 4G network containing all the parameters of said network, structured according to the tree structure shown in **Tables 1-4** below.

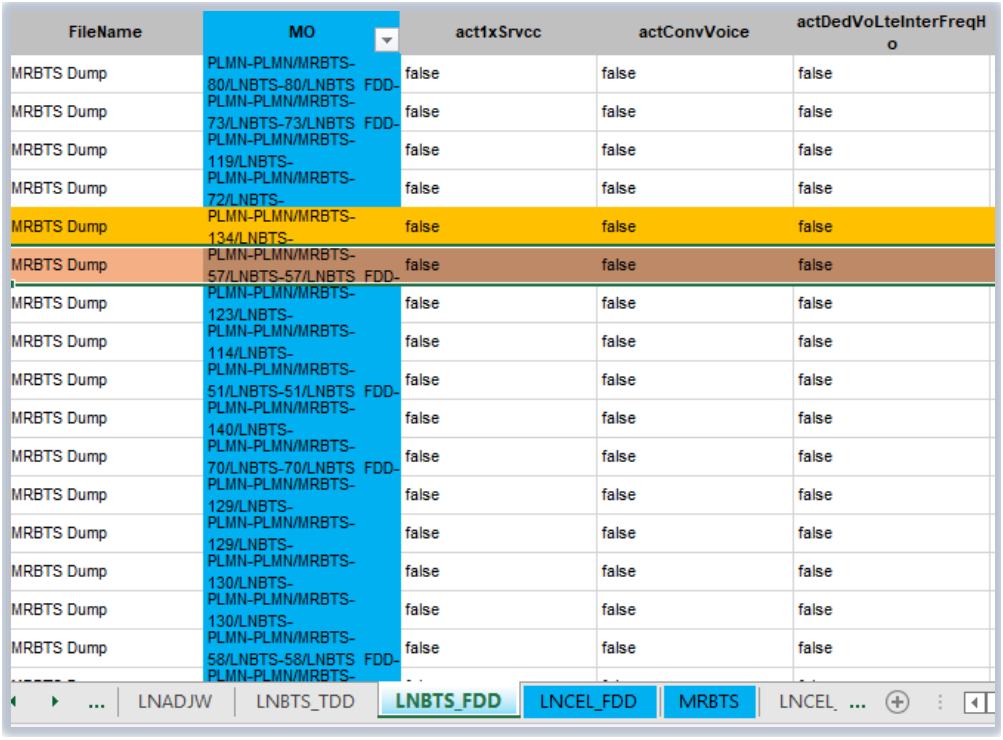

Figure 8. Structure of a Dump.xls parser file [12].

Table 1. Structure of the LTE dump.

Leaves	Information of the leaves
MRBTS	Contains all the parameters returned by the eNoDeBs
LNCEL	Contains all cell settings for a site
AMLERP	Contains all deployed site frequency bands
LNBTS	Has all the eNB modules of the 4G network
INVUNIT	Contains types of radio antennas
BTSNE	Contains ID_codes and site_names

Table 2. Structure of managed object LNCEL.

LNCEL: contains all the configuration parameters of the updated cells		
Columns	**Information contained**	**Explanations**
Cellname	Contains cell names of deployed sites	This column is used to fill in the issuer names, cells
Angle	Contains electrical tilt angles: the tilts of each transmitter, cell	Here we recover the electrical tilts of the transmitters, cells (in base 10)
pMax	Contains the maximum powers of the cells in E/R	Here we recover the power of a cell in base (10) in dbm
phyCellid (PCI)	Contains the unique identifiers of the physical layers of cells	PCI is used to identify the cell during the cell selection procedure, Proper assignment of PCIs reduces call losses, by allowing the UE to clearly distinguish one cell from another

Table 3. Structure managed object AMLERP.

AMLERP: contains all the configuration parameters of the updated cells		
Columns	**Information contained**	**Explanations**
Targetcarrierfreq	The cell-level frequency bands of the sites	Here we find the frequency bands in MHz

Table 4. Structure of managed object INVUNIT.

INVUNIT: contains the types of radio antennas of the LTE network		
Columns	**Information contained**	**Explanations**
vendorUnitTypeNumber	Contains the types of antennas deployed on the sites	Here each antenna type has the characteristics of its manufacturer

B) Implementation in our platform of several methods described by the update automation engineering.

It consists of an ETL (Extract, Transform, Load) method described in the State of the Art, and reporting:

- **The extraction** [13] which consists of browsing, for each LTE network adjacency, the files already parsed to extract all the columns containing the radio parameters to be entered in the ATOLL database.
- **The transformation** which interprets by network adjacency, the different columns containing the radio parameters, then deletes the columns which are useless and in the end, contextualizes each column that is indexed to the ATOLL Template.

In this step we work with the MO columns which contain a hierarchical tree structure
"PLMN-PLMN/BSC-402473/BCF-106/MRBTS-1/LNBTS-50475/LNCEL-1"
which serves as links between the adjacencies of the network.

- **Loading** which consists of loading and integrating the data that has been extracted and transformed into a data format to be inserted into the ATOLL database as shown in Figure 9.

4.3. File Processing Algorithm

The diagram in Figure 10 provides a summary of the processing algorithm used.

4.4. Conception of the Data Base

4.4.1. System Analysis

➢ Functional requirements

The diagram in Figure 10 provides a summary of the processing algorithm used.

- **Management of Dump imports:** The application must import and parse the raw files from the OSS NetAct entirely. We will therefore have the following functionalities:

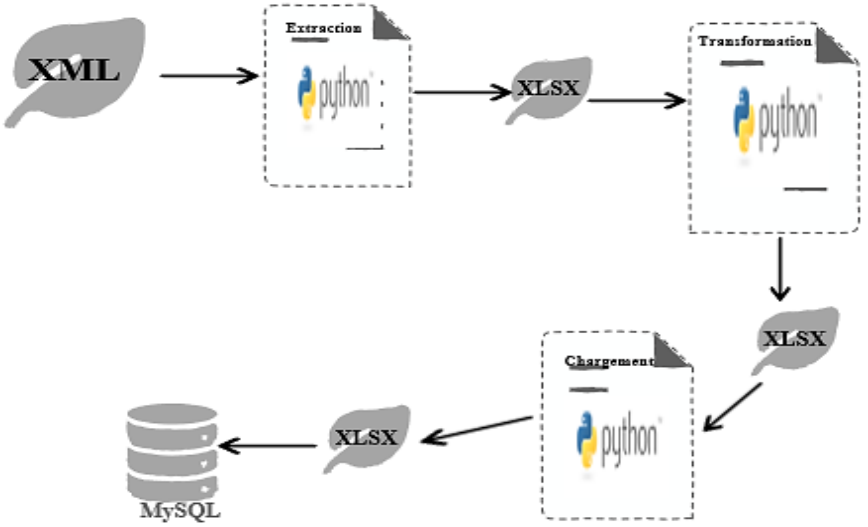

Figure 9. File processing by the ETL process [13].

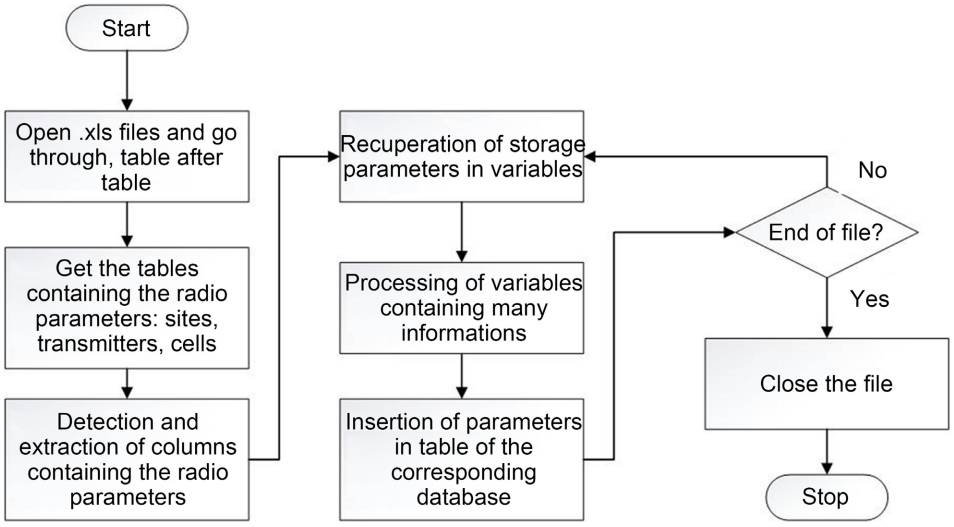

Figure 10. File processing algorithm.

- Presentation of the import interface for importing the .xml format dump;
- Filtering of Dump import by technology (2G, 3G, 4G);
- Update automation management: the application shall, after parsing the raw files, display the radio parameters according to the ATOLL Template requiring:
  - Automatically retrieve network adjacencies by technology and radio parameters by column for viewing;
  - Automatically process links for indexing with ATOLL tables;
  - Add, remove and verify manually before doing automatic updates.
- **Export management:** the user will be able to:
  - Export by technology the various updated parameters;
  - Match the exported files to the Template of the ATOLL base to save.
- **User Management:** User will be able to:

- Login, Logout;
- Change password and reset.

➢ **Non functional requirements**

- The application should be responsive and robust able to run on any type of environment;
- The nomenclature of the files which contain the radio parameters to be inserted in the ATOLL database must be read by the EXCEL software (.CSV, .XLS).

4.4.2. System Modelization

➢ **Architecture choice**

There are different types of application architectures, among which we can cite:

- Centralized architecture, in this type of architecture the data and the application are located in the same place, it has the advantage of being flexible in terms of its administration. Its disadvantages are such that if a component is defective then the whole system is affected and also at the level of resource management.
- The MVC (Model-View-Controller) software architecture: This is a model intended to meet the needs of interactive applications by separating the issues related to the different components within their respective architecture. Organizing a GUI is tricky. The "MVC" architecture does not claim to eliminate all problems, but provides a first approach to do so. Offering a standardized framework for structuring an application, it also facilitates dialogue between designers.

The idea is to separate data, presentation and processing. This paradigm groups the necessary functions into three categories: A model (data model), a view (presentation, user interface), a controller (control logic, event management, synchronization).

After this brief analysis, our choice fell on the MVC software architecture in order to facilitate future corrective and evolutionary maintenance of the web application. Thus, it is this logic that we will follow from the modeling of our system to the design of our tool.

4.5. Architecture of the Application

4.5.1. Presentation

Figure 11 presents the application architecture.

➢ **Functioning of the application**

1) The OCM engineer Ran connects from his machine to the FTP server where he retrieves the Dumps files that have been deposited in the remote OSS NetAct server;

2) It issues a set of requests on the NETOL application to: import, parse, view, verify and correct and update, radio parameters of deployed sites;

3) It issues a request to export the updated files which will be stored later in the ATOLL database.

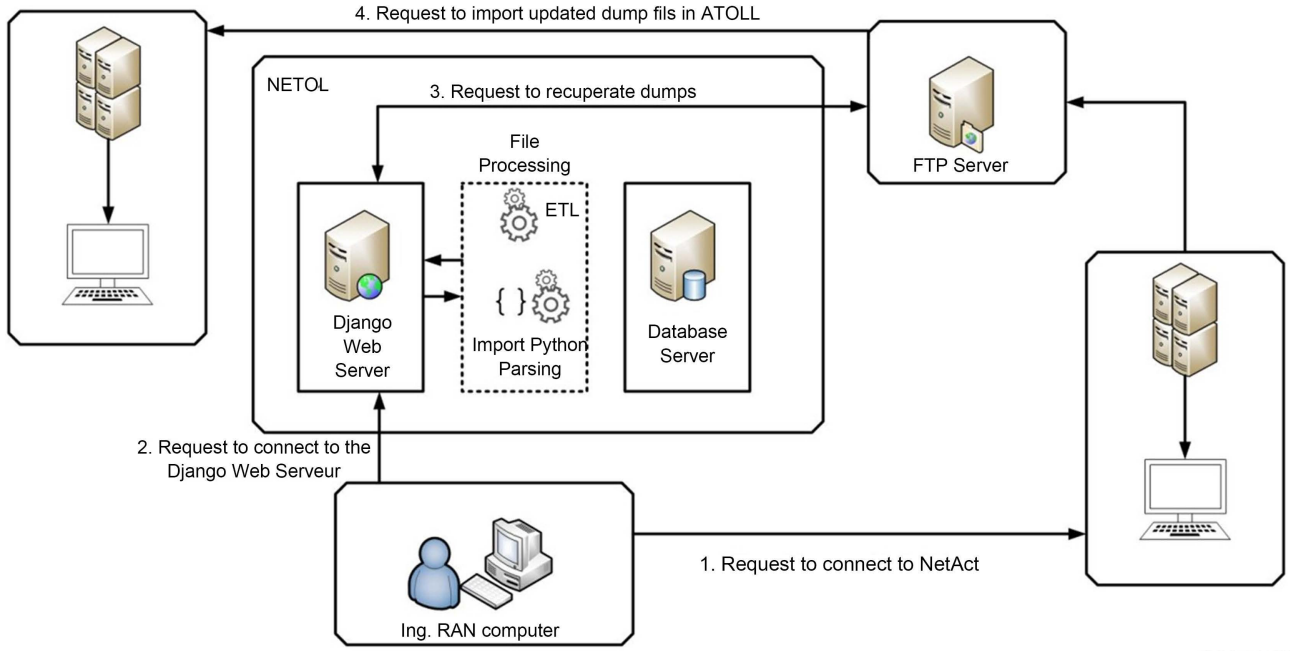

Figure 11. Application architecture.

4.5.2. Use Case Diagram

The use case diagram makes it possible to express the needs of the users of a system by collecting, analyzing and organizing the needs of the users and by identifying the main system features.

- **Identification of system actors**

The actors or users of a system are entities that interact directly with this system. These actors can also be external applications that respond to calls made by the system.

By analyzing the functionalities of the tool we can distinguish the following actors:

- Ran Engineer: is the main user of the system, having the need to synchronize updates automatically between the ATOLL and NetAct databases containing the physical parameters of the sites deployed in the network;
- The administrator: who is a second actor for the system, he administers the system by giving access rights to users.
- The system, internal actor, is our application, which will have the role of sending and receiving information to users.

- **Identification of use cases**

In the design we have formulated the different use cases meeting the following specifications according to Figure 12:

- **Import the Dump:** this involves retrieving the xml files containing the physical parameters of the sites deployed by NOKIA from their NETACT platform, which will then have to be converted (parser to .csv, .xlsx);
- **Update network parameters:** this is to display the parameters that have been parsed and store them in a Template that can be retrieved by the ATOLL database;

- **Check for changes:** this is to check the changes made by the system, while thinking about making manual readjustments;
- **Consult the history:** here the system gives us the possibility of exporting the updated data in order to insert them into the ATOLL database, as well as checking all the modifications made in the application;

4.5.3. System Sequence Diagram

The sequence diagrams in Figure 13 and Figure 14 make it possible to represent

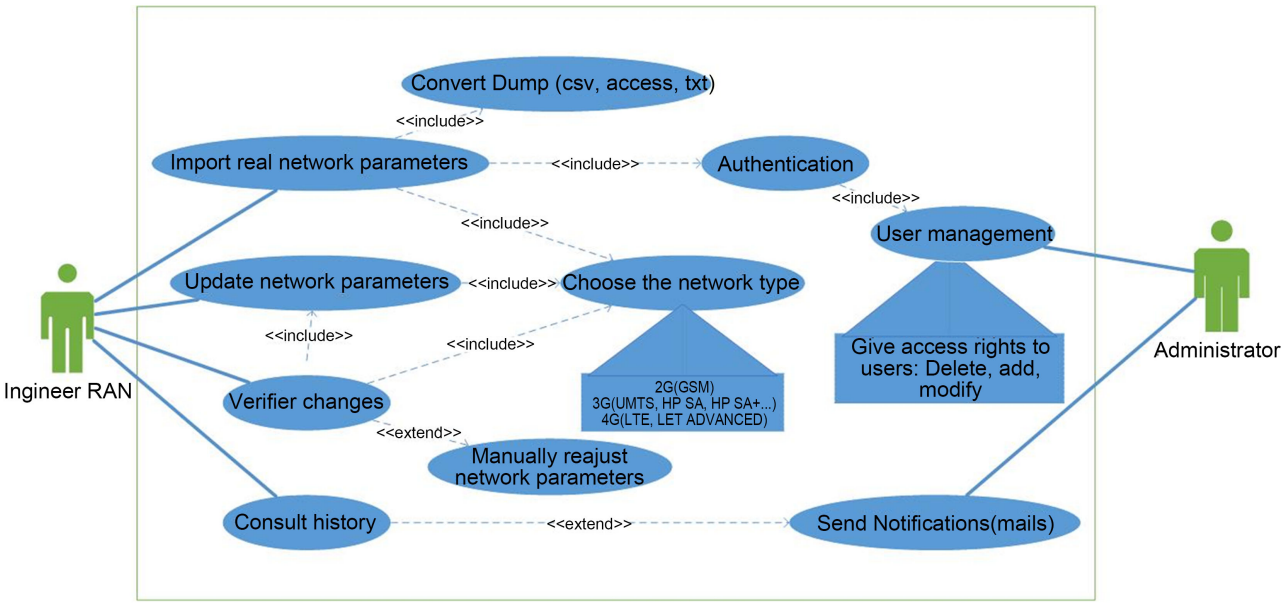

Figure 12. System use case diagram.

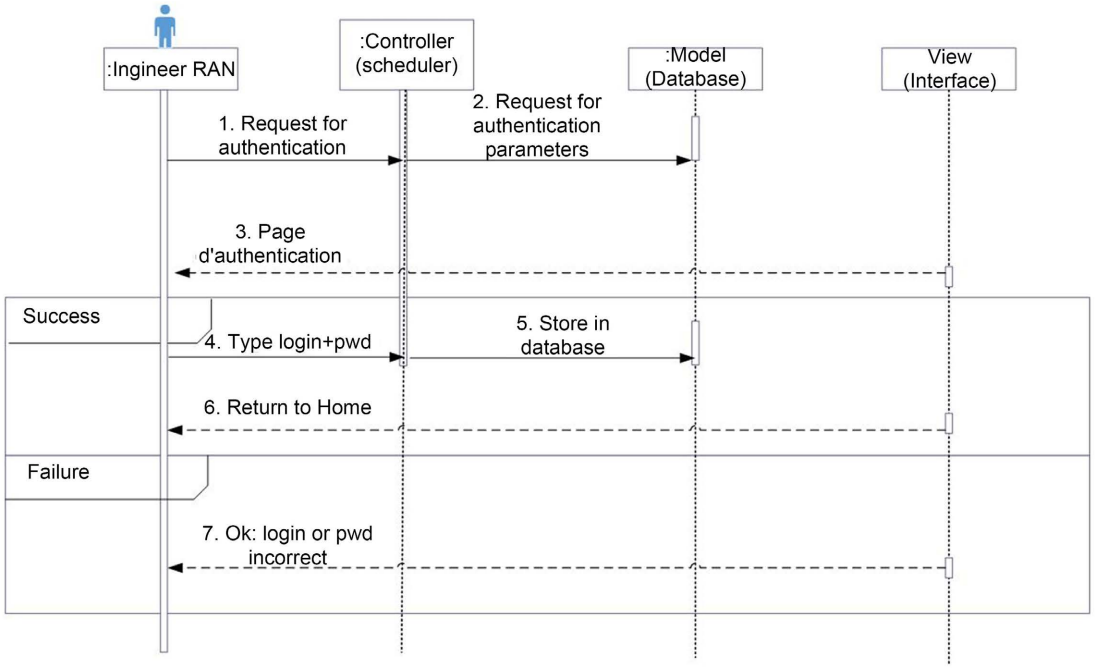

Figure 13. Authentication sequence diagram.

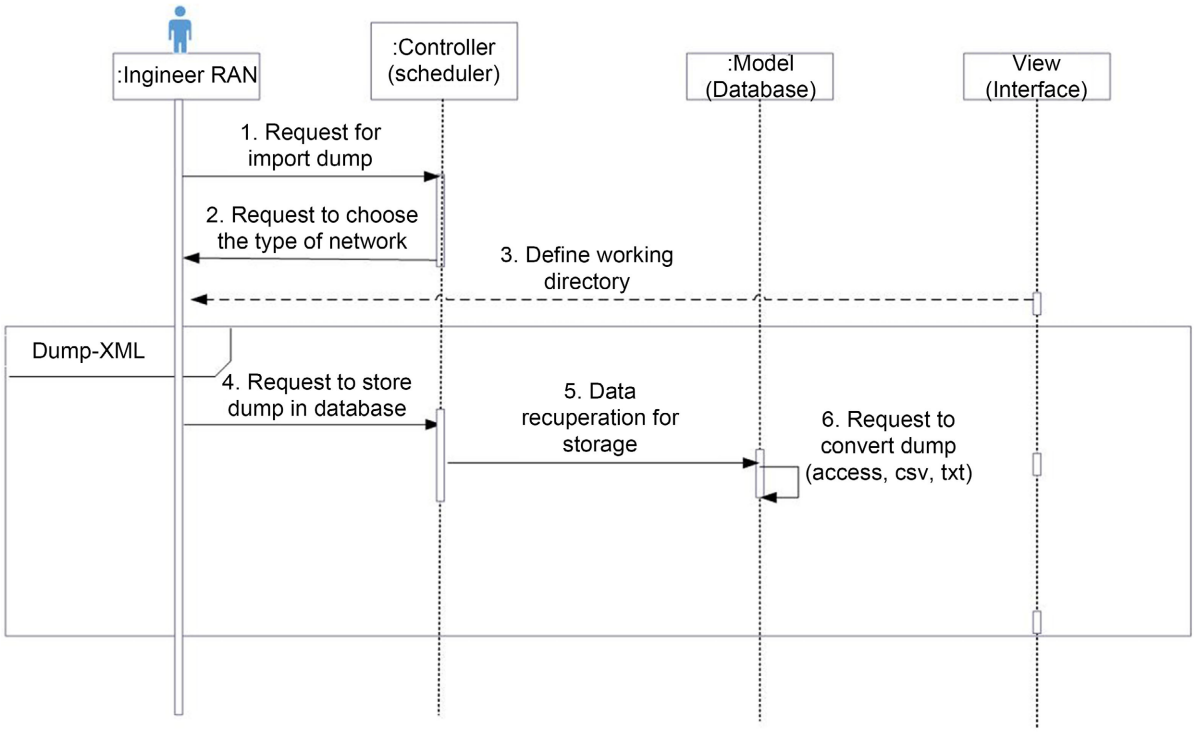

Figure 14. Dump import sequence diagram.

the different operations carried out by an actor, the messages exchanged between the lifelines and this in chronological order.

- Authentification diagram
- **Dump import diagram**

4.5.4. Class Diagram

Our class diagram as shown in **Figure 15** consists of 4 main classes as follows:

- **Ran Engineer class:** this class allows the management of the authentication of the engineers who would like to import the dump, update and check the history of the modifications made in the dump
- **Dump class:** this class allows to organize dumps files by category of networks
- **History class:** this class has the history of all the backups made in the application, it is from this that we can export the dump files to update the ATOLL database
- **Old_Dump class:** This class owns the files for recovering dumps at a later date.

5. Results and Comments

5.1. Application Tree

The architecture of our application has several modules which are presented as shown in **Figure 16** below.

The tool is made up of:

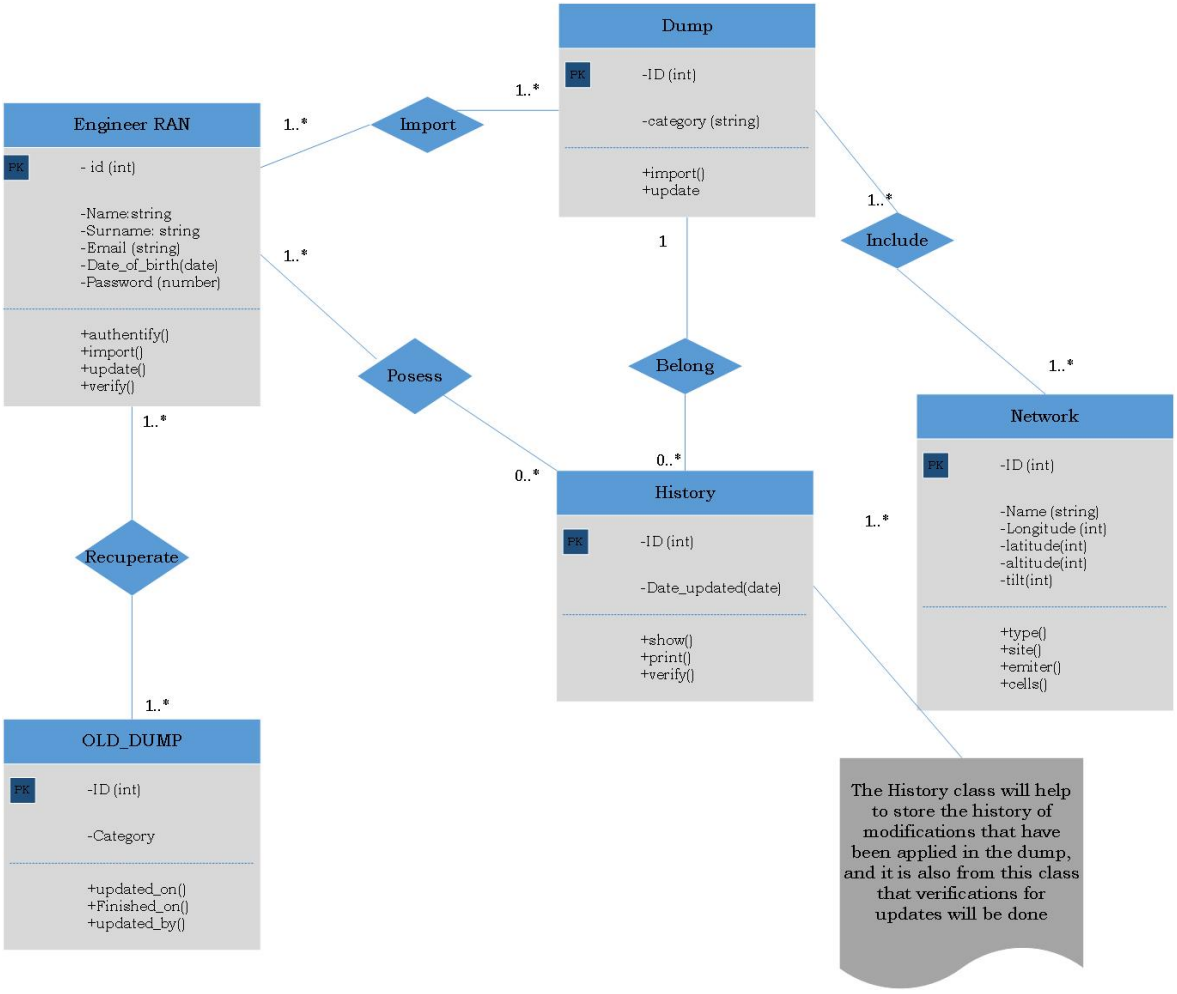

Figure 15. System class diagram.

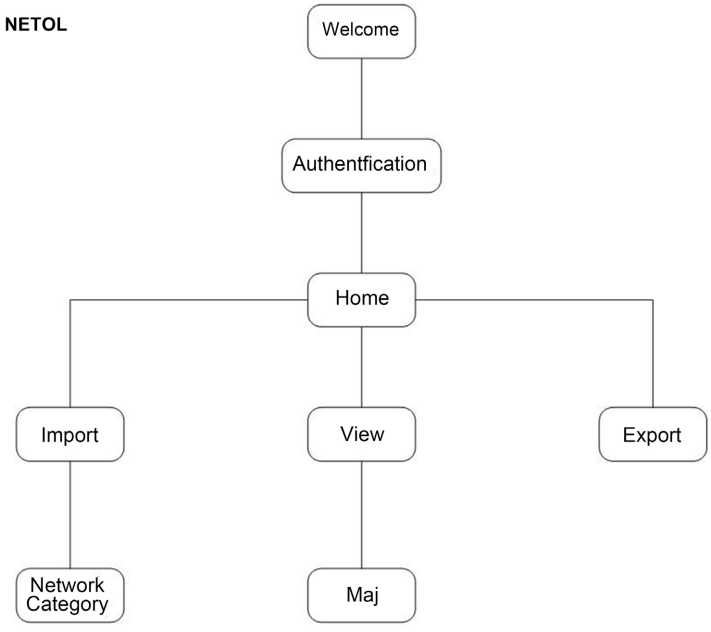

Figure 16. Tree structure of the update tool, NETOL.

- **The welcome page:** the first page when the application starts;
- **The authentication page:** for creating accounts, identifying and logging out;
- **The home page:** gives the possibility to the user to make a choice between 04 menus: import the Dump; View files; Export files and Log files.
- **The import dump page** gives the possibility to choose for which categories of networks one should import a dump (2G; 3G; 4G);
- **The view files page** which presents all the radio parameters that we have imported from the OSS NetAct. We visualize these following Templates from the ATOLL database. In this page we make the updates of the parameters;
- **The export files page** which retrieves update data by network category and loads it into the ATOLL database.
- **The log file page** which gives the possibility to retrieve all activities; errors that occurred in the application during a manipulation.

5.2. Presentation of the Application

The first window of our application is the welcome page in our application which describes what the application does. Then, the login page which is used to authenticate anyone wishing to connect. The application is a tool for automating the updates of the radio parameters of the 2G, 3G and 4G networks of ORANGE CAMEROUN. Thus only radio engineers from the RAN Planning department will have access rights.

➢ **Home page: WELCOME**

After the authentication phase, we have access to the home page. This page has several modules that meet the objectives set. Each time, it counts the number of use cases of our solution. Compound:

- An import tab that imports raw files in .xml format (this file contains the network radio parameters for 2G, 3G and 4G technologies), then parses them in .xlsx format.
- A view tab that displays the radio parameters already parsed to the Template from the ATOLL database. On this interface, you have the possibility to make a manual readjustment, to check the accuracy of all the information contained and to make automatic updates of these parameters.
- An export tab which has all the up-to-date network files to be inserted into the database. These are grouped by 2G, 3G and 4G technologies.
- A log files tab: which contains all the activities and errors of the application during the day.
- A language module: which allows you to translate into the desired language.
- Import page.

In this page, we import the dumps in .xml, .zip, .rar format, deposited in an FTP server from the OSS NetAct. Then, from a python script implemented for this purpose, we parse these files and store them in the database (MySQL) of the application (the parsed data is generated in .xlsx format).

Illustration of this page is shown in **Figure 17**.

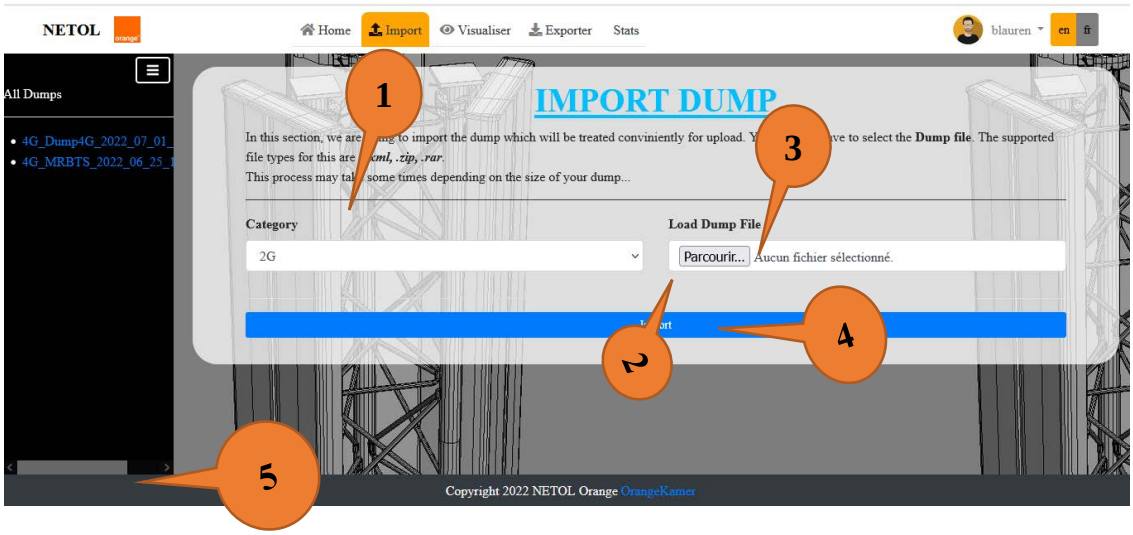

Figure 17. Import page.

On this figure:

- 1 shows the choice of network type: 2G, 3G, 4G.
- 2 describes the category of dump imports by type of technology and displays it by date.
- 3 load the dump according to **Figure 18** on one of the formats: .xml, .zip, .rar. Here we browse to find where the file is, when we do this another window opens.
- 4 here by clicking on "import" we retrieve the browsed file and parse it in .xlsx format.
- 5 here we have a left right scroll bar.

By clicking on "import" the extraction process will start.

An illustration of the extraction in **Figure 19** is as follows:

➢ **View page**

After the import and the parsing of the radio parameters, by the ETL method (Extraction, Transformation and Loading), we recover the essential radio parameters to update the ATOLL database and we load in this interface for verification, readjustment and update. up to date.

An illustration of the features of this page in **Figures 20-23**:

➢ **1:** lists all dumps that are imported by the most recent date.

➢ **1**: we visualize the radio parameters of the sites for the 4G network

Figure 22: Visualization of site parameters

Here we see parameters such as:

- Site name
- Longitude
- Latitude
- site code

➢ **2**:the radio parameters of the transmitters are displayed

➢ **3**: the radio parameters of the cells are displayed

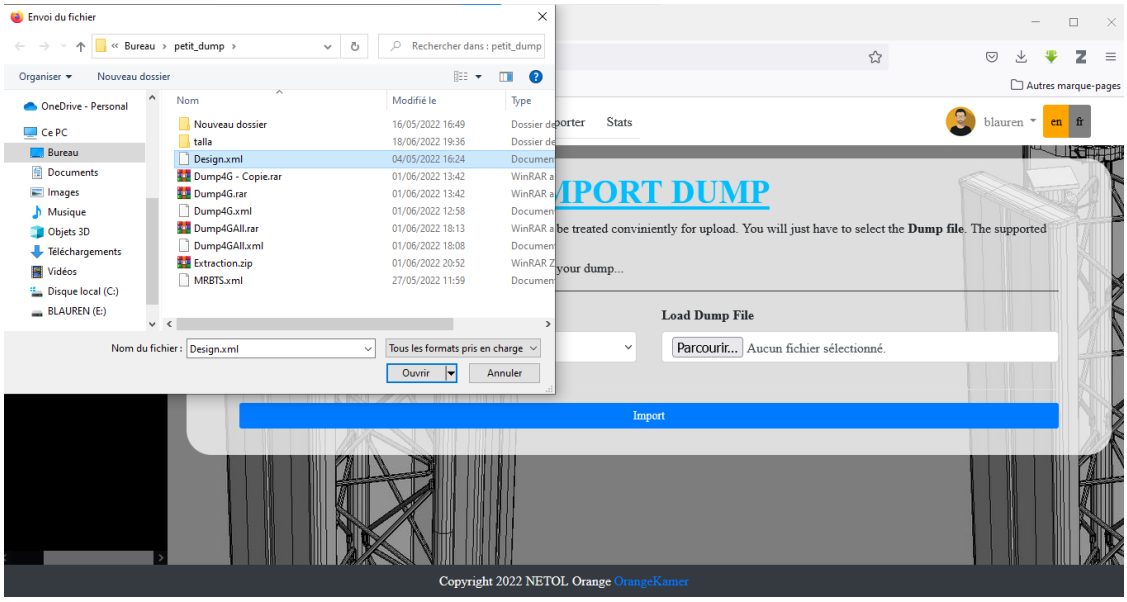

Figure 18. Import page, browsing of the file to import: MRBTS.xml.

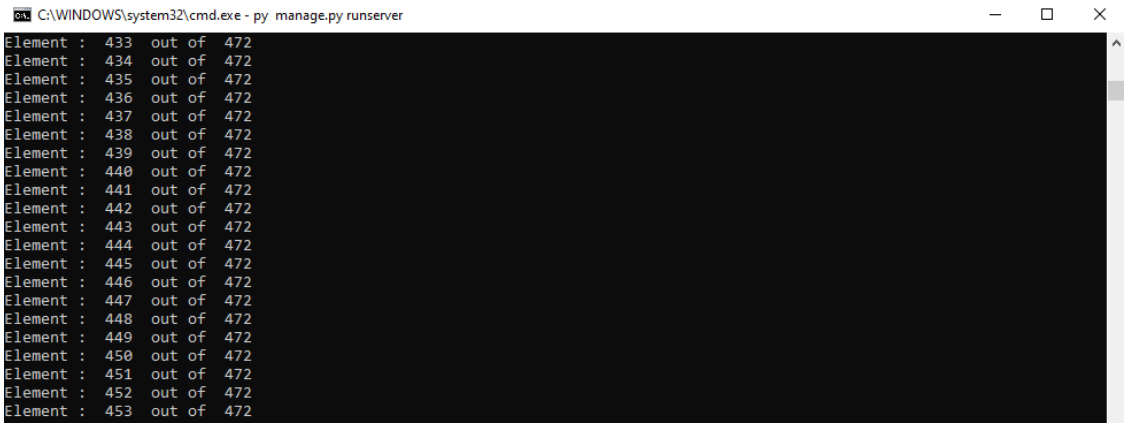

Figure 19. File parsing process being processed.

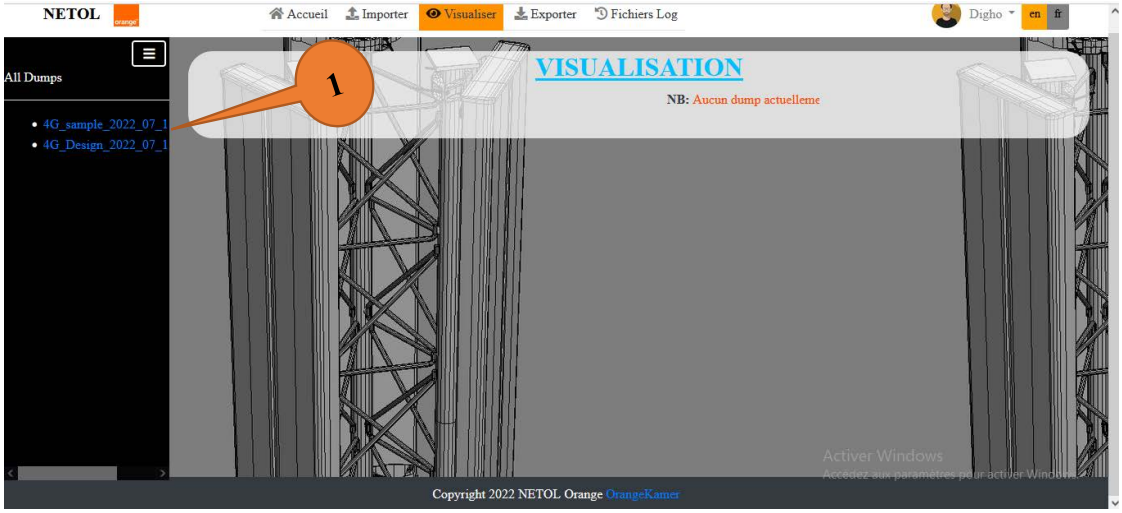

Figure 20. Page for viewing Dump 4G radio parameters.

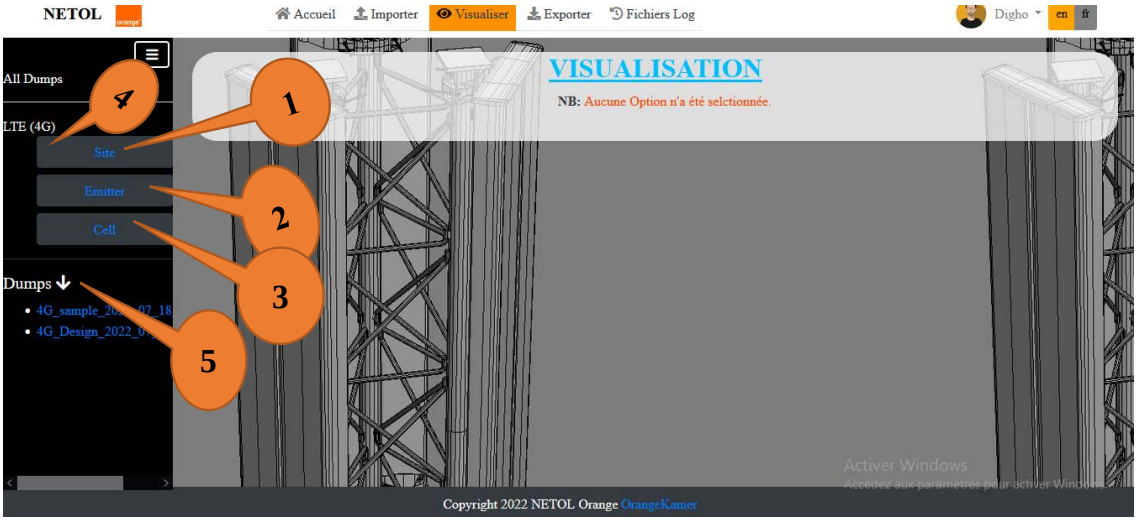

Figure 21. 4G network settings visualization page.

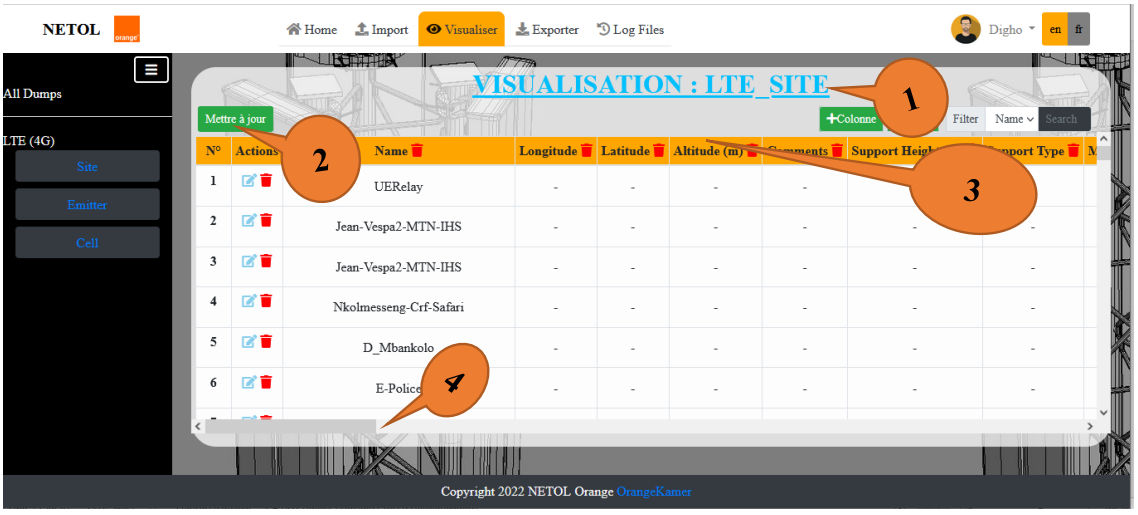

Figure 22. Visualization of site parameters.

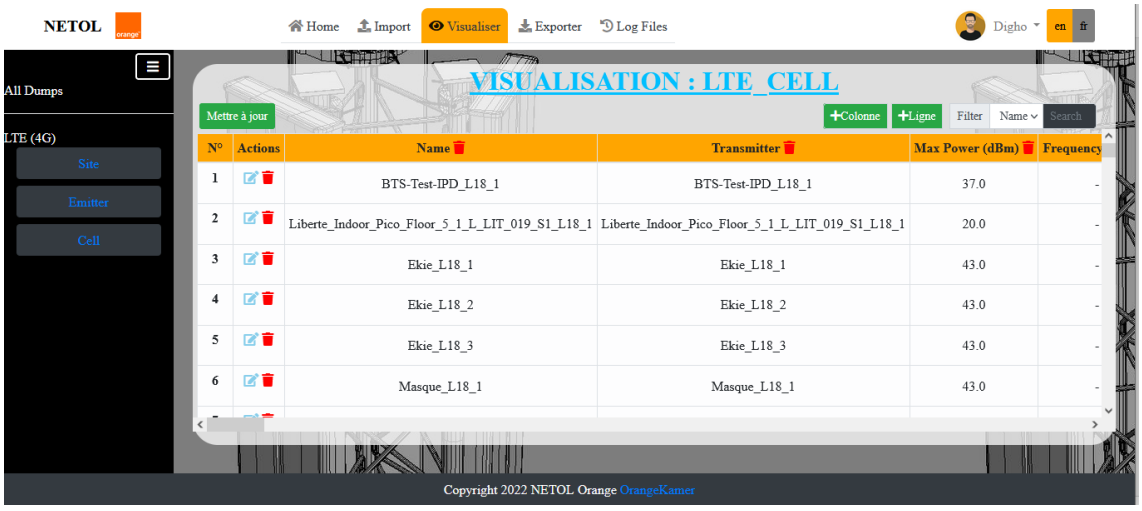

Figure 23. Visualization of cell radio parameters for a 4G network.

When importing and parsing Dumps files, our application automatically updates said radio parameters as can be seen in Figure 24.

Following a bad manipulation on the part of the engineer, it is possible to recover the files in the initial state. See Figure 25.

- **4:** network technology to visualize
- **5:** list of dumps already imported.

Exportation page

After updating the radio parameters, it is necessary to store this data in the ATOLL database (database containing all the network parameters) by exporting them to this page. The updated files are retrieved by type of technology as shown in Figure 26 and Figure 27.

Log page

This module gives us the traces of all the activities that take place in the application. This is a file that is generated each time a user logs in.

An illustration of this page is given in Figure 28.

```
Element :  466  out of  472
Element :  467  out of  472
Element :  468  out of  472
Element :  469  out of  472
Element :  470  out of  472
Element :  471  out of  472
Element :  472  out of  472
←[91mValueError: ←[0m (LTE_SITE) Worksheet named 'LNCEL' not found
←[91mValueError: ←[0m (LTE_CELL) Worksheet named 'LNCEL' not found
←[91mValueError: ←[0m (LTE_CELL) Worksheet named 'AMLEPR' not found
←[91mValueError: ←[0m (LTE_EMITTER) Worksheet named 'LNCEL' not found
←[91mValueError: ←[0m (LTE_EMITTER) Worksheet named 'LNBTS' not found
←[91mValueError: ←[0m (LTE_EMITTER) Worksheet named 'INVUNIT' not found


============================================================
All the updates where operated !!!
============================================================
```

Figure 24. Auto update radio settings.

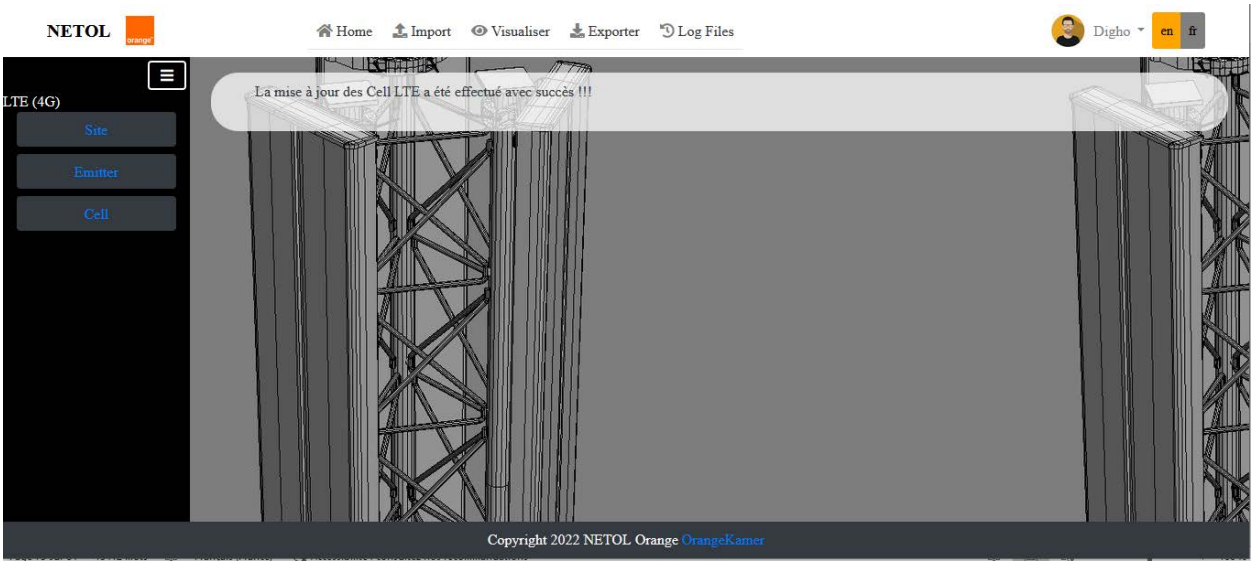

Figure 25. Restore cell data, 4G.

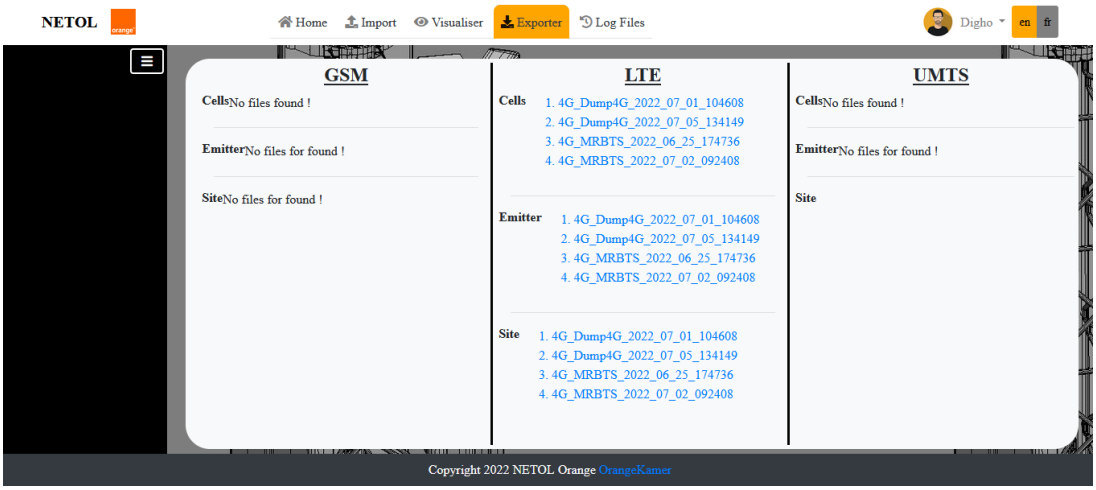

Figure 26. Update files export page.

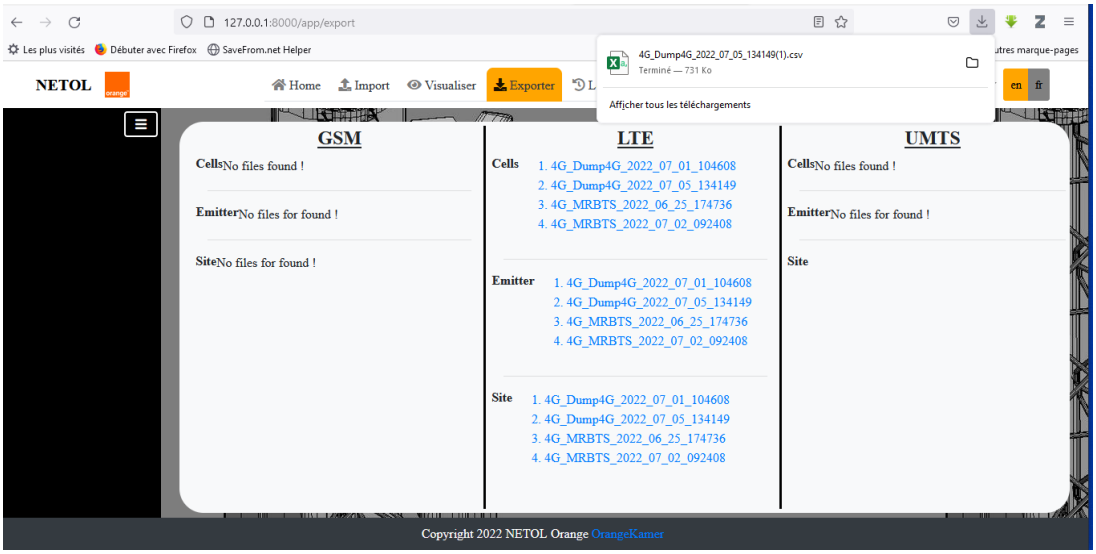

Figure 27. Export page, exported cell file.

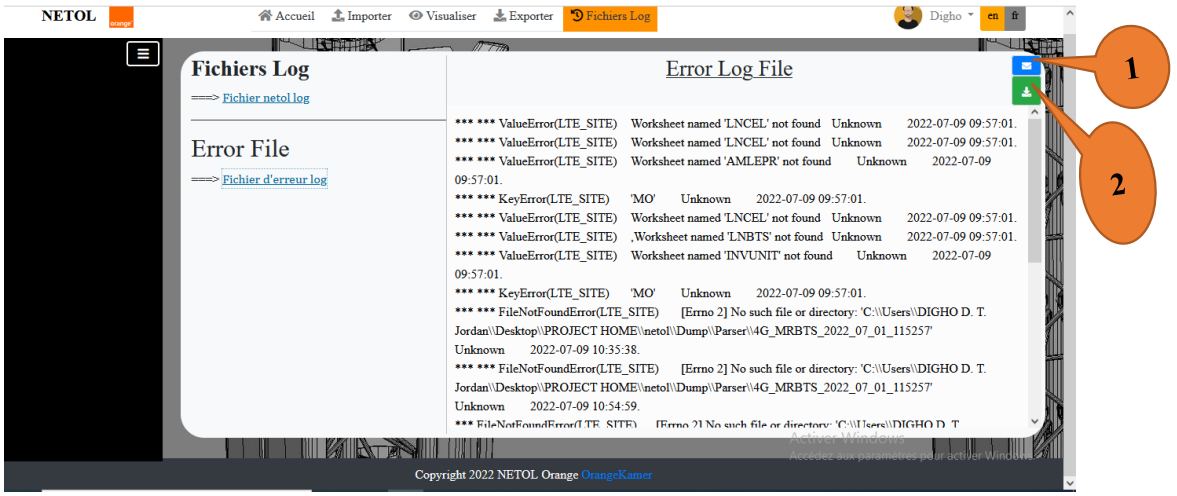

Figure 28. Log files page.

- **1:** here, we retrieve the log files by email.
- **2**: Here, we recover the history of log files by downloading in a txt file.

6. Conclusion and Perspectives

In this work, it was a question for us of optimizing the processing times for the update of the radio parameters of the ATOLL database used to ensure the planning of the radio coverage of the mobile networks. Updating these parameters in real time has a direct impact on increasing the performance of mobile networks in terms of availability, quality of service and network maintenance.

Parsing and processing methods of automatic updates of the files based on the concepts that describe the tools ETL (Extraction, Transformations and Loading) were used to solve this problem.

Finally, we presented the results for a use case of a 4G network. We have an xml file that the application imports from the remote OSS NetAct server, parses it, transforms it and loads it, views it and updates its files in the database and gives us a file update that we must insert in ATOLL in order to make said database up to date.

It appears that our solution (Web application, implementing an algorithm that describes the ETL concept) satisfies the objectives that we set ourselves at the start, thus solving the problem of manually updating the ATOLL database.

Currently our solution:

- Automatically updates the ATOLL database containing network parameters;
- Returns on the working time of RAN engineers to OCM;
- Avoids possible errors that may occur during network performance analyses.

Our solution can be improved, so there are the suggestions from the following perspectives:

- Optimize this task to make the application even more flexible in terms of working time for file parsing;
- Making our solution directly interact with the NetAct OSS to retrieve files;
- Making our solution directly interact with Atoll to directly insert updated files;

In short, remotely manage all types of RAN team servers.

Conflicts of Interest

The authors declare no conflicts of interest regarding the publication of this paper.

References

[1] Davis, J.R. (1999) Datalinks: Managing External Data with db2 Universal Databases. IBM Corporation.

[2] https://www.axysweb.com/processus-etl-talend/

[3] IAAS, PAAS, SAAS, DAAS, BAAS: QUEL MODÈLE CLOUD CHOISIR POUR VOS SOLUTIONS DE PERFORMANCE INDUSTRIELLE? ATYS. Consulted on 24 June 2022.

https://www.atys-concept.com/blog-de-la-performance/articles-performance-industrielle/iaas-paas-saas-modele-cloud-choisir-solutions-de-performance-industrielle/

[4] ATANGANA NGA Aristide (2016) Automation of the Capacity Management of the RAN Subsystem of the Orange Cameroon Network. End-of-Study Dissertation for a Telecommunications Engineering Design Engineer, National Advanced School of Engineering of Yaoundé.

[5] NYA YONKEU BEN Boris (2018) Design of a Support Tool for the Automatic Management of the Quality of Service of UMTS/4G LTE Radio Access Networks: Case of CAMTEL. End-of-Study Dissertation for a Telecommunications Engineering Design Engineer, National Advanced School of Engineering of Yaoundé.

[6] GUILLAUMME CHARPENTIER. *Analyseurs Syntaxiques xml*: *parseur xml,* le 27/02/2004, Consulted on 24 April 2023. http://www-igm.univ-mlv.fr/~dr/XPOSE2003/xml/contenu_parser.htm

[7] https://expleo.com/global/fr/perspectives/blog/automatisation-des-processus-en-ingenierie-les-humains-augmentes-par-une-main-doeuvre-numerique/

[8] https://www.lebigdata.fr/business-intelligence-definition/

[9] https://www.talend.com/fr/resources/data-lake-vs-data-warehouse/

[10] https://www.talend.com/fr/resources/elt-vs-etl/

[11] https://fre.myservername.com/what-is-etl-extract/

[12] Nadège Laure Bemehemie (2022) Design and Realization of a Tool for Automating Updates of the Radio Parameters of the Atoll Base from Oss Netact. End-of-Study Dissertation with a View to Obtaining the Design Engineer Diploma in Telecommunications Engineering at ENSPY, UYI.

[13] Mohamed Motaouakal (2017) Towards the Automation of the Updating of Spatio-Temporal Databases for Aid to Navigation: Case of a Database for the Navigation of People with Reduced Mobility. Masters in Geomatics, University of Laval, Québec.