POLITECNICO DI TORINO

Master Degree course in Computer Engineering

Master Degree Thesis

Design and development of an application to update the Fixed Decoder parameter

Presented by

Wamba Foukmeniok Briand

s2510954

Under the supervision of

Prof. Maurizio REBAUDENGO

Prof. Rémi SHARROCK

INDEX

Acknowledgement

I would like to thank all the people who contributed to the success of my internship. I would particularly like to thank:

- Aurélien FLEGEAU, Head of the development department of the *Mediation Projects Directorate*, for welcoming me into the team and for following up the progress of the internship.

- Marie-Thérèse CONSTANTIN, Tutor of the internship, for her indications in the analysis of the project and for her support throughout my internship.

- I also thank all members of the team's Mediation *Projects Department* for receiving and sharing daily information and practices.

- Thank you also to the teaching and administrative staff of Télécom Paris, especially Professor Rémi SHARROCK for his internship follow-up.

- Thank you also to the teaching and administrative staff politecnico di torino, specially professor Maurizio REBAUDENGO for his internship follow-up.

1 Introduction

As part of my fifth year of study program in computer engineering at Télécom Paris, course SOFTWARE, I am currently working on an internship in the Orange company within the *Mediation Projects Department*, for the design and development of an application that updates parameters of the Fixed-Decoder using new technologies such as Angular, mySQL, Spring MVC.

The Fixed-Decoder constitutes a sensitive link at the level of the transmission of legal data, for the decoding of certain communication information. The Landline Communication Details are in a very specific format. They are encapsulated data that must be decoded in order to be read by legal services.

The Fixed-Decoder makes it possible to translate these Communication Details into a fully configurable format.

The objective of my internship is therefore to develop an application allowing the user to update the configuration data by following a succession of steps.

The development of the application started after the analysis and migration from old to new databases. Technical documentation has been produced for the installation and handling of this application.

2 Presentation of the host organization

2.1 Orange

Formerly France Telecom, Orange became the group's unique brand for internet, television and telephony. The company has internationalized its presence among the general public in several countries in Europe and Africa as well as among companies in 220 countries and territories.

Orange becomes one of the main telecommunications players with 252 million customers including 3 million fiber customers worldwide and nearly 153,000 employees, and one of the world leaders in business telecommunications services with the brand Orange Business Services.

More recently Orange launched into the banking sector with its mobile bank called Orange Bank, with the aim of reaching two million customers in ten years. It also launched into new services such as "The protected house" and "The house connected ".

2.2 The Information Systems Department of orange

The Orange France Information Systems Department (DSI) manages and develops a portfolio of more than 3,400 applications. Which are used to build customer knowledge and support business processes such as ordering, delivery, service activation and billing.

The main mission of the IT department is to design, develop and integrate orange information systems in order to better meet the expectations of business needs and better satisfy our customers.

2.3 The Mediation Department Orange France

This department is in charge of studies, development and production support for equipment, allowing the collection of data from all networks and mobile equipment in order to route it to Information System's (IS) applications, in particular billing applications and legal services.

The different network equipment provides data in multitudes of formats using very specific protocols, therefore it is impossible to make them communicate with IS applications. It is therefore necessary to have between the two equipment a software capable of collecting and formatting the data so that it can be interpreted by the IS.

This figure shows the plan for the next 5years:

figure 1: Orange's plan for the next 5years

3 Project Context

The internship takes place at its own company premises in Orange village, Arcueil site. The company provided me with a Windows 10 desktop machine.

The tutoring is done by Mrs. Marie-Thérèse CONSTANTIN Software Specialist within the orange mediation department, who has great knowledge on the principles of perimeter-mediation and who works on the Decoders project.

She provided me with explanations on the existings, how to manually update the parameters of the Decoder, as well as instructions for the improvements to be made on the application. She equally took care of the validation of the different phases of development that I realized.

Telecommunication companies have mandatory and legal roles to **collect and store** the users' communication details. Communication details represent the information of an SMS, or a call between two users. Before storing the communication details in the database, they are encrypted to respect the RGPD and security policies. Sometimes these communication details are used for marketing or by the STATE POLICE. In these cases, there is the necessity to decrypt the communication details before giving them to the STATE POLICE or to the end users. Before storing the communication details in the database, they are encrypted to respect the RGPD and security policies. The application in charge of collecting and storing details of communication at Orange is COMETE. Sometimes these communication details are used for marketing or by the STATE POLICE.

The interaction between different actors in the process of collection and decodification of communication details is as follows:

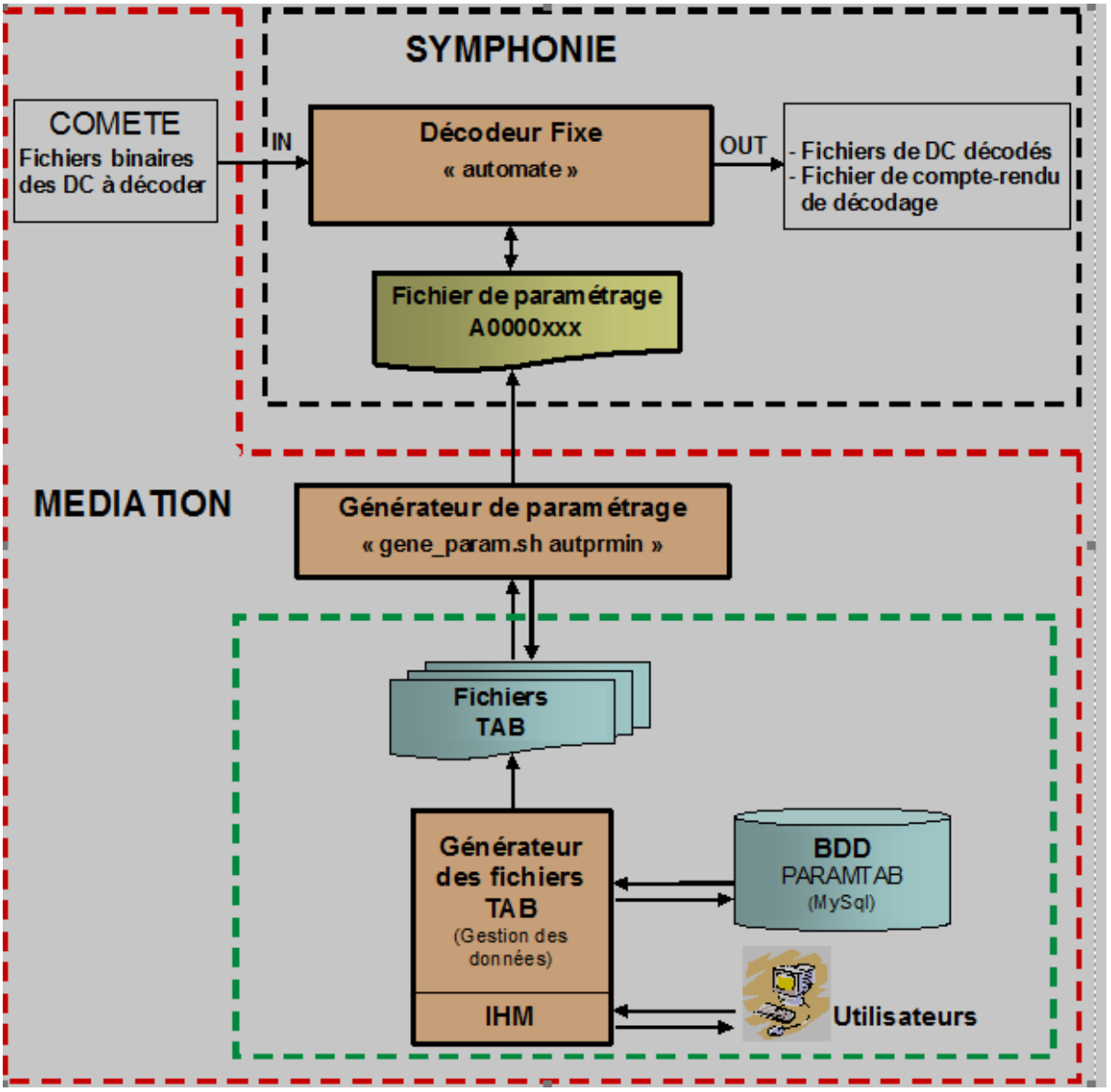


figure 2: functional architecture

COMETE which is in charge of collecting Details of Communication (DC) in different formats and forwards them to SYMPHONIE

SYMPHONIE which includes the FIXED-DECODER.

GENERATEUR DE PARAMETRAGE which provides parameters to SYMPHONIE to decode Details of Communication.

When The state or a team perform a request:

Need or Request is a list, or a set of information of a detail of communication asked by a team or the state police. Examples of need could be:

- The duration of the communication detail (call).
- The content type (video, picture, text) of a communication detail (SMS).
- Call history of a given phone number.
- The localization of the caller or the callee.

Needs / Requests are managed based on the architecture depicted in figure 3: The owner of the need/request contacts COMETE TEAM and COMETE TEAM provides the encrypted communication details to SYMPHONIE TEAM. Once SYMPHONIE TEAM receives the encrypted communication details, it requests the tab files. The tab files received by SYMPHONIE TEAM are then used as keys to decrypt the communication details using the FIXED-DECODER application. The decrypted communication details are sent to the requester.

4 Presentation of the existing project

The **FIXED-DECODER** is the application which decodes details of communication. To decode the details of communication it receives as inputs encrypted details of communication and tab file. The Tab files play the role of keys in the decodification process. Details of communication come from the application COMETE and the tab files come from the execution of a script called GENERATOR OF TAB FILE. To generate the tab file we have to provide some input data to the generator of the tab file, this input data set is called: donnée d'enregistrement de sortie (registration of exit data), donnée de traitement (treatment data), donnée d'entrée (input data).

Currently the configuration of the data streams for the generator of tab file is done manually under the Windows operating system with the VBA and Macros Excel programming languages.

Once we receive a request to generate a tab file, we also receive the specifications. Based on these specifications we will generate tab files following these steps:

- Before any operation we launch a script with the goal of creating a backup of the database.
- We take the template, which is an excel file with 24 sheets. Each sheet represents a table in the database. In the template there are some macro VBAs having the role to copy the value of some cells into others cells and applying specific rules. This step is very complex because it is difficult to discover if there is an error or not.
- Once the template is filled, we launch a first script. This script has a role to read the template file, extract data and store them into the database. This script can detect some errors on data quality, for example it checks if values that should be in a given range effectively belong to that range.
- Once the database is up to date, we launch the script to generate the tab files.

The activity diagram for the entire generator of tab file is as follows:

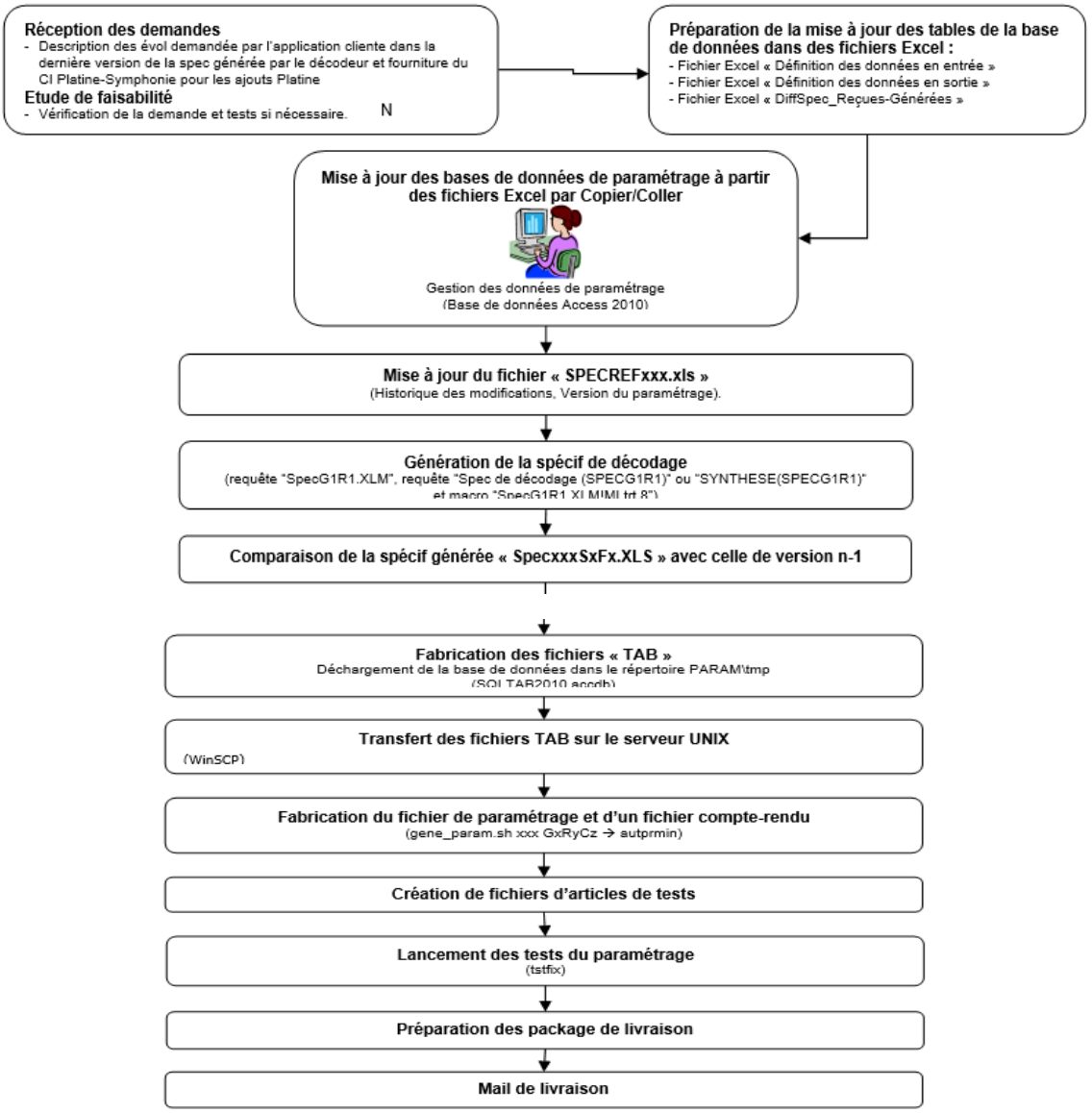
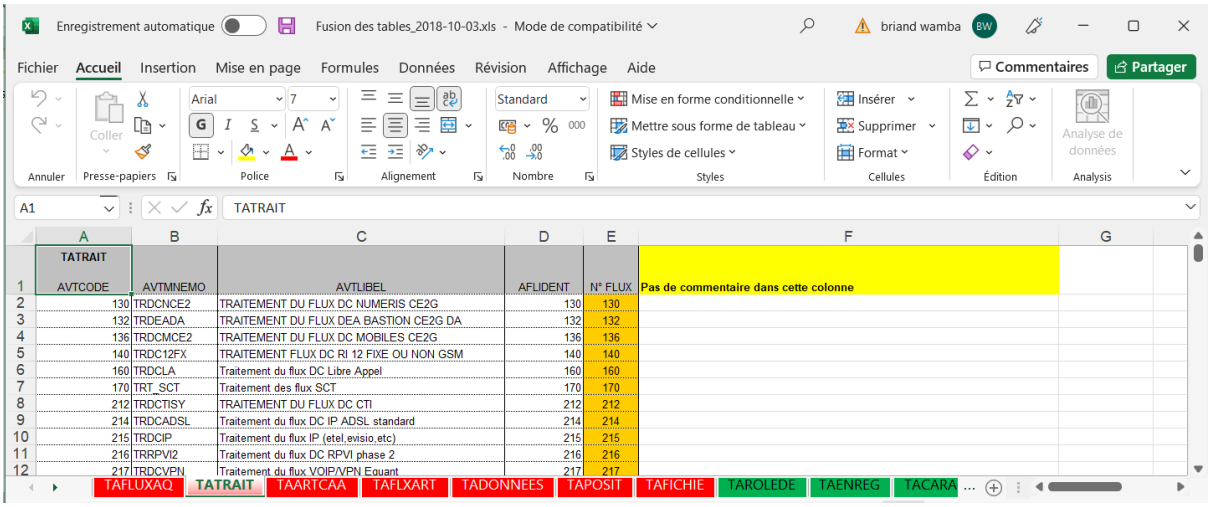


figure 2: activity diagram of a generator of tab file

The figure 3 shows the template excel with some data.

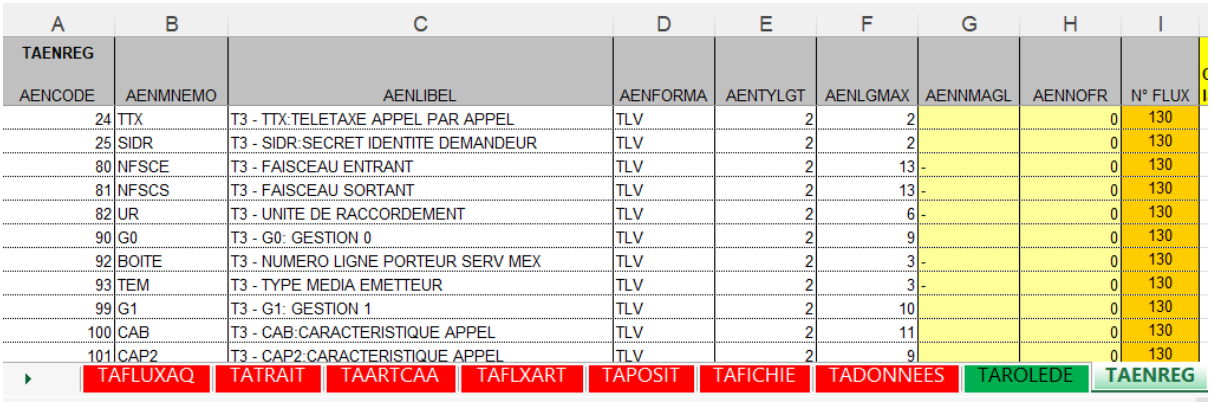
Here we have the representation of the treatment table in the excel sheet. The treatment represents the operations we are going to apply for communication details coming from a given flux. The cell **N°FLUX** in the sheet represents the version of the machine who sends the details of communication to the application COMETE. The cell **AVTLIBEL** is the label to describe the type of treatment. The cell **AVTCODE** is the identifier of the treatment.



TATRAIT

AVTCODE	AVTMNEMO	AVTLIBEL	AFLIDENT	N° FLUX	Pas de commentaire dans cette colonne
130	TRDCNCE2	TRAITEMENT DU FLUX DC NUMERIS CE2G	130	130	
132	TRDEADA	TRAITEMENT DU FLUX DEA BASTION CE2G DA	132	132	
136	TRDCMCE2	TRAITEMENT DU FLUX DC MOBILES CE2G	136	136	
140	TRDC12FX	TRAITEMENT FLUX DC RI 12 FIXE OU NON GSM	140	140	
160	TRDCLA	Traitement du flux DC Libre Appel	160	160	
170	TRT_SCT	Traitement des flux SCT	170	170	
212	TRDCTISY	TRAITEMENT DU FLUX DC CTI	212	212	
214	TRDCADSL	Traitement du flux DC IP ADSL standard	214	214	
215	TRDCIP	Traitement du flux IP (etel,evisio,etc)	215	215	
216	TRRPVI2	Traitement du flux DC RPVI phase 2	216	216	
217	TRDCVPN	Traitement du flux VOIP/VPN Equant	217	217	

TAFLUXAQ | TATRAIT | TAARTCAA | TAFLXART | TADONNEES | TAPOSIT | TAFICHIE | TAROLEDE | TAENREG | TACARA



TAENREG

AENCODE	AENMNEMO	AENLIBEL	AENFORMA	AENTYLGT	AENLGMAX	AENNMAGL	AENNOFR	N° FLUX
24	TTX	T3 - TTX:TELETAXE APPEL PAR APPEL	TLV	2	2		0	130
25	SIDR	T3 - SIDR:SECRET IDENTITE DEMANDEUR	TLV	2	2		0	130
80	NFSCE	T3 - FAISCEAU ENTRANT	TLV	2	13	-	0	130
81	NFSCS	T3 - FAISCEAU SORTANT	TLV	2	13	-	0	130
82	UR	T3 - UNITE DE RACCORDEMENT	TLV	2	6	-	0	130
90	G0	T3 - G0: GESTION 0	TLV	2	9		0	130
92	BOITE	T3 - NUMERO LIGNE PORTEUR SERV MEX	TLV	2	3	-	0	130
93	TEM	T3 - TYPE MEDIA EMETTEUR	TLV	2	3	-	0	130
99	G1	T3 - G1: GESTION 1	TLV	2	10		0	130
100	CAB	T3 - CAB:CARACTERISTIQUE APPEL	TLV	2	11		0	130
101	CAP2	T3 - CAP2:CARACTERISTIQUE APPEL	TLV	2	9		0	130

TAFLUXAQ | TATRAIT | TAARTCAA | TAFLXART | TAPOSIT | TAFICHIE | TADONNEES | TAROLEDE | TAENREG

figure 3: Example of template having 24 sheets.

5 Problem Objectives and solution

The following applications: COMETE, FIXED-DECODER, Generator of tab file are part of the most important applications of the company. They grant legal conformity to state laws to the company, because telecommunication companies have the role of collecting and storing details of communications.

The generator of tab file is one of the oldest applications still in production and it is built with deprecated technologies. Orange decided to create a new version of the application for the following reasons:

- Answer to new business requests.
- Maintainability: There are few employees with knowledge of the old technology.
- Scalability issue: the application runs only on Windows server. It's not compatible with Linux or mac os.
- Knowledge transfer: The time for a new developer to be independent on how it works was 1 year on average.
- The user experience: the time to answer a request was very high because there is no good interface but only excel files.

To overcome the aforementioned issues, the following objectives were laid down within the Orange group:

- The collection of voice communication: the application in production was not entirely able to collect, store and use voice communication details. This is due to the fact that it could only extract detail of the caller, callee and duration of the call but not details of the voice message exchanged during the conversation
- Improve the software performance (multithreading) and decrease the cost of the hardware.
- Be up to date with the software version

To achieve the goal, we decided to create new versions of many applications, amongst which the generator of the tab files. The technologies used to develop the new versions are very recent and popular. These new technologies stick to the objectives of the company. The advantages of the new version of the generator of tab files include:

- Have a friendly user interface.
- Accessibility, the new version is a web application, and is therefore accessible from any browser, in contrast to the old version which is a desktop application.
- To speed up the data update process, the new version autocompletes many fields.
- Facilitate the accessibility of the application, putting it on a server accessible from any computer connected to the internet.
- Make updating of data possible for people with minimal knowledge of the subject. Now there are help messages that guide the user. The presence of the forms in the new version allows you to ask for the minimum number of mandatory data with clear and understandable labels.

  The order in which the forms are presented is now consistent with a certain logic. For example, in a form if we want to take the information of a person it will be more logical to ask for the following information : surname, firstname, age, place of birth, address in the following order:

  1 - surname

  2 - firstname

  3 - age

  4 - place of birth

  5 - adress

  and NOT in the following order:

  1 - place of birth

  2 - adress

3 - surname

4 - age

5 - firstname

The new application guides those who fill the information by asking for the information in a sequential and logical order.

To make evolutions easier we decided to use the most modern technologies. Now it’s easy to find people who can implement new features, all this is possible because we make use of the most popular frameworks of the moment such as SPRING BOOT and ANGULAR. Spring Boot is based on the high-level JAVA language while Angular is based on JavaScript.

Throughout my internship i had to achieve the following goals:

- Migrate and merge 13 existing DataBases.
- Create GUIs for the management of different types of Communication Details.
- Generate configuration tables respecting a precise format required by the Decoder.
- Save and restore the database.
- Creation an GUI with the following functionalities:
    - Data stream Management.
    - Management of Articles.
    - Database Data Management.
    - Making a set of configuration files.
    - Database Backup and Restore.
    - Information menu.

I built and delivered in production the new version of GENERATOR OF TAB FILE.

6 Working Method

The definition of Scrum is more of a framework than a complete project management method. It nevertheless allows the management of the development of complex applications. The project is organized around development "sprints" (iterations) usually lasting two to four weeks. A typical example of using Scrum is an IT development project where the client has not yet defined all the functionalities he needs and where some organizational flexibility is therefore required.

The project begins with "sprint 0", dedicated to carrying out all the preparation and implementation work: design and architecture, development environments, monitoring and integration tools ...

The requested functionalities are listed and described in the form of “user stories” and placed in the product backlog. At the start of a sprint, the team is brought together (developers and client) to determine which user stories will be developed. They are encrypted during a meeting entitled "poker planning" and prioritized. All of the selected user stories constitute the "sprint backlog". The sprint planning and the objectives to be achieved are then set. For messaging software, an example of product backlog could contain the user stories "Connect to the messaging server", "consult the list of messages" and "create a message”.

7 Overview of tools used

Outils de gestion: GIT

So, what is Git in a nutshell? This is an important section to absorb because if you understand what Git is and the fundamentals of how it works, then using Git effectively will probably be much easier for you. As you learn Git, try to clear your mind of the things you may know about other VCSs, such as CVS, Subversion or Perforce . Doing so will help you avoid subtle confusion when using the tool. Even though Git's user interface is fairly similar to other VCSs, Git stores and thinks about information in a very different way, and understanding these differences will help you avoid becoming confused while using it.

Snapshots, Not Differences

The major difference between Git and any other VCS (Subversion and friends included) is the way Git thinks about its data. Conceptually, most other systems store information as a list of file-based changes. These other systems (CVS, Subversion, Perforce, Bazaar, and so on) think of the information they store as a set of files and the changes made to each file over time (this is commonly described as delta-based version control).

Outils de développement

Visual Studio Code is an editor of source code developed by Microsoft for Windows, Linux and MacOS. Includes support for debugging , an integrated Git control, Syntax highlighting, IntelliSense, Snippet and code refactoring . The editor theme, keyboard shortcuts and preferences are customizable. It is free and free software , even if the official version is under a proprietary license .

Eclipse is an integrated development environment (IDE) used in computer programming. It contains a base workspace and an extensible plug-in system for customizing the environment. Eclipse is written mostly in Java and its primary use is for developing Java applications, but it may also be used to develop applications in other programming languages via plug-ins, including Ada, ABAP, C, C++ …

8 Development

Our application has a 3-third architecture, that is: we have 3 main components, which are the Database, the server(back end) and the front-end. We will then describe the role of each of these components as well as their implementation in our specific case.

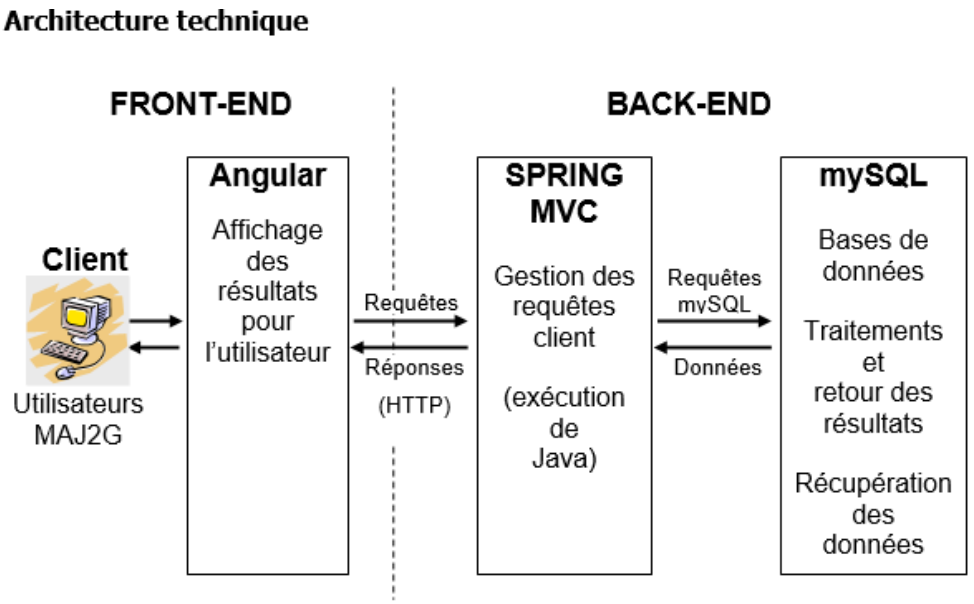


figure 4: Technical architecture.

8.1 DataBase

At the resumption of the project there were 13 databases. For reasons of performance we decided to merge these databases to form a single one. Our application being a single page application, all the data is loaded into the user's browser at his first connection. If we had to open and close the connection 13 times with the databases it would have had a considerable cost in terms of response time.

This merge prevented redundancy because these databases had identical tables and data (reference table). We have for example the table Taformat. The formats used were the same in our 13 databases. A change on this table was repeated 13 times for each database.

After a study on the fixed decoder, we made changes to the databases. Each stream was stored in a database of its own. There was therefore redundancy because the tables common to all streams were duplicated. We had to be careful when carrying on the modification because we had to keep the information consistent in the streams. For example, we will take the TAFORMAT: this table is common to all streams and contains information about the different formats handled by the applications. If we modify this table in the stream 130 (we change the code of the HEXA format: 2 -> 6) and we do not modify it also in the other streams, this will raise the problem of Consistency of the data. It will be difficult to know which data are correct. We therefore merged the databases to have a single copy of the common tables in order to avoid the problem of data consistency.

Problem: We have several databases called parmXXX. XXX is a unique number, representing a stream, (the stream XXX is the device that stores the data in the database parmXXX). All databases contain identical tables. Our goal is to unite them in a single database called paramtab in order to carry out the processes more easily, this is because simultaneous access of several databases is not suitable for the next steps.

Here is the architecture of the two applications that will allow us to merge these databases.

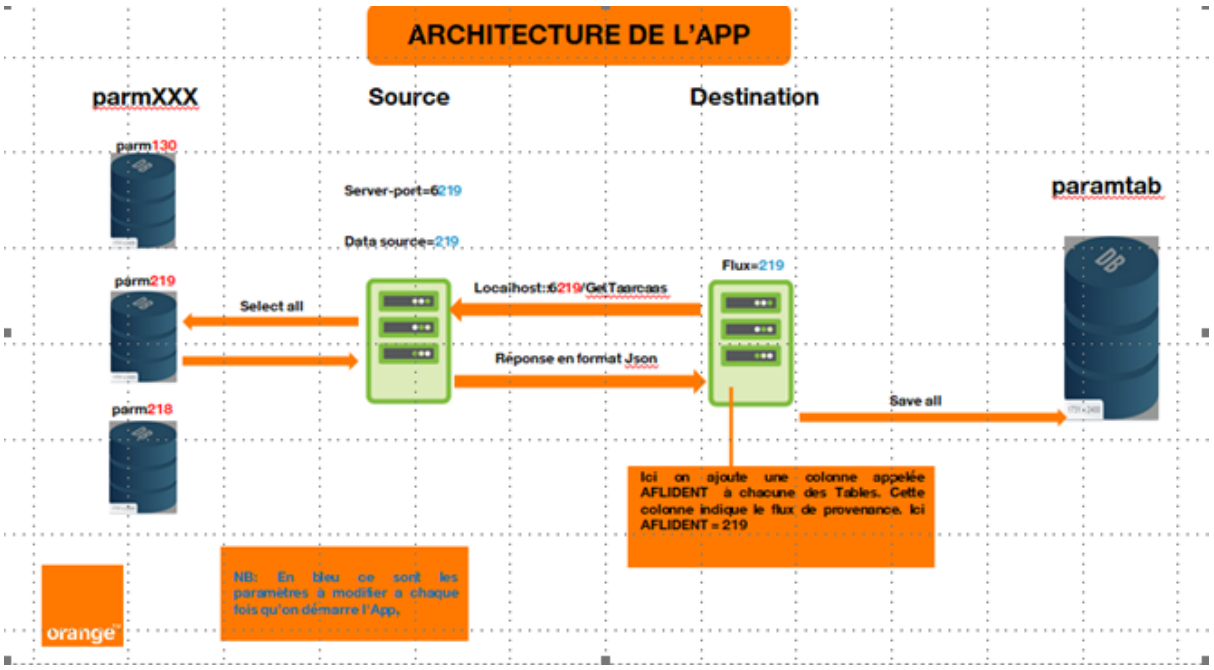


figure 5: architecture of the merged databases

Here is the general idea of the program that merges the tables. The Destination application receives the data from the Source application and stores it in the new database.

Step 1: Open the application.properties file of the Source application. Change the value of server-port (6XXX) and data-source variable (paramXXX). Example: replace XXX with 219.

Step 2: Launch the Source application.

Step 3: Open the Wrapper.java file of the Destination application. In the function public void Start (), modify the value of the stream variable = XXX. For example stream = 219;

Step 4: Launch the destination application. Wait for the data transfer to complete.

Step 5: Turn off applications and resume in step 1 to transfer data from another stream.

8.2 Back End

Spring Boot makes it easy to create stand-alone, production-grade Spring-based Applications that you can run. We take an opinionated view of the Spring platform and third-party libraries, so that you can get started with minimum fuss. Most Spring Boot applications need very little Spring configuration.

You can use Spring Boot to create Java applications that can be started by using java -jar or more *traditional war deployments*. We also provide a command line tool that runs “spring scripts”.

Our primary goals are:

- Provide a radically faster and widely accessible getting-started experience for all Spring development.
- Be opinionated out of the box but get out of the way quickly as requirements start to diverge from the defaults.
- Provide a range of non-functional features that are common to large classes of projects (such as embedded servers, security, metrics, health checks, and externalized configuration).
- Absolutely no code generation and no requirement for XML configuration.

The back end was divided into 4 packages namely:

- Entities
- Repositories
- Services
- Controllers

8.2.1 Entities

The package Entities is in charge of defining the objects of our application. Spring boot uses POJO to map a class into a table in the database. POJO stands for Plain Old Java Object which is an ordinary Java object, not bounded by any special restriction other than those forced by the Java Language Specification and not requiring any class path. POJOs are used for increasing the readability and re-usability of a program. POJOs have gained more acceptance because they are easy to write and understand.

An example of java class contained in the Entity package is represented below:

```
3  import javax.persistence.EmbeddedId;
8
9  @Data
10 @AllArgsConstructor
11 @Entity
12 public class Taartcaa {
13
14     @EmbeddedId
15     private TaartcaaPK id;
16
17     private String aaclibel;
18
19     private String aacmnemo;
20
21     private Long aacoriga;
22
23     private Long amrcode;
24
25     public Taartcaa() {
26         super();
27         this.id = new TaartcaaPK();
28     }
29
30
31 }
```

figure 6

@Data annotation provides the Java Class all getters and setters .

@AllArgsConstructor provides all constructor possible for our class.

@embeddedId annotation specifies the primary key of the table. It is used when we have a composite primary key.

@Entity annotation translates this class into a table in the database. using the properties of this class as columns of the table.

8.2.2 Repositories

This package contains all classes in charge of performing operations on the database.

We use JpaRepository, a particular Interface which provides common methods equivalent to database queries. By Extending JpaRepository interface, we can personalize some queries and use them. With this Interface we do not have to write common queries like delete by id, find by id, select all and many other queries of this nature.

Below is a figure of a java class contained in the Repository package.

```
1  package DecodeurFixe.repository;
2
3  import org.springframework.data.jpa.repository.JpaRepository;
6
7  public interface TaformESRepository extends JpaRepository<TaformES, Long>{
8
9  }
10
```

figure 7: java class

8.2.3 Services

Here the logic and services offered by our application are implemented. Of all packages, this package is the most important and complex one. Spring boot provides @service annotation, enabling this class to be injectable(enabling the use of this object without instantiating it). We therefore just have to annotate its declaration with @autowired.

A good practice in the implementation of these classes is to create an interface containing the methods, and then creating another class which implements them. This allows modulation and workgroup.

Below is a figure of a java class contained in the Service package.

```java
import java.util.List;

public interface TaenregService {
    public ResultatGenericBody generer_fichier_tab_taenregs(long flux, String path);

    public List<Taenreg> getAllTaenregs();

    public ResultatGenericBody backup(String path);

    public ResultatGenericBody restaurer(String path);
}
```

figure 8 : java class

```java
import java.io.BufferedReader;

@Service
public class TaenregServiceImplement  implements TaenregService{


    @Autowired
    private TaenregRepository taenregRepository;



    @Override
    public ResultatGenericBody backup(String path) {
    ResultatGenericBody resultat = new ResultatGenericBody();

        Gson gson = new Gson();

        try {
            path = Wrapper.transforme_path_en_fonction_du_so(path+  "/TAENREG.json");
            OutputStream out_strem = new FileOutputStream(path);
            OutputStreamWriter fos = new OutputStreamWriter(out_strem,Charset.forName("UTF-8"));
```

figure 9: java class

8.2.4 Controllers

This is the entry point for our application requests. When a request arrives and the Dispatcherservlet finds a controller that maps the URL of therequest, the Dispatcherservlet points the request to the controller in question and the controller will call the services of our application to serve the request.

Below is a figure of a java class contained in the Controller package.

```
3  import java.util.List;
15
16 @Controller
17 @CrossOrigin
18 public class TaenregController {
19
20     @Autowired
21     private TaenregService taenregService;
22
23     @RequestMapping( value="/user/GET/taenregs" , method = RequestMethod.GET , produces = MediaType.APPLICATION_JSON_VALUE)
24     public @ResponseBody List<Taenreg> getAll(){
25         return taenregService.getAllTaenregs();
26     }
```

figure 10: java class

8.3 Front End

Angular is a JavaScript framework, which means it provides a number of APIs and structures that help you quickly and easily create complex client-side code. Angular does a great job at providing not only features but also a basic framework and programming model to create client applications. The following sections describe the most important aspects of the Angular framework and how they contribute to make Angular a great JavaScript framework.

8.3.1 Modules

In general, Angular apps use a modular design. Despites the fact that modules are not required, they are highly recommended because they allow you to separate your code into separate files. This helps you keep your code files short and manageable while still allowing you to access the functionality from each one.

Unlike how you use modules with TypeScript, with Angular you import external modules at the top of a file and export the functionality you need at the bottom. You do this by using the key terms import and export, with the following syntax:

Import {Component} from 'angular2/core';

Export class App{}

8.3.2 Directives

Directives are JavaScript classes with metadata that defines the structure and behavior. Directives provide the majority of UI functionality for Angular applications. There are three major types of directives:

Components: A component directive is a directive that incorporates an HTML template with JavaScript functionality to create a self-contained UI element that can

be added to an Angular application as a custom HTML element. Components are likely to be the directives you use the most in Angular.

· **Structural:** You use structural directives when you need to manipulate the DOM. Structural directives allow you to create and destroy elements and components from a view.

Attribute: An attribute directive changes the appearance and behavior of HTML elements by using HTML attributes.

8.3.3 Data Binding

One of the best features of Angular is the built-in *data binding*: the process of linking data from a component with what is displayed in a web page. Angular provides a very clean interface to link model data to elements in a web page.

When data is changed on a web page, the model is updated, and when data is changed in the model, the web page is automatically updated. This way, the model is always the only source for data represented to the user, and the view is just a projection of the model.

8.3.4 Dependency Injection

Dependency injection is a process in which a component defines dependencies on other components. When the code is initialized, the dependent component is made available for access within the component. Angular applications make heavy use of dependency injection.

A common use for dependency injection is consuming services. For example, if you are defining a component that requires access to a web server via HTTP requests, you can inject the HTTP services into the component, and the functionality is available in the component code. In addition, one Angular component consumes the functionality of another via dependency.

8.3.5 Services

Services are the major workhorses in the Angular environment. Services are singleton classes that provide functionality for a web application. For example, a common task of web applications is to perform AJAX requests to a web server. Angular provides an HTTP service that houses all the functionality to access a web server.

The service functionality is completely independent of context or state, so it can be easily consumed from the components of an application. Angular provides a lot of built-in service components for basic uses, such as HTTP requests, logging, parsing, and animation. You can also create your own services and reuse them throughout your code.

For the first connection, all the html code and java scripts are loaded into your browser. Also, necessary data are loaded and stored into your browser. For the rest of the time only new data are sent into the server. We need to code very well by the ways to keep synchronised data between server and client.

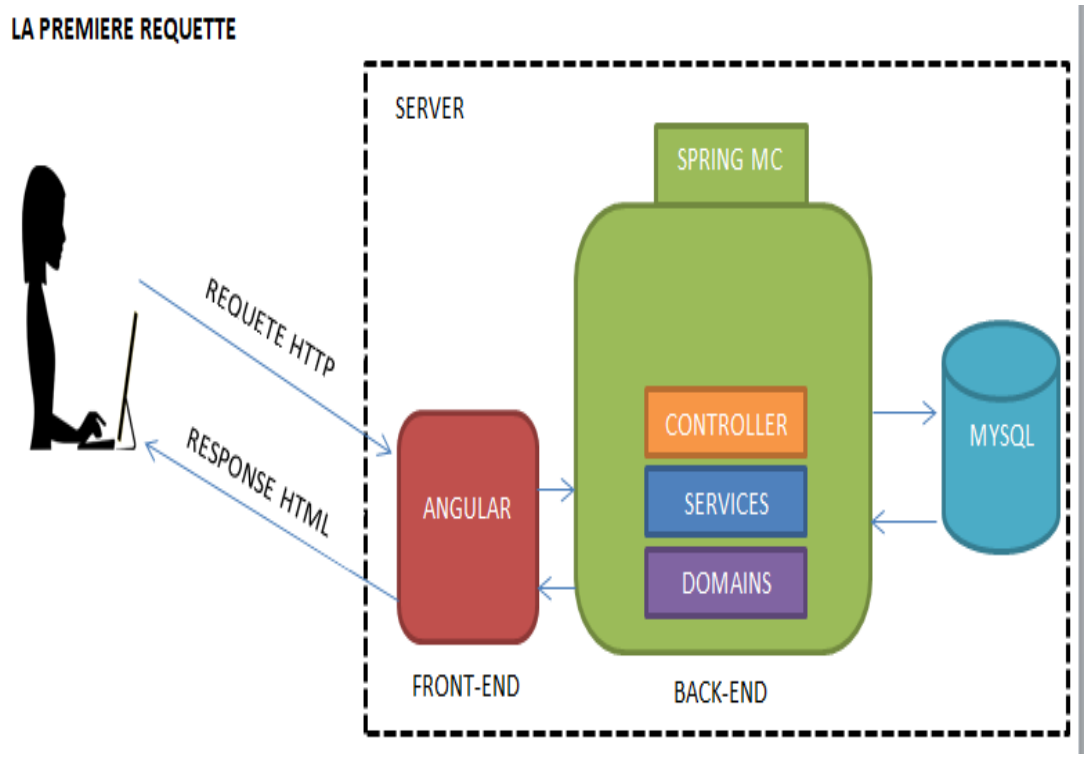


figure 11

After the client loads all code and data, he can browse them without any access to the internet. This is the main reason why we say « WEB APPLICATION ». We need to pay attention to synchronization of data, and we also need to avoid loading the same data on two different browsers at the same time. (we can use session to avoid that).

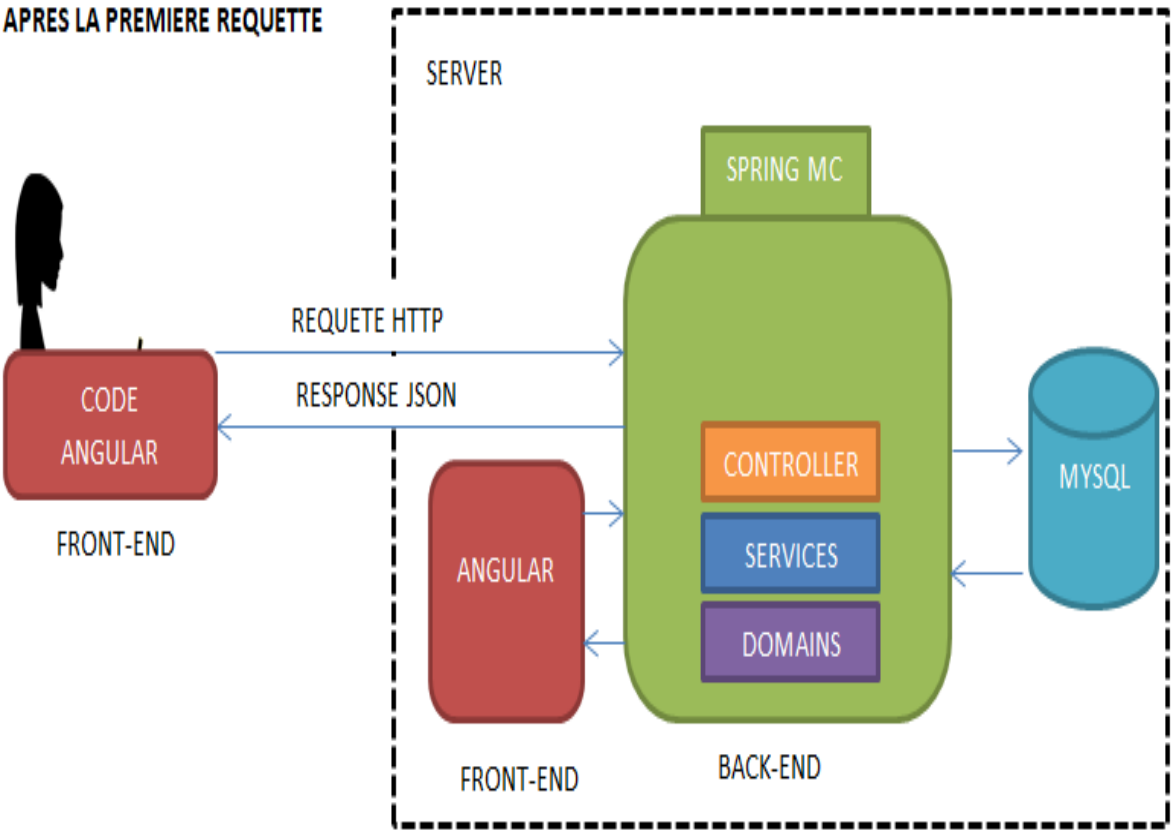


figure 12

Conclusion

One of the first things done in this project was to get the functionality of the application. It was not easy because there was a misunderstanding between the operator and the developers. The way of expressing the need was not very clear. We opted for the flow diagram for better understanding of the the requirements of the functionality. After this step we decide which technologies we should use according to the expectation of the application. for example robustness, fastness and usability. I learned new ways like SCRUM, how to take functionalities using flow diagrams and the life cycle of a big application.