

Academic year : 2018-2019

UNIVERSITY OF SOUSSE
NATIONAL SCHOOL OF ENGINEERING OF SOUSSE



Graduation project report

Applied Computer Engineering
Option: Distributed Systems

Design and development of a broadband QoS monitoring platform

Realized in:



By: Ismail Mekni

Company supervisor : Mr. Sabri Mtibaa, Sofrecom Tunisia
University supervisor : Mr. Aref Meddeb, ENISO

To my family and all my beloved.

Acknowledgements

In conducting this project, I have received meaningful assistance from many quarters which I like to put on record here with deep gratitude and great pleasure.

First and foremost, I express my sincere gratitude to my supervisors, Mr. Sabri Mtibaa, Ms. Marwa Drissi, and Mr. Aref Meddeb who extended their complete support and helped to make me deliver my best.

I would also like to thank all of my teachers at the National Engineering School of Sousse for their continuous help and treasurable training during my study years.

Finally, a special thanks to the jury members who honored us by examining and evaluating this modest contribution.

Contents

1	Project Context	4
1.1	Host company presentation	4
1.2	Project presentation	4
1.2.1	Problem statement	4
1.2.2	Project objectives	5
1.3	Critical analysis of the state of the art	6
1.3.1	State of the art	6
1.3.1.1	SamKnows One	6
1.3.1.2	SMAQ	7
1.3.2	Critical of the state of the art	8
1.3.3	Proposed solution:	8
1.4	Modeling language	9
1.5	Software development methodology	9
1.6	Conclusion	9
2	Specifications of the SMAQ Pobes application	10
2.1	Identification of the actors:	10
2.2	Functional requirements	11
2.3	Non-functional requirements	12
2.4	Requirements analysis	13

2.4.1	Manage devices	14
2.4.2	Manage tests	15
2.4.3	Manage network monitoring	17
2.4.4	Manage statistic and chart configuration	18
2.4.5	Manage user	20
2.5	Conclusion	21
3	Architecture of the SMAQ Probes application	22
3.1	Physical architecture of the SMAQ Probes application	22
3.2	Logical architecture of the SMAQ Probes application	24
3.2.1	Conceptual model	24
3.2.2	Modular decomposition	26
3.2.2.1	Class diagram of entities	26
3.2.2.2	Package diagram	28
3.2.2.3	Device management module	28
3.2.2.4	Metric and test management module	33
3.2.2.5	Network monitoring management module	36
3.2.2.6	Statistic and chart management module	38
3.2.2.7	Authentication and user management module	43
3.2.3	Broadband supervision and monitoring process	45
3.3	Conclusion	47
4	Project Achievements	48
4.1	Development environment	48
4.1.1	Hardware environment	48
4.1.2	Software environment	49
4.1.3	Frameworks and technologies	50
4.2	Achieved work	51
4.2.1	Authentication and user management	51
4.2.2	Device management	53
4.2.3	Metric and test management	55

4.2.4	Network monitoring management	58
4.2.5	Statistics and dashboard management	60
4.3	Conclusion	64
5	General Conclusion and Future Work	65

List of Figures

1.1	Screenshot of SamKnows One dashboard	7
1.2	Screenshot of SMAQ online dashboard	7
2.1	General use case diagram	13
2.2	Device management sequence diagram	14
2.3	Test management sequence diagram	15
2.4	Network monitoring management sequence diagram	17
2.5	Chart management sequence diagram	18
2.6	User management sequence diagram	20
3.1	Physical Architecture of SMAQ Probes solution	23
3.2	Logical architecture	25
3.3	Database entities class diagram	27
3.4	Package diagram of the backend application	28
3.5	Class diagram of the device management module	29
3.6	Sequence diagram of device updating operation	30
3.7	Sequence diagram of Cron updating operation	32
3.8	Class diagram of the test and metric management module	34
3.9	Sequence diagram of the metrics and tests creation and updating operations	35
3.10	Class diagram of the network monitoring management module	37

3.11 Sequence diagram of the alert creation and updating operations . . .	38
3.12 Class diagram of the statistic and chart management module	39
3.13 Sequence diagram of view creation operation	40
3.14 Sequence diagram of dashboard generation operation	42
3.15 Class diagram of the authentication and user management module .	44
3.16 Sequence diagram of the user creation and updating operations . . .	45
3.17 Device and Kafka interactions	46
4.1 User list screen	52
4.2 User updating screen	52
4.3 Authentication screen	53
4.4 Device list screen	54
4.5 Device updating screen	54
4.6 Cron list screen	55
4.7 Cron updating screen	55
4.8 Test list screen	56
4.9 Test general information form	56
4.10 Test elment's configuration form	57
4.11 Test formula's configuration form	58
4.12 Test element scheduling configuration form	58
4.13 Alert list screen	59
4.14 Alert configuring screen	59
4.15 Chart customization screen	60
4.16 Online dashboard screen	61
4.17 Time period filter	61
4.18 Device position filter	62
4.19 Comparison filters	62
4.20 Triggered alert detail screen	63
4.21 Upload speed chart	64

List of Tables

1.1	Comparison of the state of the art	8
2.1	Device management description	14
2.2	Test management description	16
2.3	Network monitoring management description	17
2.4	Chart management description	19
2.5	User management description	20

Abstract

This report describes our graduation project which was carried out at Sofrecom Tunisia. The project aims to design and develop a remote and communicating device platform to achieve supervision and monitoring and assess the quality of service (QoS) of fixed access broadband networks.

With our solution, the user could manage connected probes and create broadband test configuration before running them in the probes. The user can also manage the statistics configuration of testing results with attractive charts with customized alerts to ease broadband supervision. The probe also helps network supervisors get accurate statistics according to time periods and zone areas.

Keywords

Network – Broadband – Quality of service - Quality of experience – Key Performance Indicators – Data analytics

Résumé

Ce rapport décrit notre projet de fin d'études mené chez Sofrecom Tunisie. Le projet vise à concevoir et développer une plate-forme d'appareils distants et communicants pour la supervision et surveiller et évaluer la qualité de service (QoS) des réseaux d'accès fixes large bande.

Avec notre solution, l'utilisateur peut gérer les sondes connectées et créer une configuration de test large bande avant de les exécuter dans les sondes. L'utilisateur peut également gérer la configuration des statistiques et des résultats de test avec des graphiques attrayants et des alertes personnalisées pour faciliter la supervision. La sonde aide également les superviseurs de réseau à obtenir des statistiques précises en fonction d'une période et d'une zone données.

Mots clés

Sonde – Haut débit – Qualité de service – Qualité d'expérience – Indicateurs clé de performance – Analyse des données

General introduction

With the global spread of the internet nowadays, many mobile and internet services operators appeared. In the coming days, connected devices will spread massively worldwide, especially with the appearance of new technologies such as the internet of things, big data, personal area networks (PAN), artificial intelligence (AI). So mobile and internet services operators desire to achieve a better quality of service for their customers. To do so, operators design and develop solutions for broadband monitoring and supervision, mainly, to massively assess and monitor the quality of service of networks access perceived by customers. Sofrecom Tunisia, subsidiary of Orange Group, attempts to serve this purpose by designing and developing a platform, "SMAQ Probes", for broadband supervision and monitoring and specifically to massively assess the quality of service (QoS) of fixed network access. In this context comes our mission during the graduation project internship.

This report contains four chapters as follows:

The first chapter titled "Project context" will be devoted to the company presentation and to set the project in its general context.

The second chapter "Requirements analysis and specification" will be about the global system analysis to design and develop.

The third chapter "Design of the physical and logical architectures" will be dedicated to present the physical and logical architectures of our solution, also, we will explain the process behind the broadband monitoring and supervision.

Last but not least, the final chapter "Project achievements" will be about the application achievements. First, we will present the development environments and technologies and tools, secondly, we will illustrate our application with several interfaces.

Finally, we will close the report with a general conclusion. Future work and perspectives will be mentioned at last to give some ideas to improve our solution.

In the first place, this chapter will be about the host company, Sofrecom Tunisia presentation, a consulting and engineering firm specializing in telecommunications. Then, we will talk about work and project environments, by putting the light on the project goals and the analysis of the state of the art. Finally, we will present our software development methodology and the modeling language that we are going to use.

1.1 Host company presentation

Sofrecom, a subsidiary of Orange Group, has built up 50 years' worth of unique know-how in the telecoms operator line of business, making it a world leader in telecom consultancy and engineering. Sofrecom Tunisia is the youngest subsidiary of Sofrecom. Launched in November 2011, Sofrecom Tunisia expands Sofrecom's presence in North Africa and the Middle East region to meet the growing demand for dedicated solutions and to offer its customers adapted and competitive offers.

1.2 Project presentation

1.2.1 Problem statement

Operators of mobile and internet services have probes, Key Performance Indicators (KPIs), installed in several network elements, but they desire to appreciate the qual-

ity perceived by their customers (Quality of Experience, QoE) in order to improve their network performance and reliability.

In order to get the accurate information about the quality of experience, we should get the measurements according to zone area and specific periods of time. Also, it is difficult to manage the configuration and QoS/QoE tests running on the widespread probes, which indicates a lack of flexibility. The internet usage is improving fast; therefore we should improve the quality of service measuring as well.

Sofrecom was involved in an RFP (Request For Proposal) from a regulator where a part of the request was to massively assess and monitor the QoS of fixed networks access (ftth).

None satisfying “on the shelf” tool solution is found either inside or outside the Orange Group.

1.2.2 Project objectives

This work will be considered as a graduation project to obtain applied computer science engineering degree from the national engineering school of Sousse (ENISO).

Taking into account this situation, Sofrecom brainstormed about a tool that is internally developed. The main project goal is to design and develop a platform for remote and communicating devices, to serve broadband monitoring and supervision. SMAQ Probes application should fulfill these needs:

- The creation and interpretation of messages exchanged with terminals and embedded equipments.
- Information exchanges with client applications (northbound interface).
- Management of messages from terminals (southbound interface).
- Sending requests and messages to the terminals.
- On-the-fly supervision of exchanges.
- Management of supervised elements.
- Resolution of message referral rules.

1.3 Critical analysis of the state of the art

This step is essential to propose our solution in the relation to those offered by other companies. To do so, we studied the characteristics and the features of some available apps while we focus on their weak points.

1.3.1 State of the art

1.3.1.1 SamKnows One

SamKnows One is a cloud-based analytics platform that includes a full range of measurement agents for fixed and cellular internet connection with a global test infrastructure. SamKnows solution stores and visualizes performance data in real-time. This solution is implemented by a UK company “Sam” founded in 2008 by Sam Crawford. SamKnows One has developed a suite of tests[1] as follows:

- Speed tests: includes download and upload over TCP and UDP speed tests.
- Latency, loss and jitter: latency, jitter and packet loss over UDP, latency over HTML5.
- DNS resolution: DNS resolution time and failure rate (UDP).
- Web browsing: web browsing test over TCP.
- CDN performance: content delivery network (CDN) measurements over TCP.
- Video streaming: video streaming measurements that stream real content from major video streaming providers.
- Gaming: measures performance for a number of major games.
- Online storage: tests upload and download from popular online storage services.
- Voice over IP: measures the quality of a voice call between client and test server.
- Traceroute: tests the path that traffic takes around the internet, it is useful in diagnosing routing issues.
- Data usage: measures of data used on the broadband connection.

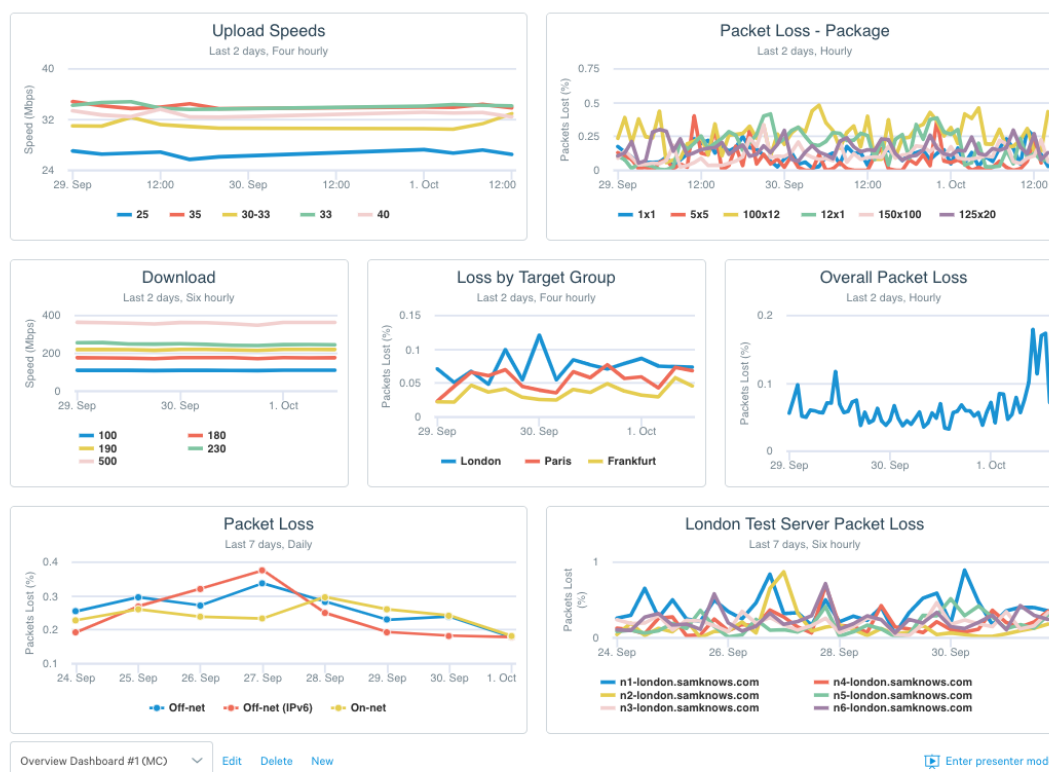


Figure 1.1: Screenshot of SamKnows One dashboard

1.3.1.2 SMAQ

SMAQ is a solution implemented by Sofrecom to generate reports and analysis based upon different broadband tests, this solution is used by the Orient Middle East and Africa Orange affiliates.

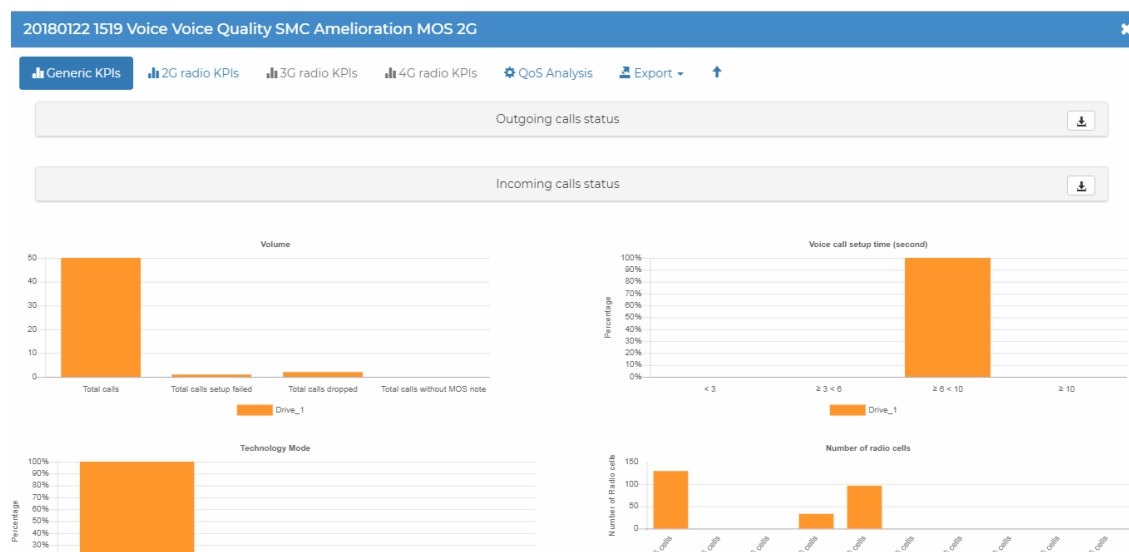


Figure 1.2: Screenshot of SMAQ online dashboard

1.3.2 Critical of the state of the art

The following table shows in detail the differences between the two mentioned solutions:

Table 1.1: Comparison of the state of the art

Solution	SamKnows One	SMAQ
Paying solution	YES	NO
Analytics	YES	YES
Custom dashboard	YES	NO
Mapping data	YES	NO
Generate reports	YES	YES
Network monitoring (alerting)	YES	YES
User management	YES	NO
Device management	NO	NO
Test management	NO	NO

Sofrecom Tunisia is looking for an open source solution for the issue in question, so SamKnows One solution is not convenient for them. In the other hand SMAQ provides just the basic functionalities. The weak point in the mentioned solutions, they don't give users the access to the devices' configuration. Network supervisors can't customize tests to satisfy their specific needs.

1.3.3 Proposed solution:

The solution proposed by Sofrecom Tunisia is to design and develop a platform for broadband monitoring and supervision, "SMAQ Probes", the solution should respond the following needs:

- Online device management and task scheduling.
- Online test management.
- Online statistic and customized chart configuration.
- Online alert configuration and customization.
- Online user management.

1.4 Modeling language

During the work on our solution we used UML “Unified Modeling Language” for describing and modeling the specifications of our project. UML is a flexible and versatile modeling language, also it is the most popular and widely used by the community. We are going to present some diagrams from UML that we find it useful during our work:

- Use case diagram: it helps to structure the needs of users and the corresponding objectives of our system by identifying its users and their interactions.
- Sequence diagram: it is a time focus representation of objects and their interactions.
- Package diagram: it gives an overview of the application packages. It is a high abstraction that presents the application modularity.
- Class diagram: it gives a presentation of classes and interfaces of our system and relations between them.

1.5 Software development methodology

Before starting the project design and development, we should choose the appropriate software development methodology to work with. The software development methodology helps to describe the different phases and the sequences of the application development process.

During our project, we used agile kanban because it is the most convenient method to us. I am the only intern working on the project. Changes in the project can happen any time. We are continuously improving the flow of our work. We are trying to limit work in progress and to maximize efficiency. Also, we focus on reducing the time it takes to take a project from start to finish.

1.6 Conclusion

In this chapter, we presented the general context of SMAQ Probes application by presenting the host company Sofrecom Tunisia, the problem statement and the state of the art.

In the next chapter, we will model the requirements of our solution through use case diagrams.

Specifications of the SMAQ Pobes application

The requirements analysis and specification phase is an essential step for the development of a new application for broadband monitoring and supervision. It allows presenting SMAQ Probes' features in detail.

In this chapter, the first part is devoted to identify the different actors of the application who are interacting with the system, and to give the functional and non-functional requirements definitions. Subsequently, we present the general system analysis using use case diagrams.

2.1 Identification of the actors:

An actor is an abstraction of a role of actual user who is in a perpetual interaction with the application. Following on, our system's actor along with his role and granted permissions.

Internal actors

- **Application administrator:** the administrator is responsible for managing user information and permission, he has the permission to create, read, update, and delete users. Also, he has the permission to check all the configuration like the device and test configuration.

- Network supervisor: the network supervisor has the permission to read the configuration without editing it. He has the right to supervise the quality of experience “QOS”.
- Network and broadband administrator: he has all the rights of the network supervisor. In addition, he has the permission to create, read, update, and delete the broadband monitoring configuration.

External actors

- Probe: the probe is the entity able to receive the device and test configuration, and sends the tests’ results to the server after running tests.

2.2 Functional requirements

Functional requirements refer to the primary functions that each component of SMAQ Pobes must exhibit. It is a set of services which are:

For the web application

- The application should give the administrator the hand to manage user account.
- The application should give permitted users the possibility to customize their dashboards.
- The application should give permitted users the access to manage the network monitoring configuration(alerting).
- The application should provide permitted users with the access to manage the test and device configuration.
- The application should be able to process received metrics data in real-time.

For the hardware

- The boards should be able to receive and implement their configuration in real-time.

- The boards should be able to run tests according to the time schedule and send results to the server.
- The boards should be able to send their current configuration (location, IP address, device identifier, jobs configuration).
- The boards should be able to keep tests results if the server is not available.

2.3 Non-functional requirements

Non-functional requirements refer to several key features that are beyond the purpose of the solution, they specify criteria that judge the operation of the system, rather than specific behaviors in order to ensure the client's satisfaction.

Extensibility

The system must be open to some extension like for example adding new features if needed without radical modification in the code.

Performance

The web application should be as efficient as possible with especially a good response time. Users should be able to receive the quality of experience (QOE) from the cloud server within a reasonable amount of time.

Re-usability

The system shouldn't be exclusive for our case and must be adaptive to other use cases.

Robustness

The system must cope with errors during execution and should be able to reboot within a short time in case of failure.

Security

The user's personal information must be kept safe from others and only system administrator has permissions to access it. The broadband monitoring and supervision access must be permitted to the supervisors of the network in question.

2.4 Requirements analysis

On one hand, this section offers a better understanding of the mentioned requirements by declaring them in a semi-formal way. On the other hand, it emphasizes the interactions between the actor and our application. In contemplation of breaking down the complexity of these goals, we use the use case diagrams.

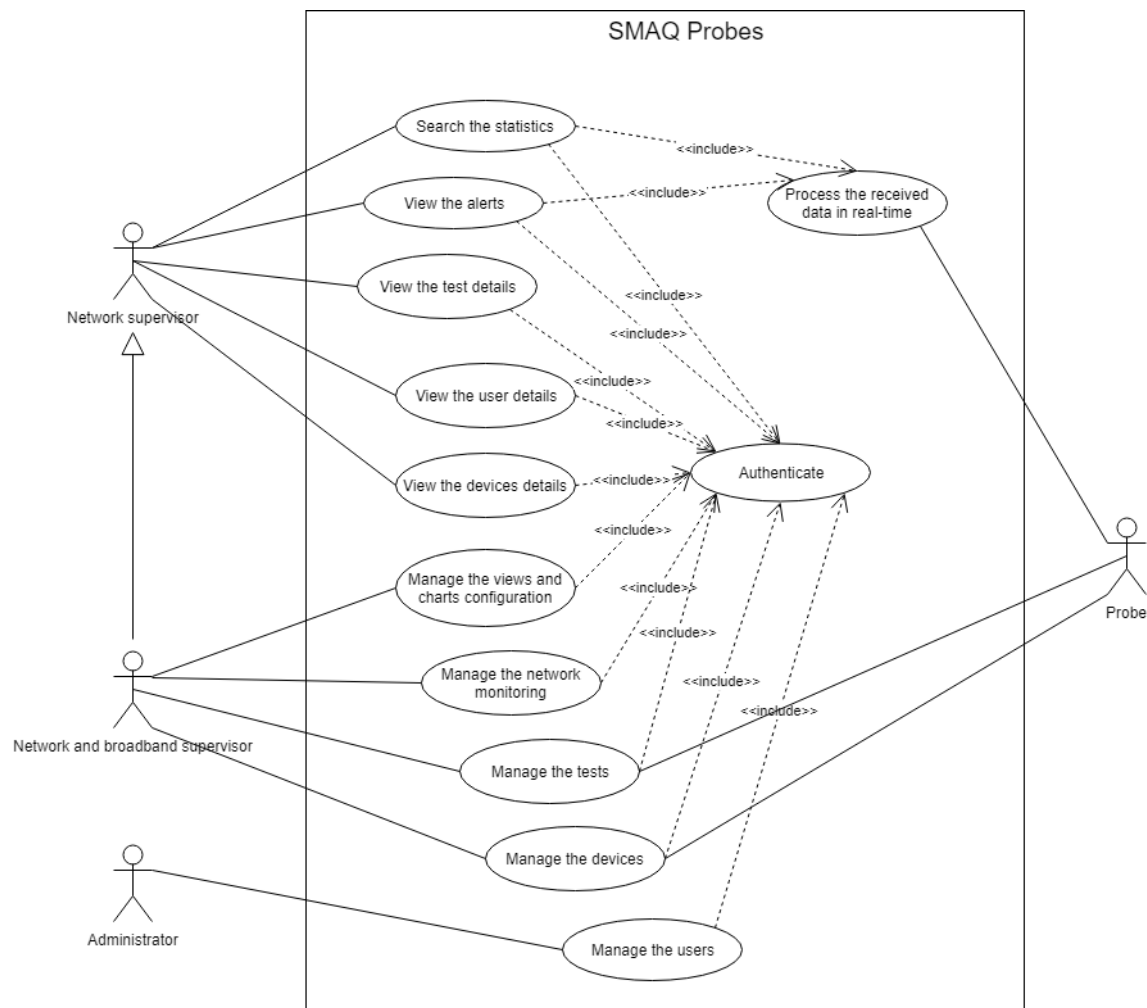


Figure 2.1: General use case diagram

As shown in the general use case diagram (fig.2.1), only the administrator can register all types of users. All the features of the application must go through authentication. The network and broadband supervisor is responsible for managing network monitoring, devices configuration, tests, and statistics configuration. Any configuration that concerns probes is delivered to them. Probes send their information and runs tests according to a job scheduler then probes send tests results to the server. Thus, our system process the received data in real-time. Finally, our application is prepared to generate statistics and quality of service for the network supervisor.

2.4.1 Manage devices

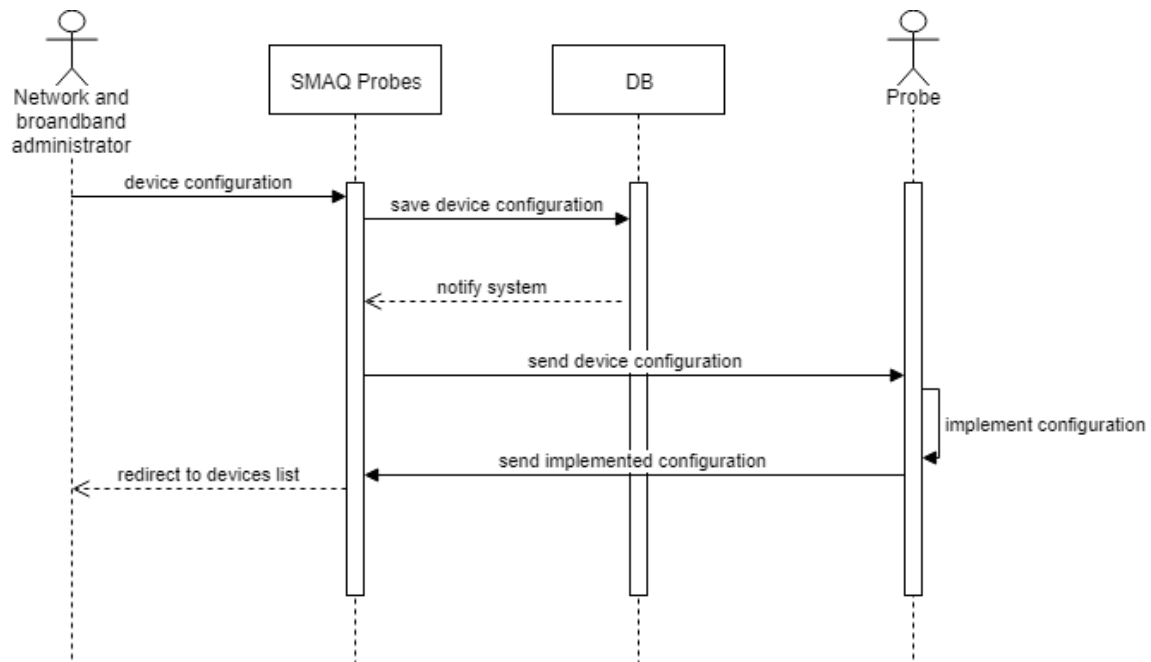


Figure 2.2: Device management sequence diagram

Table 2.1: Device management description

Title	Device management
Author	Ismail MEKNI
Version	1.0
Objectives	Allow users to manage the connected devices' configuration
Actors	Network and broadband administrator – SMAQ Probes – Probe
Pre-conditions	The user should authenticate as network and broadband administrator. The device should be connected.
Post-conditions	New device configuration is persisted in the database. New device configuration is implemented in the probe.
Story	1. The user enters the new device configuration (status, IP address, client name, job scheduling). 2. The user submits the changes.
Alternative story	
Exceptional story	The device in question is not connected; the configuration's message is suspended waiting the device to reconnect.

As shown in the device management sequence diagram (fig.2.2), this functionality is permitted to the network and broadband administrator. User can access to the device list. The user can select a device to edit its configuration (IP address, location, registered client name, job scheduling), if user submits the new configuration, the

configuration is sent to the backend system to persist it to database. Also, the configuration is delivered to the device in question. The probe, device, implements the changes. Finally, the probe sends back the implemented configuration to the system as a confirmation. This feature helps the network and broadband administrator to change the device configuration remotely.

2.4.2 Manage tests

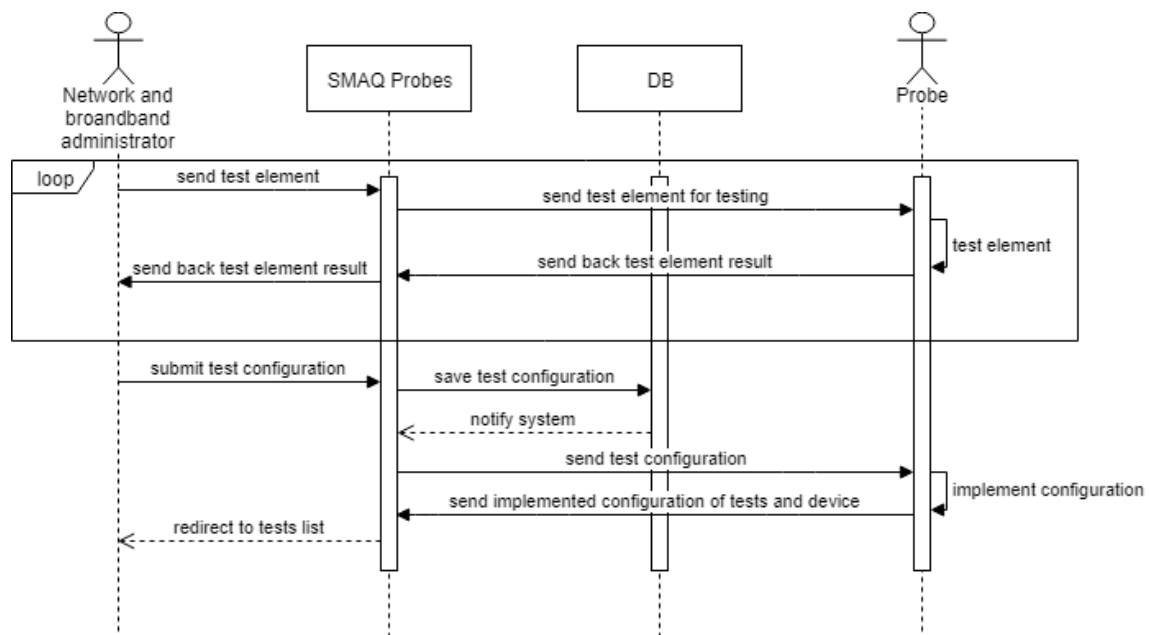


Figure 2.3: Test management sequence diagram

Table 2.2: Test management description

Title	Test management
Author	Ismail MEKNI
Version	1.0
Objectives	Allow users to manage the test configuration.
Actors	Network and broadband administrator – SMAQ Probes – Probe
Pre-conditions	The user should authenticate as network and broadband administrator. At least one device should be connected.
Post-conditions	The new tests' configuration is persisted to database. New tests are running on the probes.
Story	1. The user tests each element directly on the device. 2. The user submits the test configuration with all the elements.
Alternative story	
Exceptional story	There is no connected device, so user can't test elements, the operation with be suspended until at least one device reconnects.

As shown above in the diagram (fig.2.3), the access to the test configuration is granted to users with network and broadband administrator role. To edit the test configuration, the user should enter the test elements, each element must be tested directly on the probe, the device, and then test element's result is sent back to the user. After creating and testing all the elements, a new test configuration is sent to the backend system. The configuration is persisted to database. The new test configuration is sent to the probes. All the probes implement the new test configuration. Finally, a signal message is sent from all the probes holding the current configuration. This use case allows the network administrator to customize and improve his tests remotely.

2.4.3 Manage network monitoring

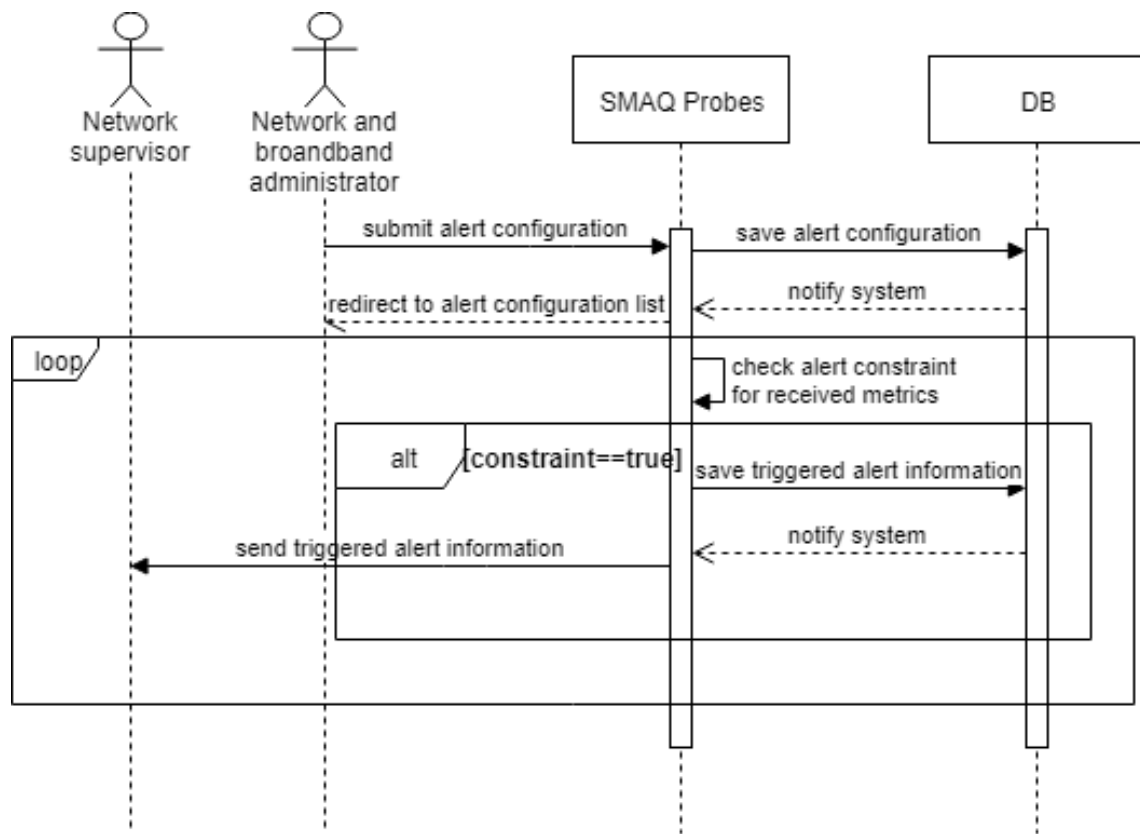


Figure 2.4: Network monitoring management sequence diagram

Table 2.3: Network monitoring management description

Title	Network monitoring management
Author	Ismail MEKNI
Version	1.0
Objectives	Allow users to manage the network monitoring configuration, alerting system.
Actors	Network and broadband administrator – Network supervisor – SMAQ Probes
Pre-conditions	The user should authenticate as network and broadband administrator to access the network monitoring management. To view triggered alerts, the user should authenticate as a network supervisor.
Post-conditions	The new alert configuration is persisted to database. An alert listener is running on the received metrics.
Story	1. The user sends the alert configuration containing the constraint. 2. The network supervisor can view alerts if it is triggered.
Alternative story	
Exceptional story	

As shown in the diagram (fig.2.4), the user authenticates as a network and broadband administrator. This feature aims to configure customized alerts. The user creates alerts with a specific constraint. This configuration is persisted to the database. An alert checker is run for every received metrics data, if the constraint is satisfied an alert with full description is triggered. The triggered alerts are persisted to the database, so, the network supervisor can check them. This feature allows users to facilitate the network monitoring.

2.4.4 Manage statistic and chart configuration

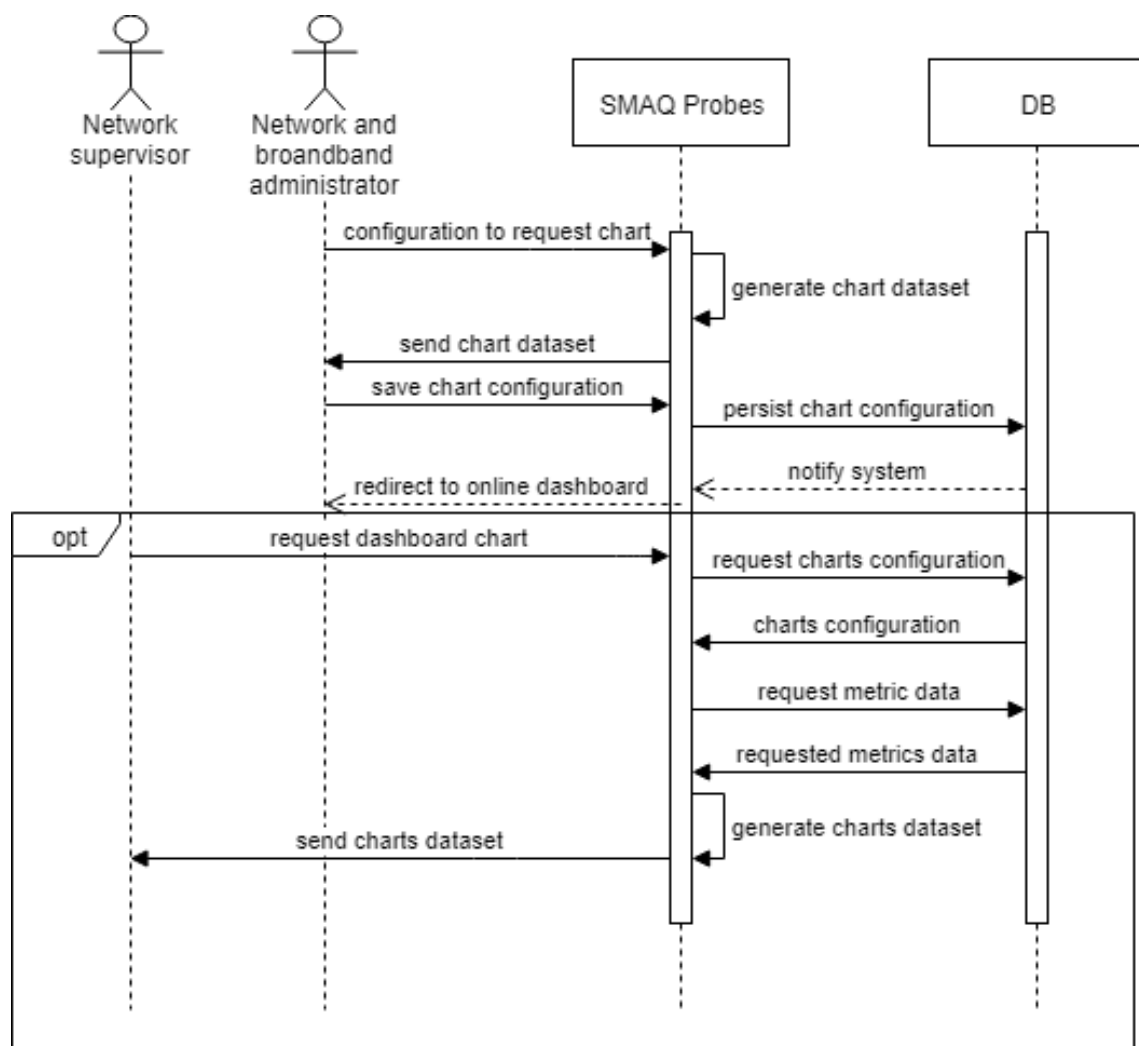


Figure 2.5: Chart management sequence diagram

Table 2.4: Chart management description

Title	Chart and statistic management
Author	Ismail MEKNI
Version	1.0
Objectives	Allow users to manage charts configuration.
Actors	Network and broadband administrator – Network supervisor – SMAQ Probes
Pre-conditions	The user should authenticate as network and broadband administrator to access chart and statistic management. To view the dashboard, the user should authenticate as a network supervisor.
Post-conditions	The new chart configuration is persisted to the database. New chart is added to the dashboard.
Story	1. The user sends the chart configuration, the system generates the chart dataset. 2. The chart displayed to user. 3. The user submits the chart configuration.
Alternative story	
Exceptional story	

As we can see in the above sequence diagram (fig.2.5), to access to the chart and statistic configuration, users should have the network and broadband administrator privileges. The user enters the chart parameters, so our system generates the chart in question. The user submits this configuration to be persisted and added to dashboard. Thus, users with network supervisor permission can see the configured charts on the dashboard. This feature aims to allow users to create customized charts and views. These views allow users to get the accurate information about the QoS, so, they can improve their network service.

2.4.5 Manage user

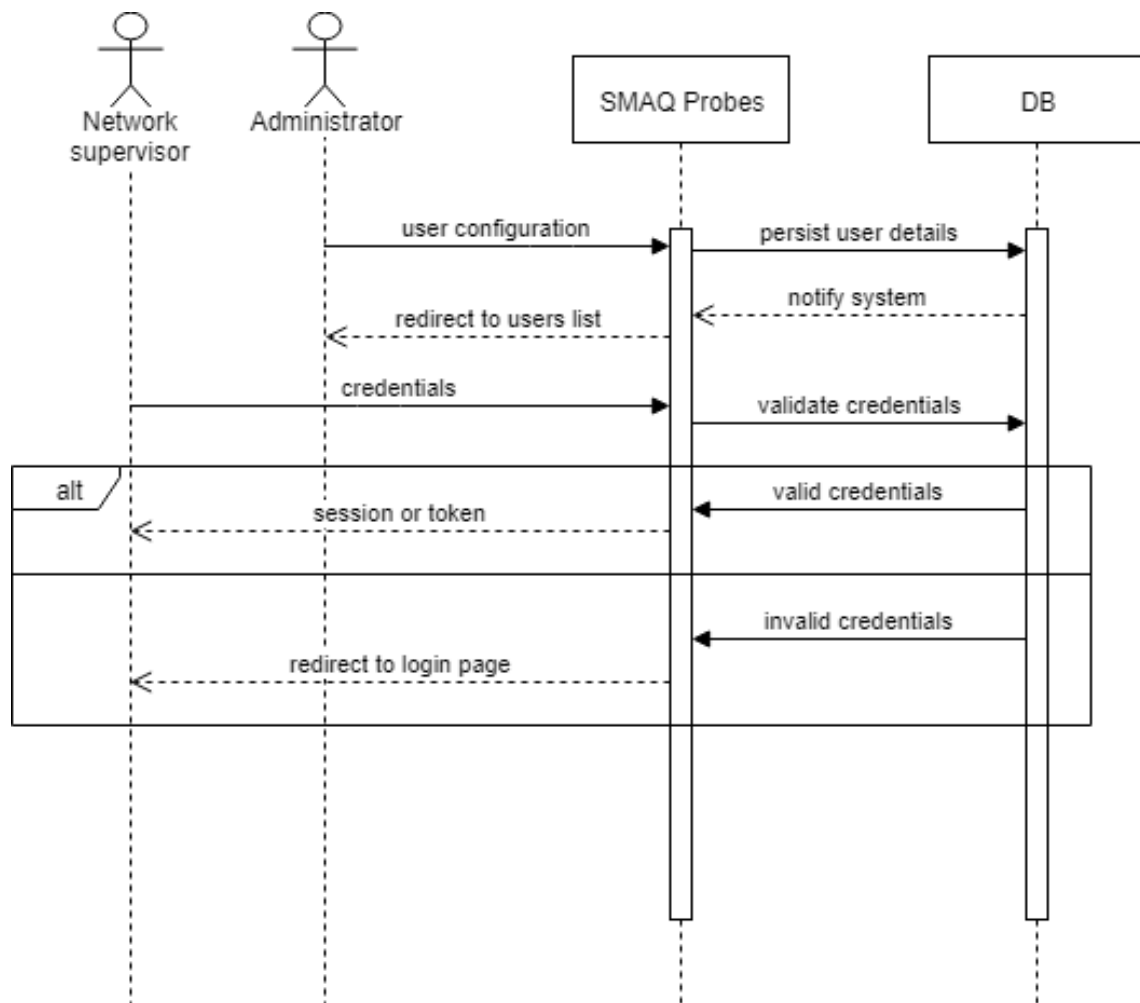


Figure 2.6: User management sequence diagram

Table 2.5: User management description

Title	User management
Author	Ismail MEKNI
Version	1.0
Objectives	Allow administrator to manage the user configuration and details.
Actors	Administrator – Network supervisor – SMAQ Probes
Pre-conditions	The user should authenticate as application administrator to access the user management feature.
Post-conditions	The user credential is persisted to database. User can authenticate.
Story	1. The administrator submits the user configuration. 2. The user submits his credentials.
Alternative story	
Exceptional story	If the user enters invalid credentials, he is prompted to try to login again.

As shown in the user management sequence diagram (fig.2.6), the user management feature is only allowed to the application administrator. The administrator enters the credentials of each user. Thus, the user is now registered to the application and he can access to the features depending to his privileges. To sign in to the application, the user enters his credentials, generally a username and a password, if the credentials are valid, he is redirected to the dashboard, and else he is prompted to login again.

2.5 Conclusion

Throughout this chapter, we specified and analyzed the requirements that the SMAQ Probes application should deliver to users, and we presented the main scenarios and the use cases that it should offer.

The next chapter aims to go a step further in the process of developing our application via presenting the design of the different components of our project.

Architecture of the SMAQ Probes application

In order to reach the appropriate result as described in the specifications of SMAQ Probes application, we need to clarify our project's main architecture as well as the architecture of its components. This chapter focus on designing a suitable structure for the broadband monitoring and supervision system. This step is considered as the most crucial of the process because it prepares the ground for the implementation phase of our application.

3.1 Physical architecture of the SMAQ Probes application

The architecture presented in (fig.3.1) is the physical architecture of our system. It represents the physical layout of our system and its components in a global diagram and it refers to some representations of the structure or organization of the physical elements that build the system.

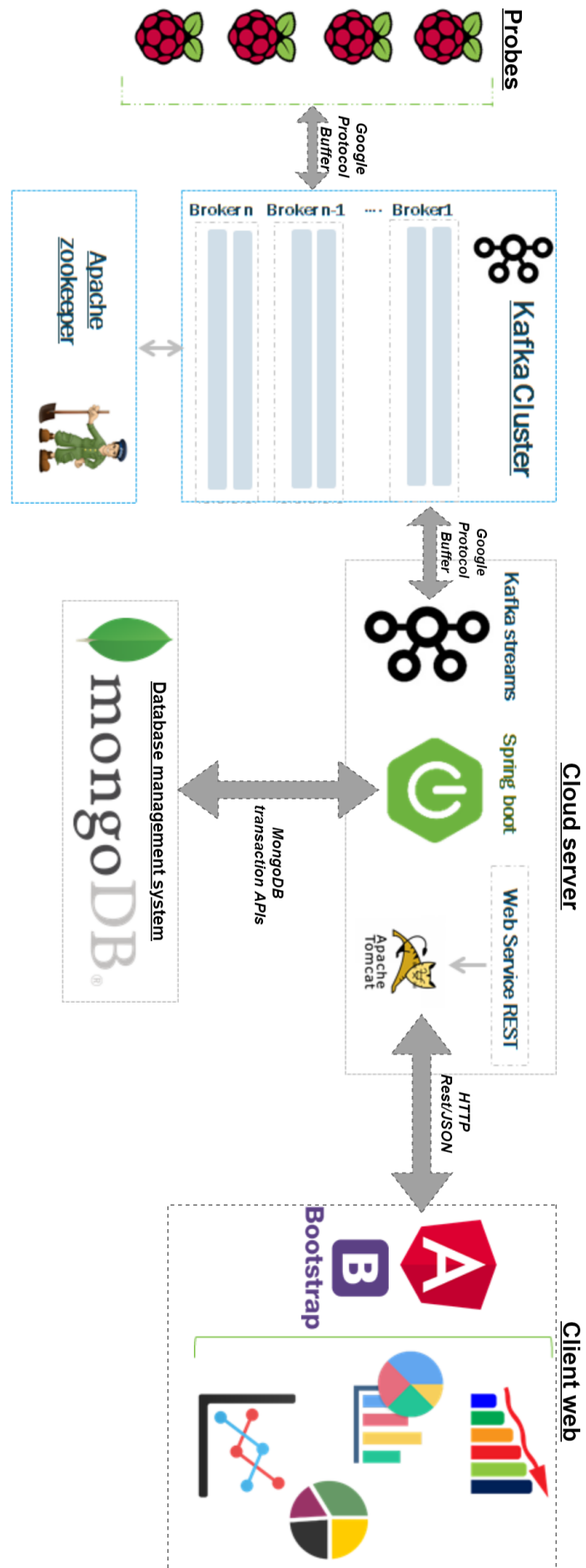


Figure 3.1: Physical Architecture of SMAQ Probes solution

This architecture describes the main components of the system and how they interact in order to achieve the objectives mentioned in the previous chapter.

The system is composed mainly from the following parts: the probes (Raspberry Pi boards), the user interface (web browser) and the cloud server including Kafka cluster, the database management system and our web application.

The probes represent the entity that executes the scheduled tests and sends the metrics to Kafka cluster through the Google protocol buffer.

The client web part represents the part with which the final users interacts and it is essentially: the Web platform accessible by the application users, application administrator, network supervisors and network and broadband administrators.

The third part, the cloud server, is where the application is hosted, this part is responsible for receiving and processing data coming from Kafka cluster, also it is responsible for the data analysis and configuration persistent to our database.

3.2 Logical architecture of the SMAQ Probes application

3.2.1 Conceptual model

The figure (fig.3.2) shows the logical architecture of the system.

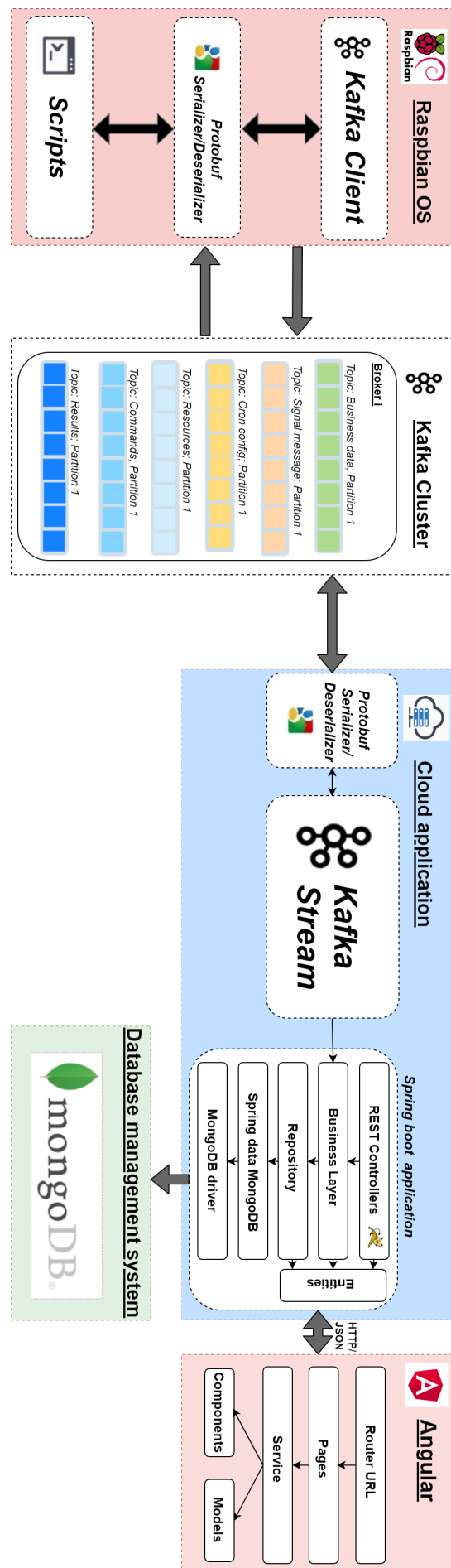


Figure 3.2: Logical architecture

According to the figure of the logical architecture, the system is composed from three major parts:

- Probes: this parts represents widespread devices, these devices are the entities that handles tests. There are several scripts responsible for running the tests with efficient schedules. All the messages are serialized with Google Protocol Buffer. There is a Kafka client responsible for publishing and receiving messages[2].
- Cloud application: this part is the entity that holds the application logic. It holds within it a Google Protocol Buffer converter. Kafka stream is the layer that process the received data in real-time with high performance[3]. Spring Boot application is a three tier web application, it is responsible for managing the features of our system including metrics analytics.
- Angular: this entity is the frontend application accessible to users across the Web[4].

The connection between the web application and the user interface is guaranteed through HTTP protocol and REST web services.

Kafka cluster is playing the role of a middleware between the probes and the back-end server.

Finally, we have our database, we have chosen MongoDB as a database management system for performance reasons, and it is accessible only through the three tier web application.

3.2.2 Modular decomposition

3.2.2.1 Class diagram of entities

The (fig.3.3) shows the class diagram of our system, this diagram summarizes relationships between our entities in the database. This class diagram is efficient in the case of MongoDB database management system. MongoDB is an oriented document database management system.

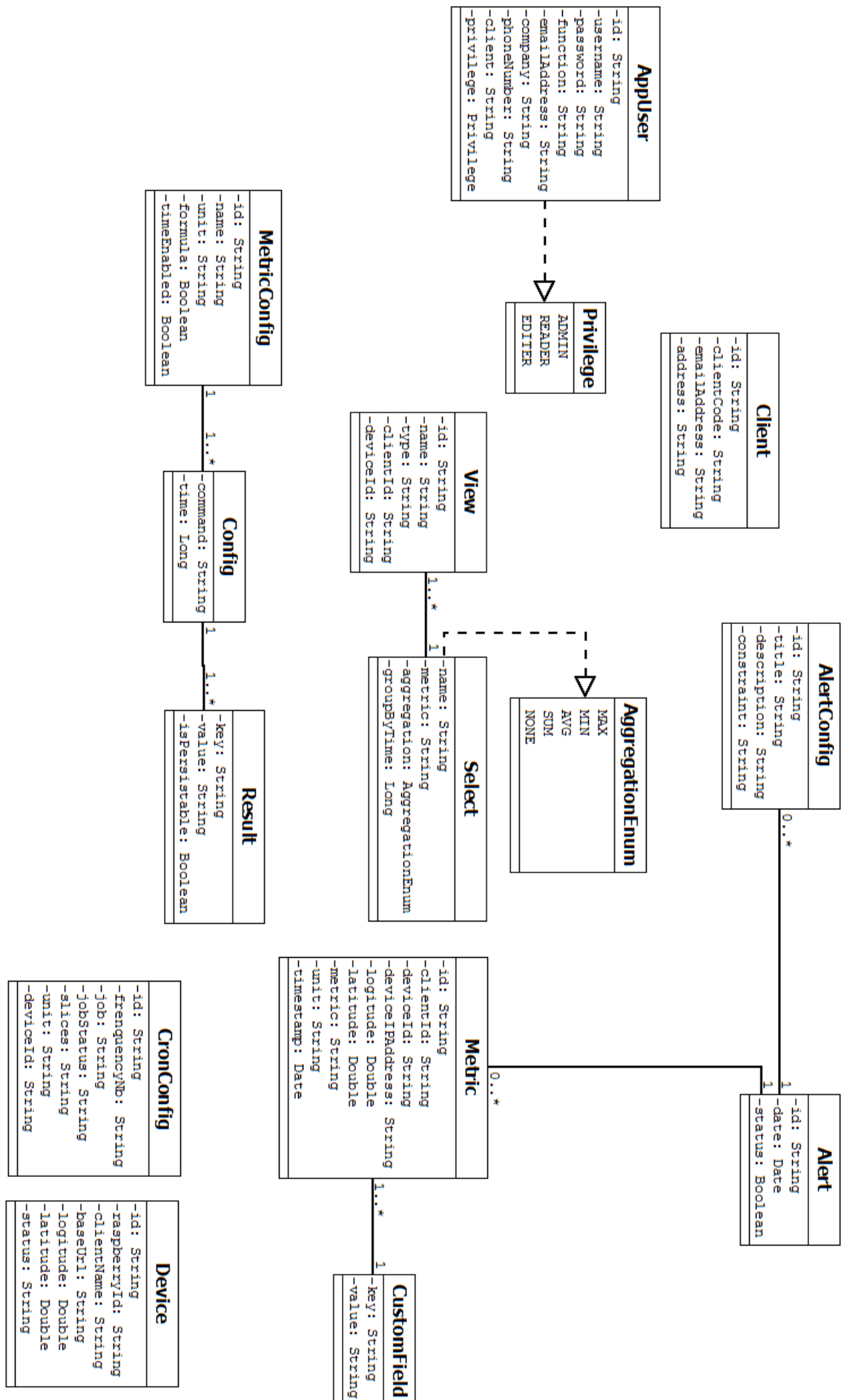


Figure 3.3: Database entities class diagram

3.2.2.2 Package diagram

The package diagram is a static view that serves to globally describe the different components of the application. The (fig.3.4) presents the package diagram of our solution in order to have an overview of its different elements.

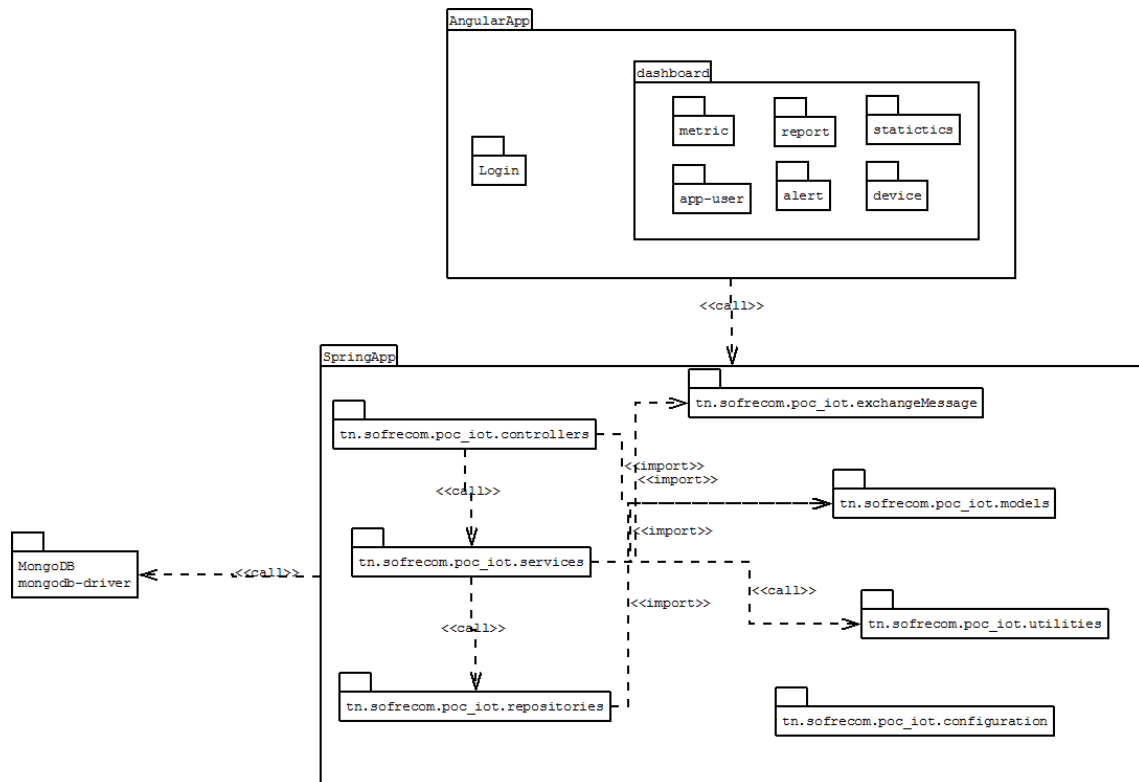


Figure 3.4: Package diagram of the backend application

3.2.2.3 Device management module

This module describes device management process. This module is about editing, deleting, and checking available devices. There is no meaning of creating device because the device information is sent by it if it is connected.

The following class diagram (fig.3.5) presents the involved classes and components that build the module.

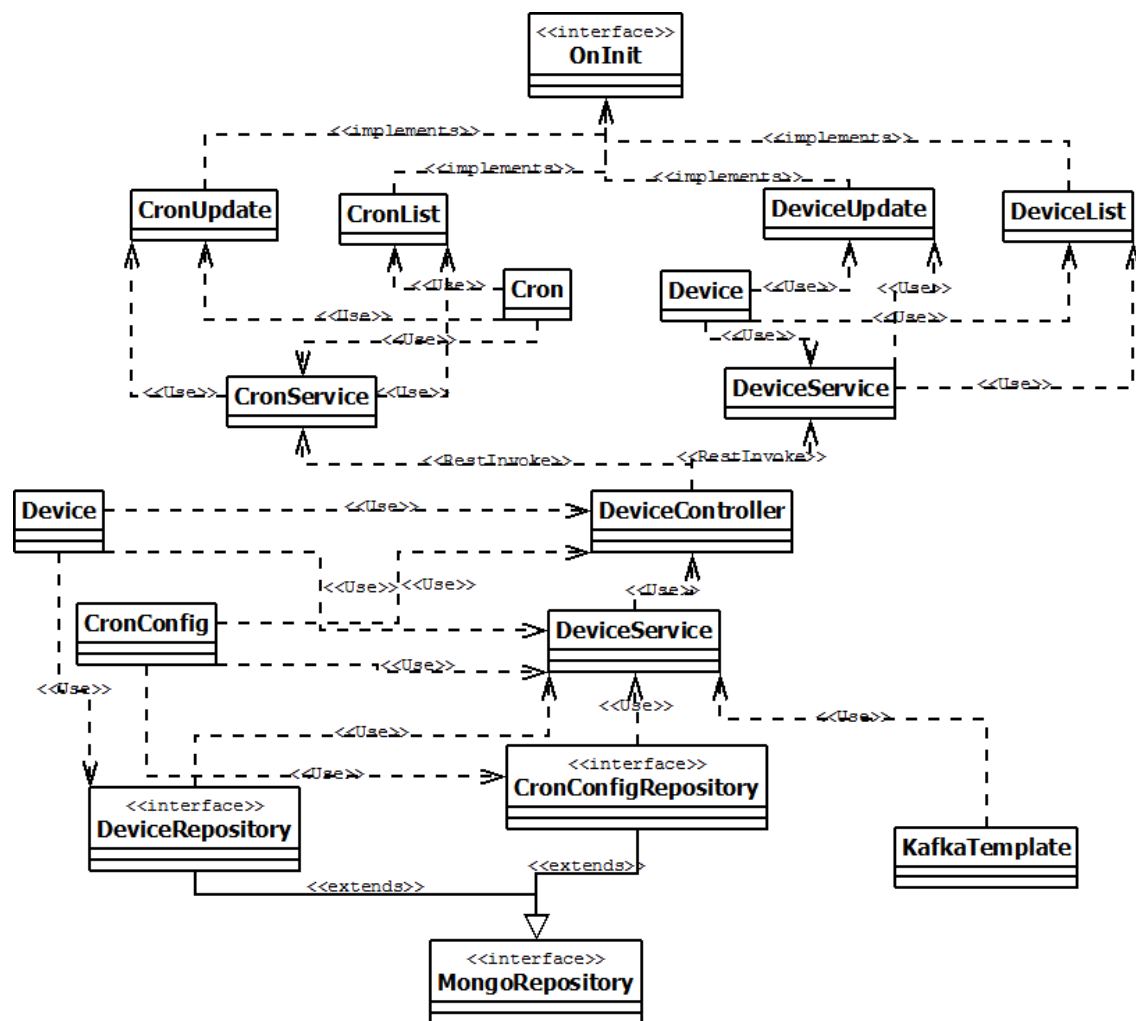


Figure 3.5: Class diagram of the device management module

Then, we have the sequence diagram that describes the interactions between the module's components; this diagram presents the update device operation:

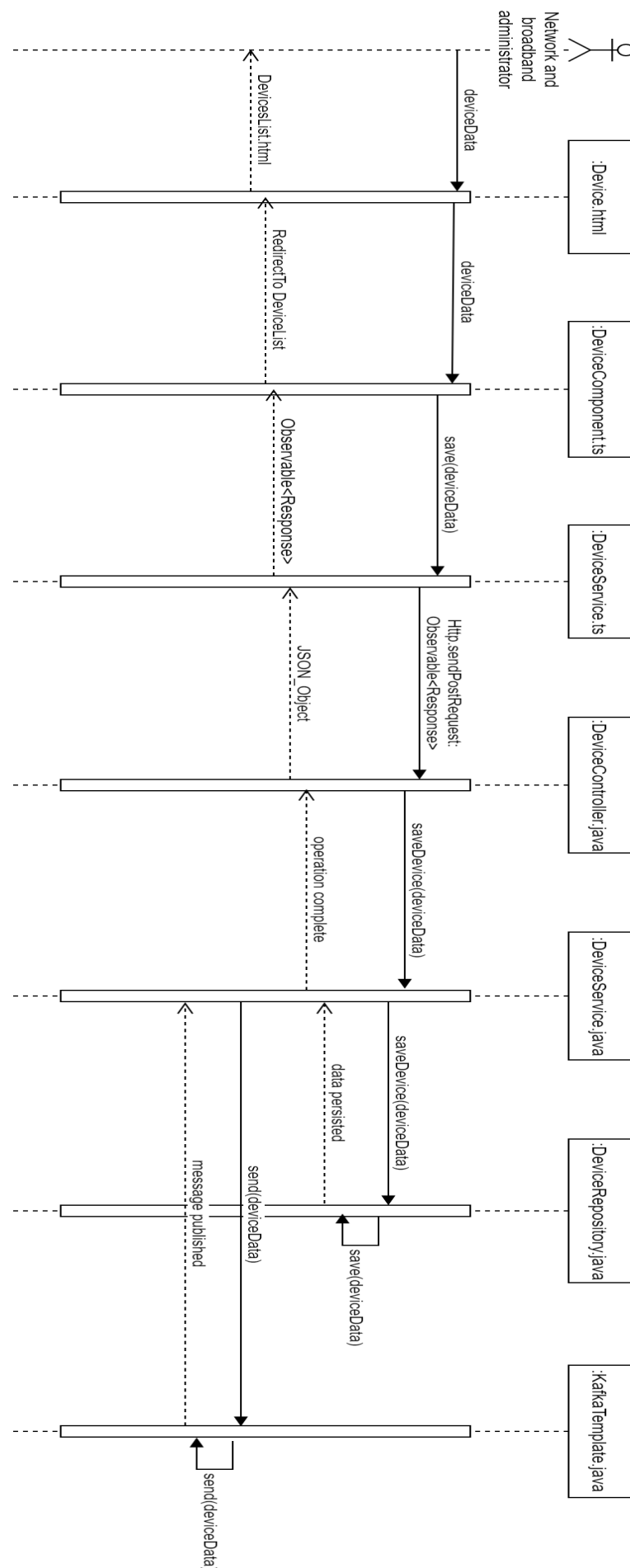


Figure 3.6: Sequence diagram of device updating operation

The diagram below is the sequence diagram for the update Cron operation. Cron is a job already running on the probe according to a specific schedule.

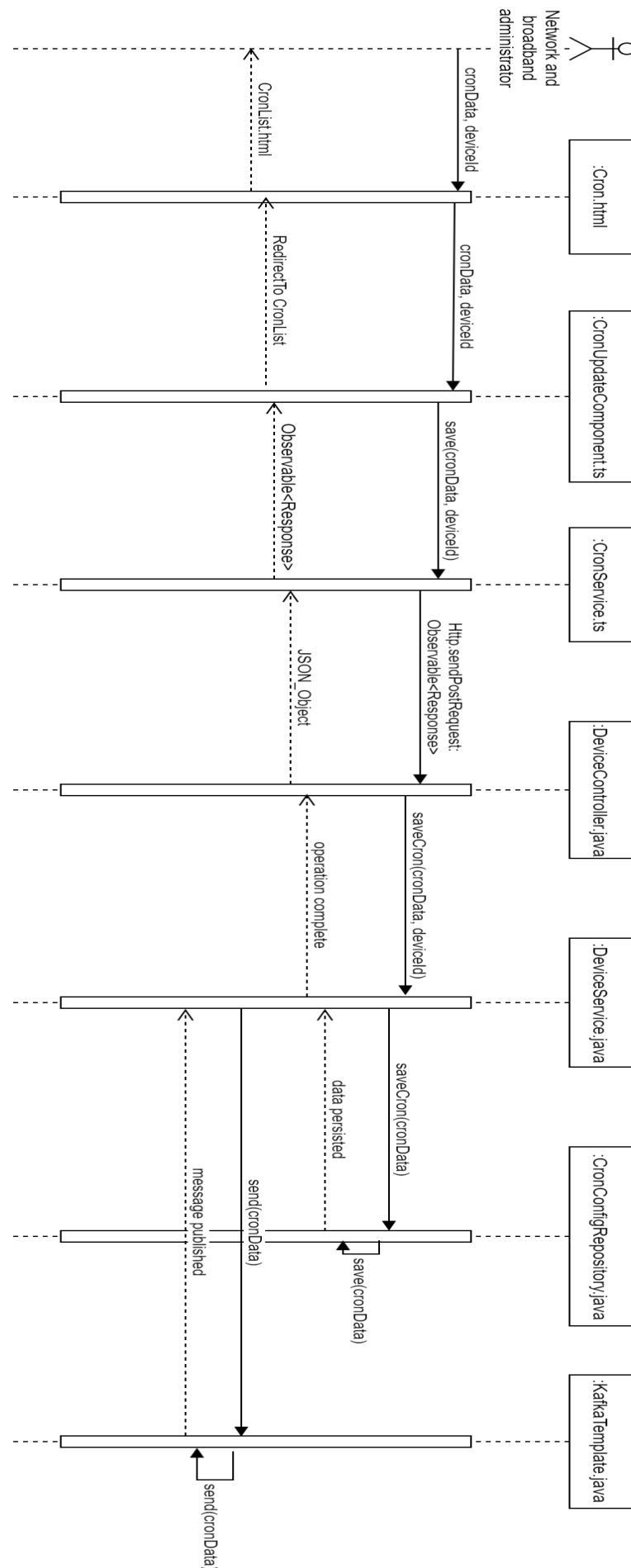


Figure 3.7: Sequence diagram of Cron updating operation

The device configuration consists of the device general information (IP address, device ID, location...) and the jobs running on it. First, the user fills the form of device information, after, he submits this information. Using REST web services we deliver the new configuration from frontend application to backend side. Finally, the backend save the new configuration in our database and send it to the device in question in order to consider the new changes.

For Cron configuration, we follow the same process; the user submits the new configuration, after, this configuration is sent to backend in JSON objects format. Finally, our backend application implements the changes in the database and the device.

3.2.2.4 Metric and test management module

This module describes test management process. This module is about creating, editing, deleting, and checking available tests. The following class diagram (fig.3.8) presents the involved classes and components that build our module.

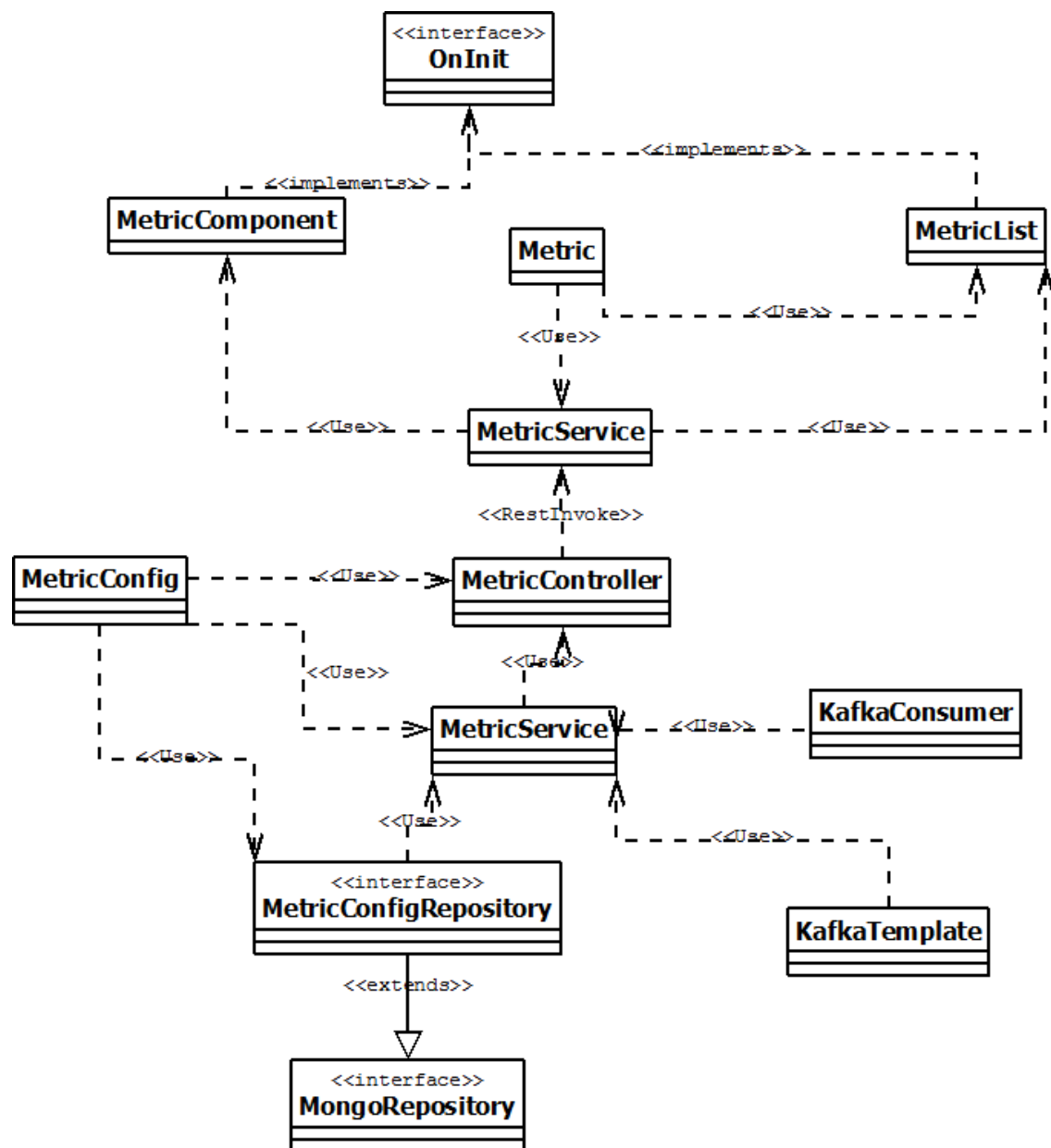


Figure 3.8: Class diagram of the test and metric management module

Then, we have the sequence diagram that describes the interactions between the module's components; this diagram presents the creation and updating tests operations:

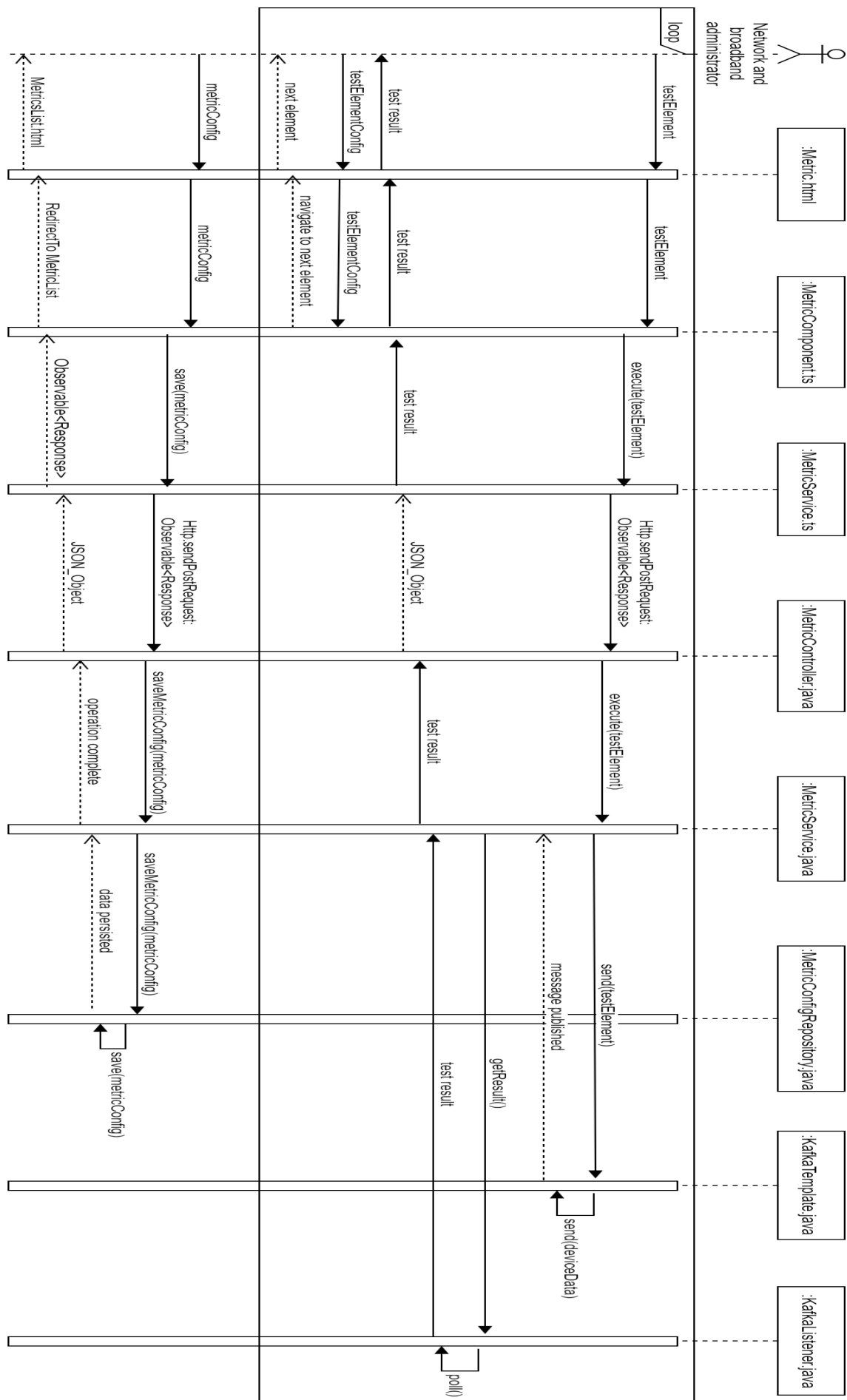


Figure 3.9: Sequence diagram of the metrics and tests creation and updating operations

The metrics and tests configuration consists of describing the general test information (test name, test unit, schedules) and the test elements' configuration. A test element is a command line or even a script. To achieve better performance, each test element is tested directly on the Raspberry board going through the frontend application to the backend application to reach our Kafka cluster, the element is tested, and the results is sent back to Kafka. Finally, the results' data takes back the same path to reach the user interface. The user configures the element with its results. To finish, the user submits the entire test's configuration to the web application.

We must note that the updating and creation operations are the same because we are using MongoDB as a database, if the object is already persisted, MongoDB edits it, and else the object is created.

3.2.2.5 Network monitoring management module

This module describes network monitoring management process. This module is about creating, editing, deleting, and checking available alerts information and constraints.

The following class diagram (fig.3.10) presents the involved classes and components that build the network monitoring module.

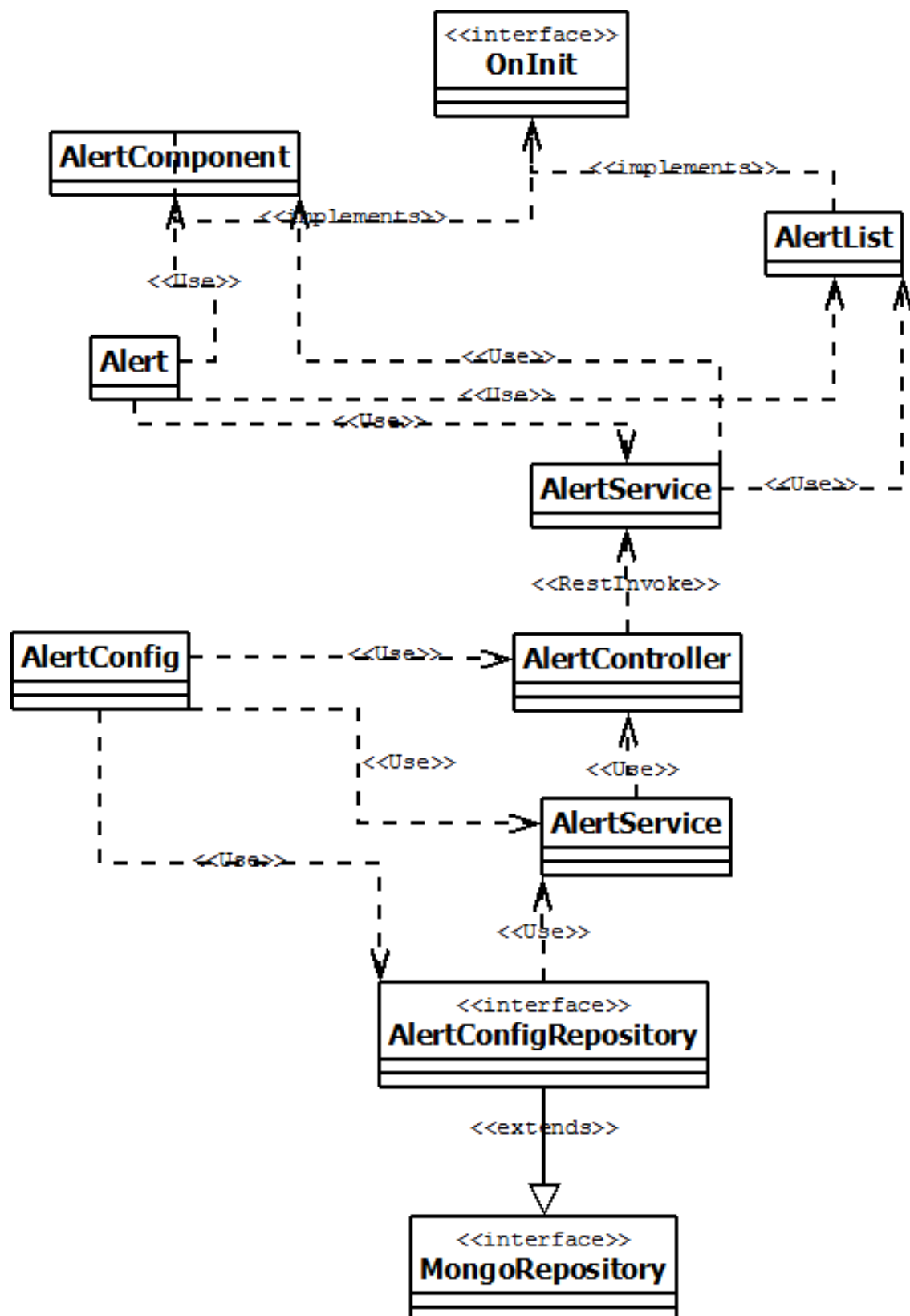


Figure 3.10: Class diagram of the network monitoring management module

Then, we have the sequence diagram that describes the interactions between the module's components; this diagram presents the creation and updating alerts oper-

ations:

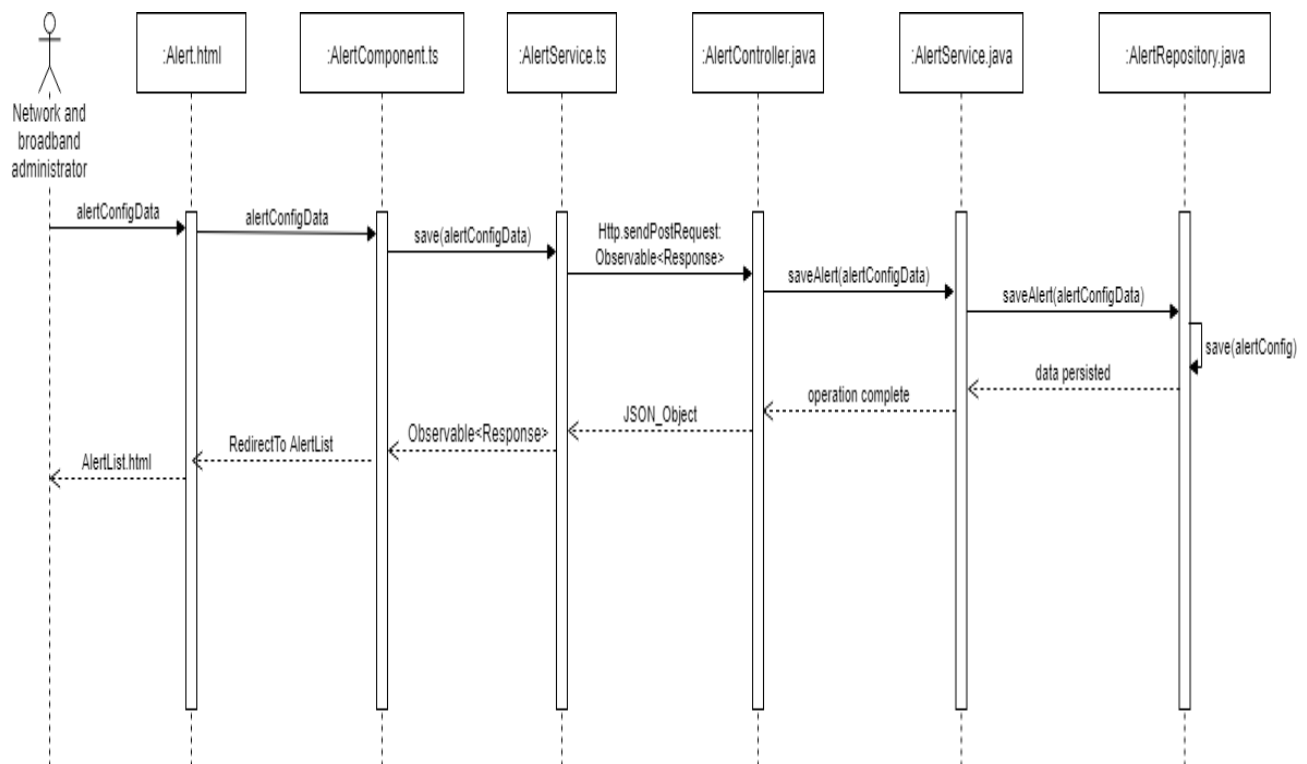


Figure 3.11: Sequence diagram of the alert creation and updating operations

The network monitoring configuration consists of creating customized alerts to help network supervisors obtain better vision of the quality of experience. The network and broadband administrator fill the form that contains alert information (alert title, alert description, involved test, alert constraint) through the graphic user interface. The alert constraint is a logic expression that defines when the alert is triggered. The configuration is sent from the frontend to backend with REST web services. The web application saves the alert configuration. The other details about the network monitoring concerning alert triggering and supervision is explained and clarified in a separate part.

3.2.2.6 Statistic and chart management module

This module describes statistic and chart management process. This module is about creating, editing, deleting, and checking charts and views.

The following class diagram (fig.3.12) presents the involved classes and components that build the module.

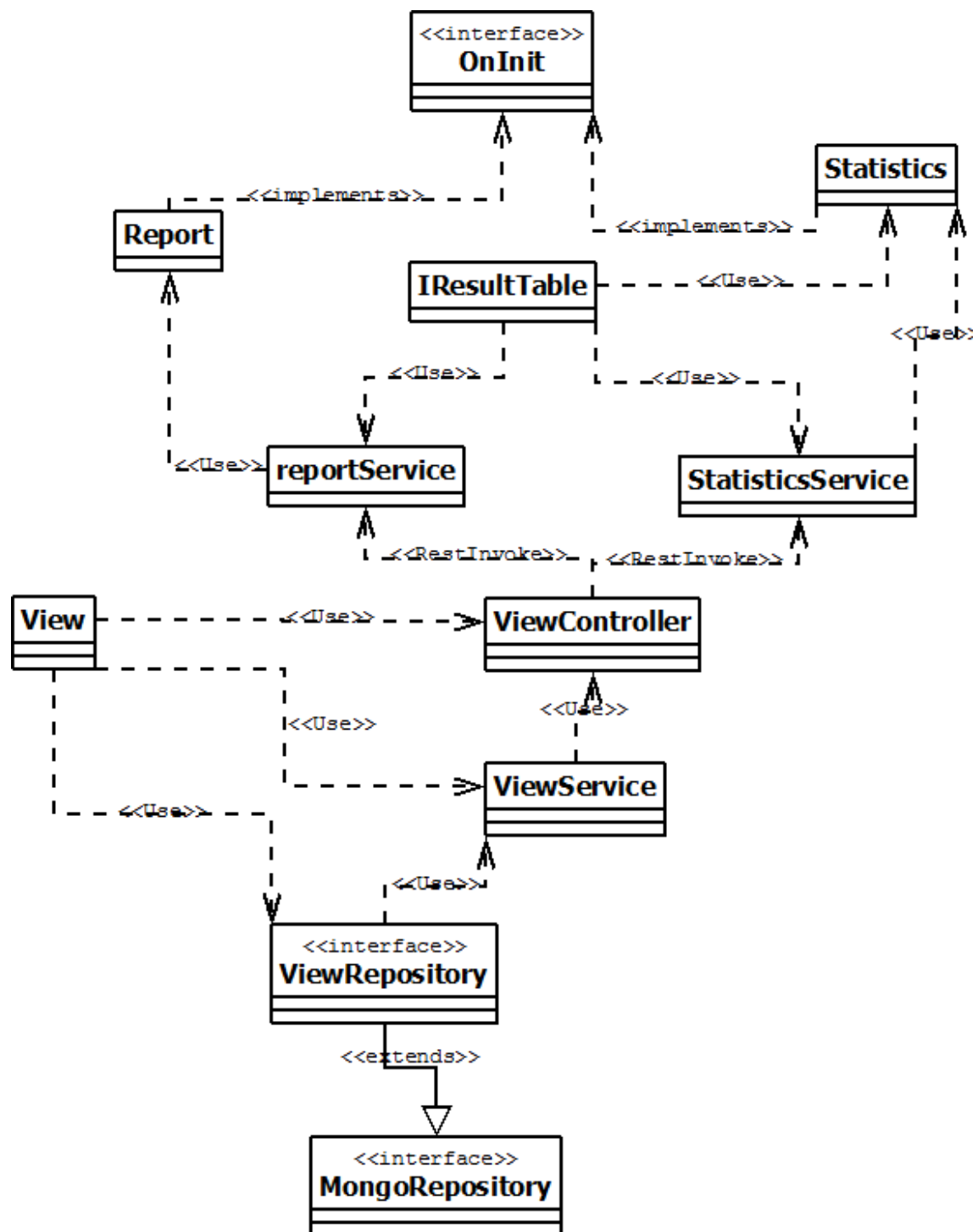


Figure 3.12: Class diagram of the statistic and chart management module

Then, we have the sequence diagram that describes the interactions between the module's components; this diagram presents the view creation operation:

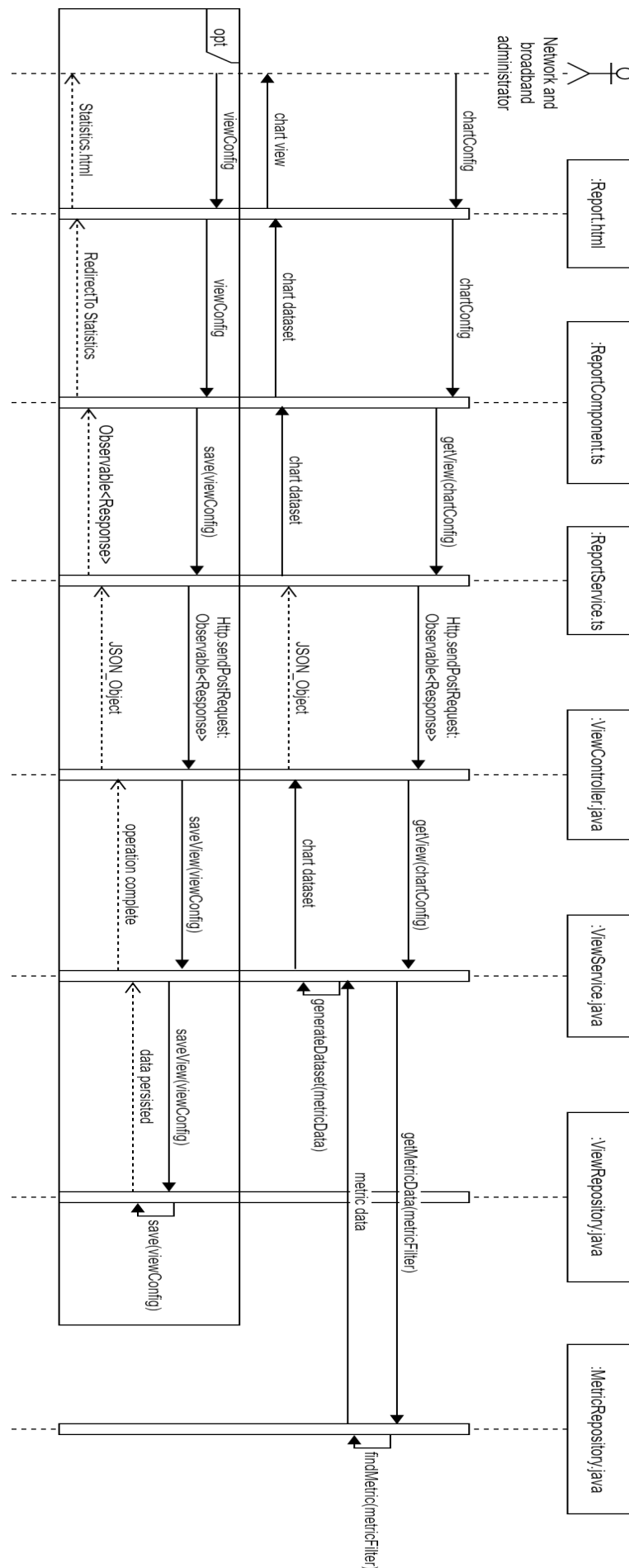


Figure 3.13: Sequence diagram of view creation operation

The diagram below is the sequence diagram for dashboard generation operation.

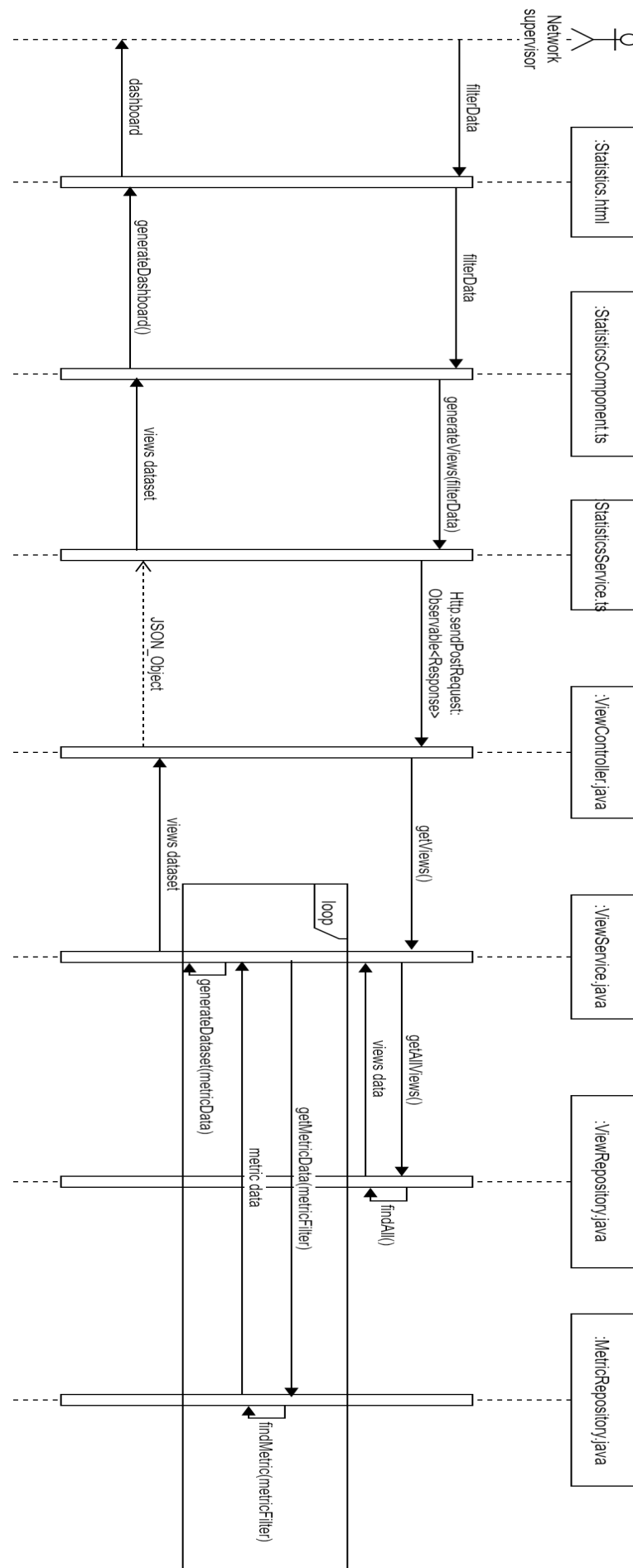


Figure 3.14: Sequence diagram of dashboard generation operation

The statistics and charts configuration consists of creating a model carrying a view configuration that we can use it to generate the same view each time. The statistic and chart management module contains two parts, the first one is the views' configuration and the second one is the dashboard generation.

To achieve creating customized views through the graphic user interface, our Angular application, the user put the configuration he needs to generate the chart. This configuration parameters go through REST web service to the backend controller, then to the service layer. The service layer takes the view configuration as parameter to fetch the involved metrics data, stored in MongoDB, and process the received data in order to create the view dataset. The chart dataset goes through REST web service again to reach the frontend application arriving to the user interface. At this point, the user has the option to save this configuration in order to add this customized view to the dashboard.

The second part is the dashboard generation, when the user navigate to the dashboard, there are three available filters, time filter, location filter, and test filter. A request is sent from the user interface and the Angular component and service to the backend through REST web service carrying the filters inputs. Views are fetched according to the test filter, and then the metrics data is fetched according to location and time filters. The metrics data is processed to generate dataset for each involved view in the dashboard. Finally, the charts datasets goes back to the user interface.

3.2.2.7 Authentication and user management module

This module describes authentication and user account management process. This module is about creating, editing, deleting, and checking available users' information.

The following class diagram (fig.3.15) presents the involved classes and components that build our module.

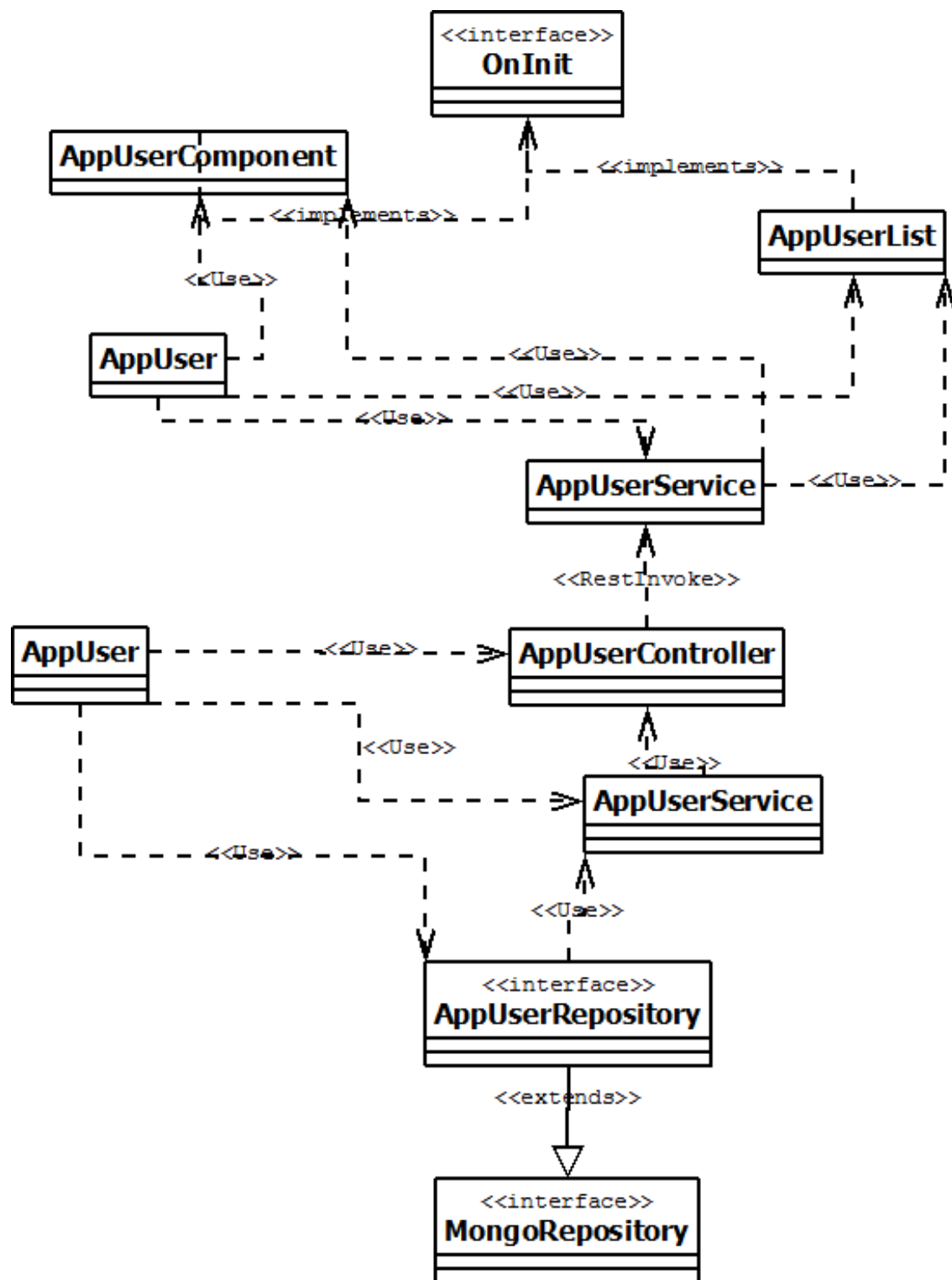


Figure 3.15: Class diagram of the authentication and user management module

Then, we have the sequence diagram that describes the interactions between the module's components; this diagram presents the creation and updating users' information operations:

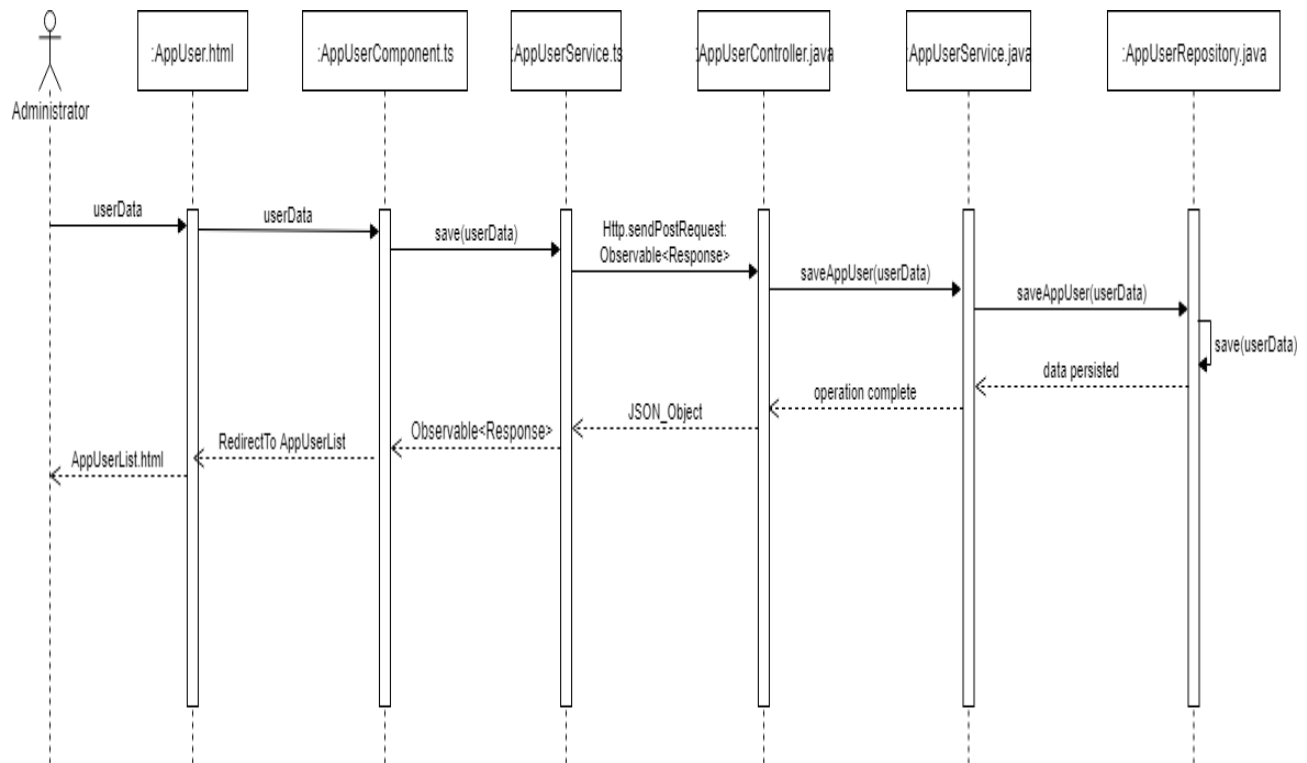


Figure 3.16: Sequence diagram of the user creation and updating operations

The user configuration consists of creating a personalized user account that holds a username, a password, and a role or privilege for the user.

The administrator just needs to fill the form with the user information. After finishing, a request is sent to the backend application for persisting. At this point, the user is able to authenticate.

3.2.3 Broadband supervision and monitoring process

In the previous parts of the report we talked about the features offered by our solution concerning analytics and management of each component, although we did not emphasize the real-time network monitoring and metrics data processing. Also, we need to clarify how the probes are handling the tests and received configurations.

Tests and jobs handling between probes and Kafka

We are using Kafka as a mediator between the server and the probes. For every reboot, devices send their initial information to Kafka, in the counterpart, the devices receives their personalized configuration, this configuration holds within it personalized jobs and tests. The embedded application schedules the jobs using Crontab[5]. There is a listener implemented on the boards in order to wait for configuration

changes. The following (fig.3.17) illustrates the interactions between a probe and Kafka.

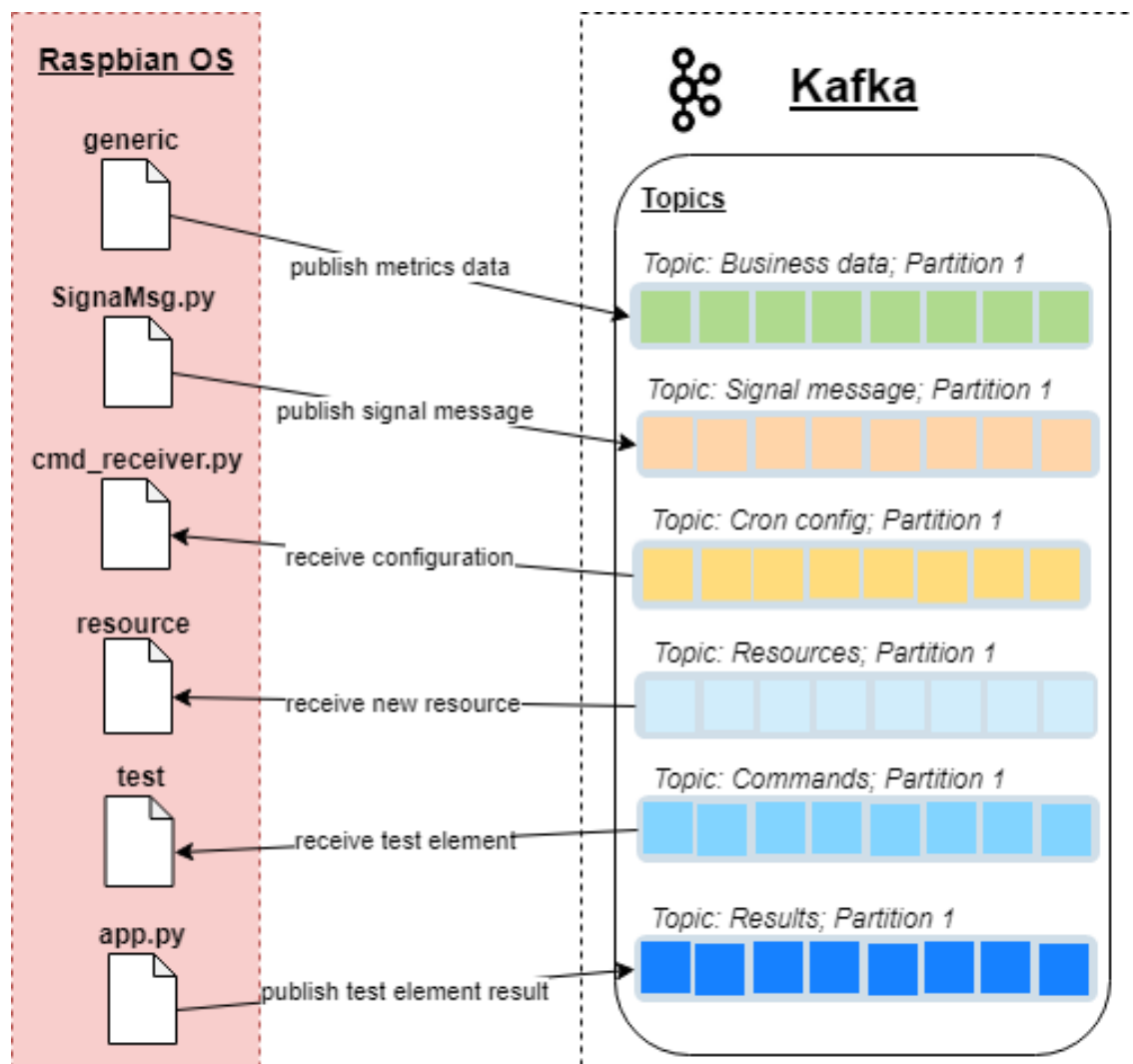


Figure 3.17: Device and Kafka interactions

Real-time processing and network monitoring

As we mentioned earlier, after setting up the test and alert configuration, we have the real-time data processing, this point is the crucial point of our solution. This part can decide the level of the performance of the system. To achieve this task, we chose Kafka Stream.

Kafka Stream is a Kafka client that can handles real-time processing on data coming from a Kafka cluster or broker. Kafka stream subscribes to Business data topic. For each received test result, Kafka Stream brings the alerts' configuration concerning the test in question, after, it tests the alerts constraints on the received data, if a constraint is satisfied, it triggers a specific alert holding a full description.

Another mission for Kafka Stream is to parse the received data in order to extract the

metrics values to store it in MongoDB. However, between data parsing and storage, there is a specific processing for specific tests. In order to explain that, we should know that defining the quality of experience needs to have measurements from different timestamps with specific periods of time. To conclude, a complex calculation is required before storing data to have a better vision of the quality of experience.

Another crucial point in our solution, we need to have a high performance data analytics, MongoDB is an oriented document non-relational database management system that well supports indexing techniques[6]. MongoDB provides advanced querying operations that we used to perform our analytics with customized filters.

3.3 Conclusion

Through this chapter, we described each part of the SMAQ Probes application, its functionalities both separately and when coordinating with other parts of our broadband supervision system. We also explained subsequently the choice of our logical and physical architectures. Concerning the detailed design of SMAQ Probes, we exhibited the class and sequence diagrams. In the next chapter, we present and expose the technologies employed during the process of the creation of SMAQ Probes.

In this chapter, we discuss the process of implementing the different parts of our system. We start by presenting the different tools both software and hardware used to achieve the task in order to complete the implementation process.

4.1 Development environment

4.1.1 Hardware environment

To achieve our project development, we have used a DELL computer with a Windows 7 operating system. The characteristics of the used computer are provided below:

- CPU: intel i5 2.3 Ghz
- RAM: 8 GB
- Hard Disk Drive: 500 GB

For the probes, we have used Raspberry Pi 3 model B+ with an embedded Raspbian operation system. The characteristics of the board are provided below:

- CPU: 64-bit quad-core ARM v8
- RAM: 1 GB
- Memory card: 16 GB

4.1.2 Software environment

In this part, we list the software programs and applications we used throughout the development of our system.

- **Netbeans IDE 8.2:** Netbeans is an integrated development environment developed by Oracle. Netbeans can be used with many programming languages; in our case we used it with Java programming language. Netbeans put in our hand many features to help developers achieving better coding performance.
- **Visual studio code:** Visual studio code is MIT open source licensed software, developed and maintained by Microsoft. It supports many programming languages with many technologies just by integrating plugins within it. We used Visual studio code for Angular development. This software supports different platforms, Linux operation system, Windows, MacOS.
- **Code Blocks IDE:** Code Blocks is an integrated development environment for Fortan, C and C++ programming languages. We used it for C programming.
- **MongoDB:** MongoDB is a non-relational database management system based in documents. It provides many advanced features like indexing and advanced operations algorithms to query on the data. To generate analytics with high performance, we need a database that supports well indexes; also we need a database that provides different and flexible querying techniques like map and reduce algorithm. So the better choice for us was MongoDB.
- **Apache Tomcat web server:** Tomcat is a web server that supports Java web logic. In our case, we used an embedded Tomcat web server.
- **Postman:** Postman allows users to execute and build personalized HTTP requests; to achieve this Postman provides many optional features.
- **Thonny:** Thonny is an integrated development environment for Python programming language running on Raspbian operating system. We used it to develop our Python scripts.
- **Geny:** Geny is an integrated development environment that supports many programming languages. This software is running on Raspbian operating system. We used it to develop C programs on the RaspBerry Pi board.

4.1.3 Frameworks and technologies

In this section, we discuss the technical choices we made to achieve our final product. We start by presenting the programming languages used in the development of the project. Afterwards, we defend our choice for the frameworks we used.

Programming languages

- **Java:** Java is a general purpose oriented object programming language. We used it to develop the backend of our solution.
- **Typescript:** Typescript is a programming language developed and maintained by Microsoft. Typescript is an object oriented programming language. We used it to develop our frontend application with Angular.
- **Python:** Python is a general purpose, high-level programming language. We used it in an embedded environment with the Raspberry Pi boards.
- **C:** C is a procedural and a general purpose programming language. We used it in an embedded environment, with the Raspberry Pi boards.
- **Shell script:** Shell script is a command line interpreter. We used it to install our embedded application.

Frameworks and technologies

- **Spring framework:** Spring is a Java application framework. Spring allows users to create enterprise services with POJO (Plain Old Java Objects). Spring uses dependency injection technique, it also provides many application configuration features.
- **Spring data MongoDB:** Spring data MongoDB is a part of Spring data project which aims to provide a Spring-based APIs (Application Programming Interfaces) for new datastores such as MongoDB. It gives the possibility to execute complex queries on MongoDB, especially with Mongo Template[7].
- **Angular 7:** Angular is an open source framework developed by Google. Angular is used for frontend applications development, it gives the possibility to handle dynamically user interfaces. Angular is written entirely in Javascript, although it uses Typescript as programming language.

- Apache Kafka: Kafka is distributed, fault-tolerant, horizontally scalable, wicked fast streaming platform. It is used for building real-time data pipelines and streaming applications with publish and subscribe technique. We used it as a middleware between widespread probes and the backend application.
- Kafka Stream: Kafka stream is a client library for building applications where the input and output data messages are stored in Kafka cluster or broker. It allows running real-time processing on the data. We used it for real-time processing.
- Google protocol buffer: Protocol buffer is a protocol for structured data serializing, it also known as Protobuf, it supports several programming languages such as Java, Objective-C, Python and C++. We used it to serialize the messages coming from the probes. We used this protocol because it is faster than the ordinary data format like JSON.
- OAuth 2.0: OAuth is a protocol for authorization flows for web applications. It is simple for developers to use. We used it for security issues in our web application.
- Spring Security: Spring Security is a framework to build applications with powerful and highly customizable authentication and access control[8].
- Linux Crontab: Crontab is a tool for job and task scheduling on Linux operating system, in our case Raspbian. We used it to schedule test running.
- MXparser: MXparser is a library for mathematical expressions parsing and evaluating. We used it with application features that need formula parsing such the case of alerting constraints.
- Bootstrap: Bootstrap is an open source frontend library created by Twitter for developing with HTML, CSS and Javascript in order to build responsive, mobile-first projects.

4.2 Achieved work

4.2.1 Authentication and user management

The figure (fig.4.1) shows the users list with their information; this screen is only accessible from an administrator account:

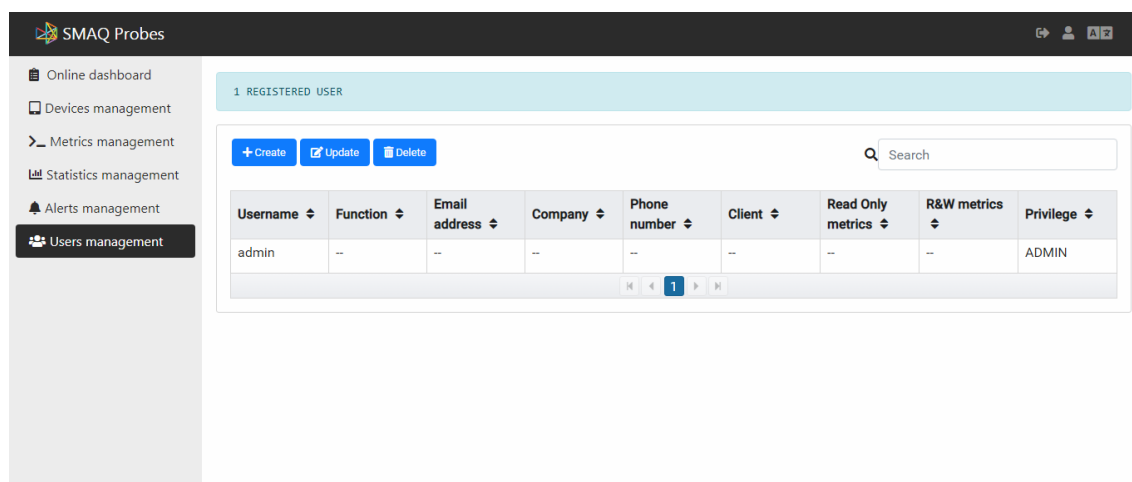


Figure 4.1: User list screen

When installing the application for the first time there is a default administrator account. With this account, the application administrator can create the accounts for the network supervisors and the network and broadband administrators.

With the above screen (fig.4.1), we can access to the user creation form.

This screen (fig.4.2) is used for creating and updating users:

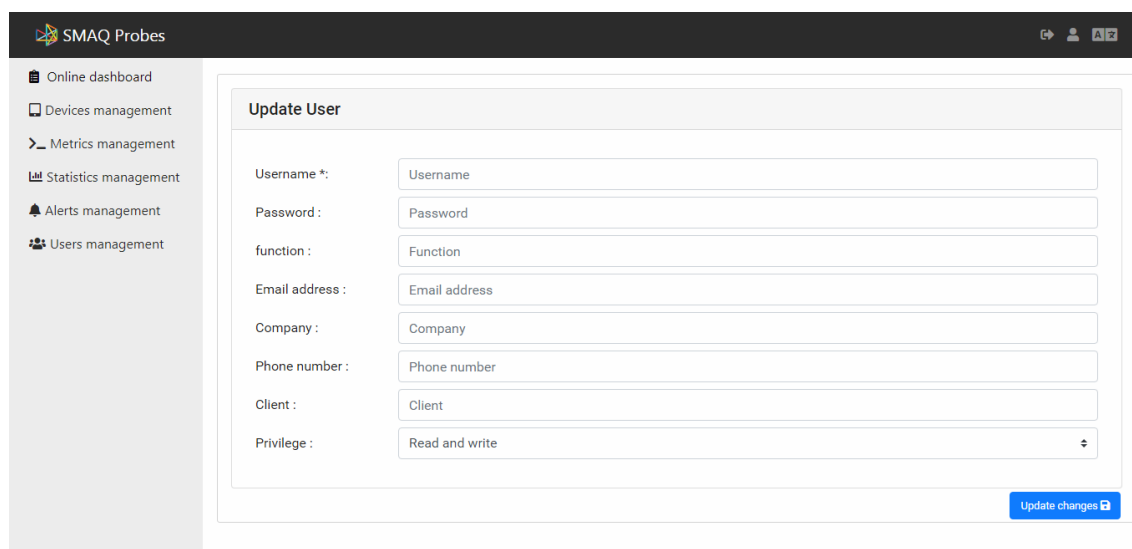


Figure 4.2: User updating screen

This form holds the user information. The username, the password and the privilege are essential for user creation.

If a user is registered successfully, an authentication is required before starting using the application features, the figure below (fig.4.3) presents the authentication screen:

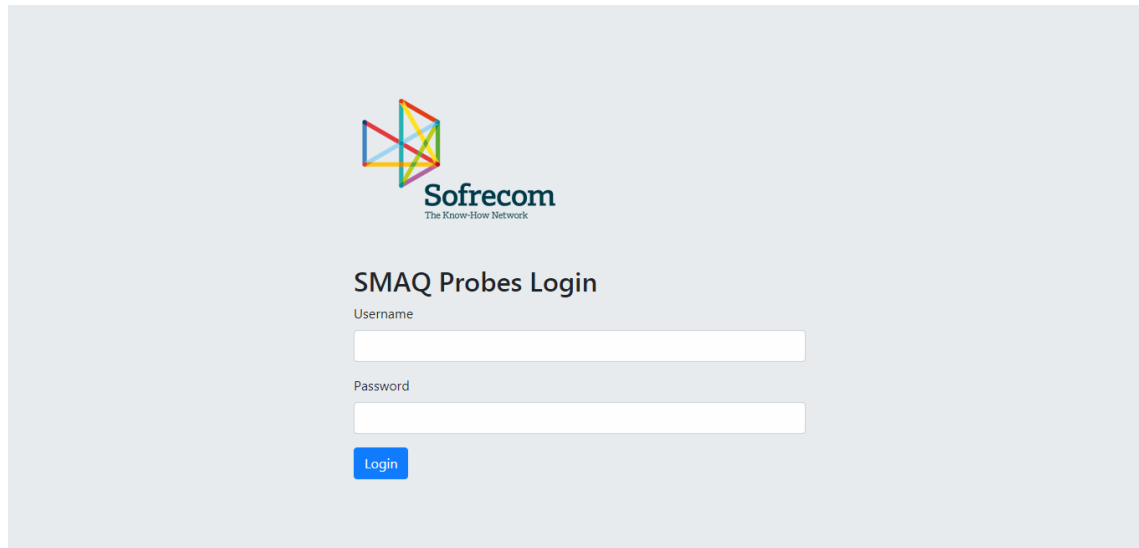


Figure 4.3: Authentication screen

If the user enters the right credentials, the application redirects him to the dashboard screen, else an error message is shown in the login screen. If the access is granted to the user, he can use the application features depending on his role.

4.2.2 Device management

After authentication, to start using all the application features, the probes, in our case Raspberry Pi boards, must be connected in addition to the installation of the embedded application.

When the probe is connected, it sends its information to the application, the information contains the IP address, the device identifier, the device position, the client name and the running tasks on it. If the application receives the device information, the relevant tasks and configuration are sent to it for implementation. The device information is persisted to MongoDB. Now, the devices' configuration can be displayed to the application users as shown in figure (fig.4.4) below:

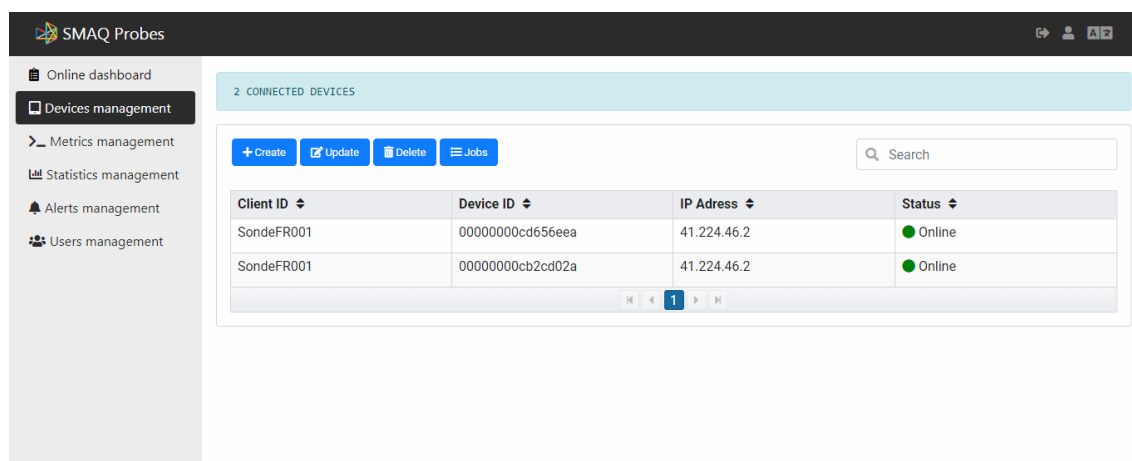


Figure 4.4: Device list screen

With the previous web page, we can navigate to the updating screen of a specified device, the figure below (fig.4.5) presents the device's updating form, this form contains the general information of the probe.

Update Raspberry

ID Raspberry *: 0000000cd656eea

Client ID *: SondeFR001

Base url *: 41.224.46.2

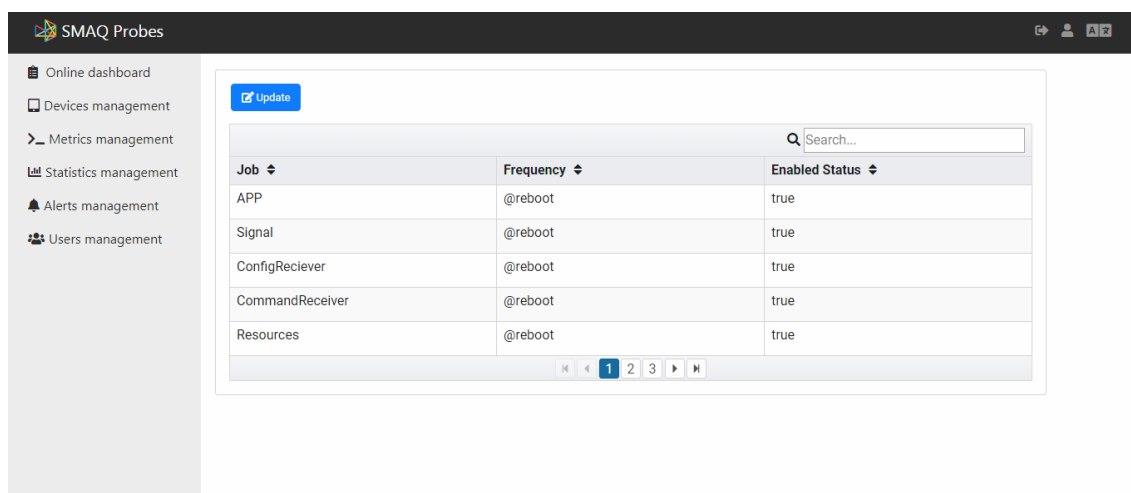
Status *: UP

Submit Cancel

Figure 4.5: Device updating screen

After updating the device information, the new configuration is persisted to MongoDB.

With the device list screen, we can navigate to the running jobs of a specified probe, as shown below in (fig.4.6):

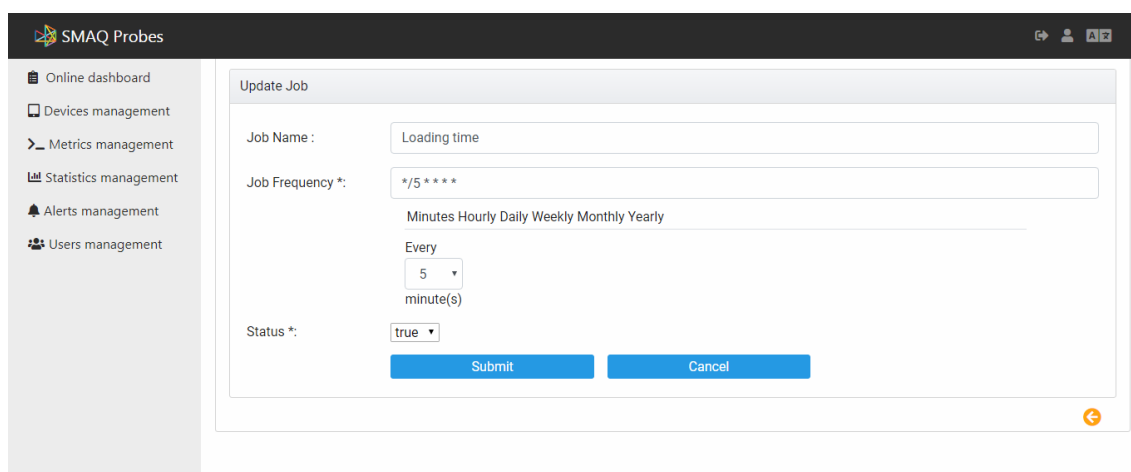


Job	Frequency	Enabled Status
APP	@reboot	true
Signal	@reboot	true
ConfigReceiver	@reboot	true
CommandReceiver	@reboot	true
Resources	@reboot	true

Figure 4.6: Cron list screen

With the previous figure (fig.4.6), we can update the configuration of device jobs. A job configuration is essentially the job status and the job frequency. For the job frequency, we are using the task scheduler Crontab.

The figure below (fig.4.7) presents the job updating screen:



Update Job

Job Name : Loading time

Job Frequency *: */5 ****

Minutes Hourly Daily Weekly Monthly Yearly

Every 5 minute(s)

Status *: true

Submit Cancel

Figure 4.7: Cron updating screen

After submitting this form, the new jobs' configuration is sent to the device in question in order to implement it.

The device jobs include the running tests. The next section is about how to manipulate these tests.

4.2.3 Metric and test management

The metric and test management is the feature that gives the possibility to the network and broadband administrator to manipulate the running tests on the probes.

The figure below (fig.4.8) presents the list of the tests already created and running on the devices:

Name	Unit	Scheduling
Latency	Milliseconds	false
Throughput	MB/s	false
Loading time	Seconds	false
test0	test	false
test1	test	false
test2	test2	false
super_latency	Mb	true

Figure 4.8: Test list screen

Using the previous web page, we can access to the creation screen of the tests. The process of test creation goes through several steps, these steps are illustrated in a wizard as described next.

First, we have the form that contains the general information of the test, its name, the unit that is used for the metrics of the test, and the job scheduling mode. If the job scheduling mode is enabled, each element of the test runs in a specified order. The figure below (fig.4.9) presents the general test information form:

New metric

Name *:

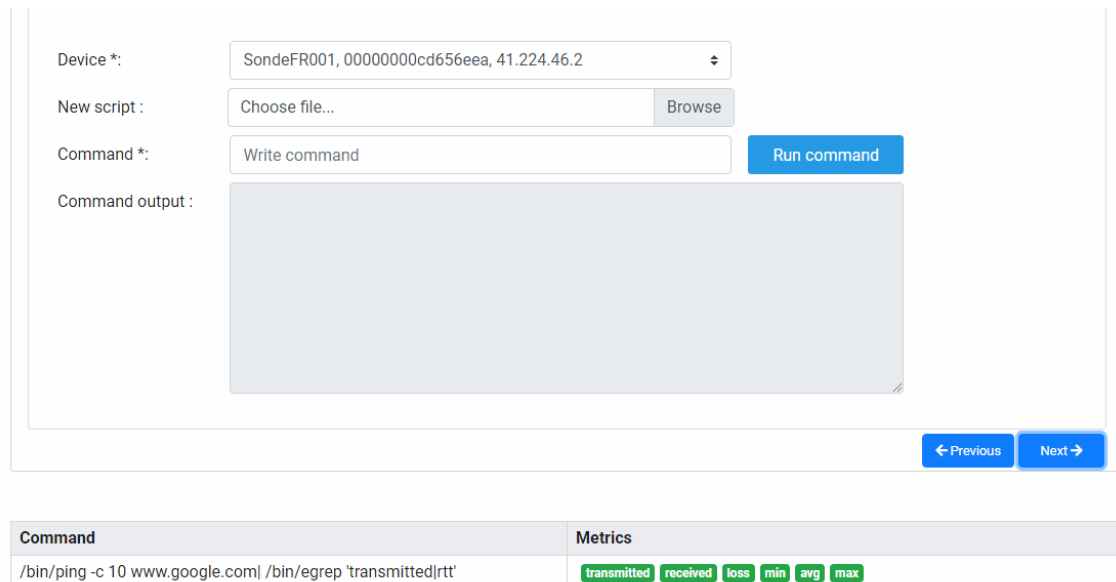
Unit *:

Job scheduling mode *:

Figure 4.9: Test general information form

Now, the user should enter the elements of the test. An element can be a Linux command line that runs an available script or an existing program. To complete this step, we must have at least one connected device, this device is used to test the

elements on it. After entering the command line and a script, in the case if we need one, we run the command on the available device. The test result is shown in the command output field. The network and broadband administrator highlights the needed metric value. Here, our system creates a parsing key that is used to extract the desired results in the process. We repeat this process for all the test elements. The figure below (fig.4.10) shows the form that contains the element information.



Command	Metrics
/bin/ping -c 10 www.google.com /bin/egrep 'transmitted rtt'	transmitted received loss min avg max

Figure 4.10: Test element's configuration form

To improve the use of our application, the network and broadband administrator has the possibility to configure some formulas using the configured metrics. These formulas help us get more relevant results about the QoS. The figure (fig.4.11) presents the form that is used to enter formula's configuration. We have used a drag and drop component to give the network and broadband administrator a better experience with the application.

Figure 4.11: Test formula's configuration form

The last step is the test elements scheduling, this step is only activated if the job scheduling mode is enabled. In this step, we have all the test elements's information displayed in the screen. The network and broadband administrator enters the schedule of each element in seconds. The test elements run on the probe according to the specified order. The figure below (fig.4.12) presents the form in question:

Figure 4.12: Test element scheduling configuration form

Finally, the network and broadband administrator confirms the test configuration. This new configuration is persisted to MongoDB and it is sent to the involved devices in order to be considered. This test is added to the jobs of devices.

4.2.4 Network monitoring management

Until now we have the connected devices' list, and we have the running tests' configuration. In this part, we talk about the network monitoring and alert management. First, we have the figure (fig.4.13) that shows the list of the configured alerts.

submits the configuration details, this configuration is persisted to MongoDB. The alert configuration is fetched in the real-time processing with Kafka Stream, the constraint is tested according to the involved metrics, if the constraint is satisfied, the alert is triggered with the custom description.

4.2.5 Statistics and dashboard management

The following screen (fig.4.15) presents the views and charts configuration:

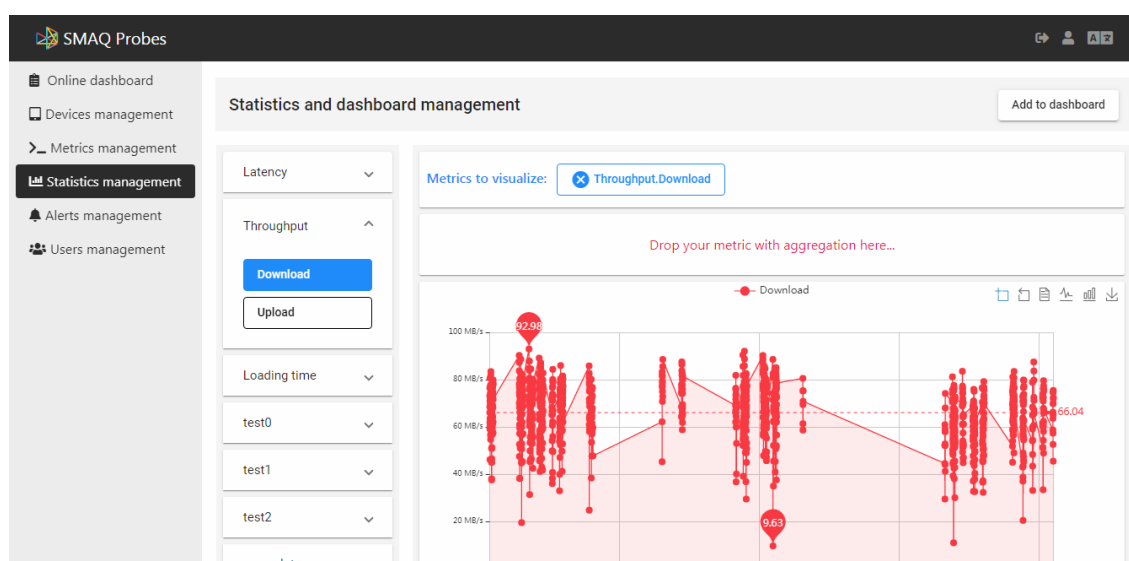


Figure 4.15: Chart customization screen

A chart is configured according to a test configuration. In the left side we have the list of the tests' configuration with its metrics. Also, we have two fields, the first one is titled "Metrics to visualize" and the second one is titled "Metrics with aggregation to visualize". If the network and broadband administrator drops the metric in the first field, the chart shows the metric values by time. If the network and broadband administrator drops the metric in the second field, the chart shows the values according to an aggregation operator, the sum, the mean, the maximum, or the minimum, with a specified period, for instance, the mean of upload speed per hour. If the network and broadband administrator desires to add the chart to the dashboard, he clicks on the "Add to dashboard" button. The configuration of the chart is persisted to MongoDB. So, with this configuration we can generate the view on the dashboard.

The figure (fig.4.16) presents an overview of the dashboard:



Figure 4.16: Online dashboard screen

As we can see in the dashboard, we have the views that we saved their configuration. Also, we have the triggered alerts that we saved their configuration. To generate the different views, we have to use the available filters, the time filter, the zone filter and the test filter.

The following screen (fig.4.17) presents the time filter:

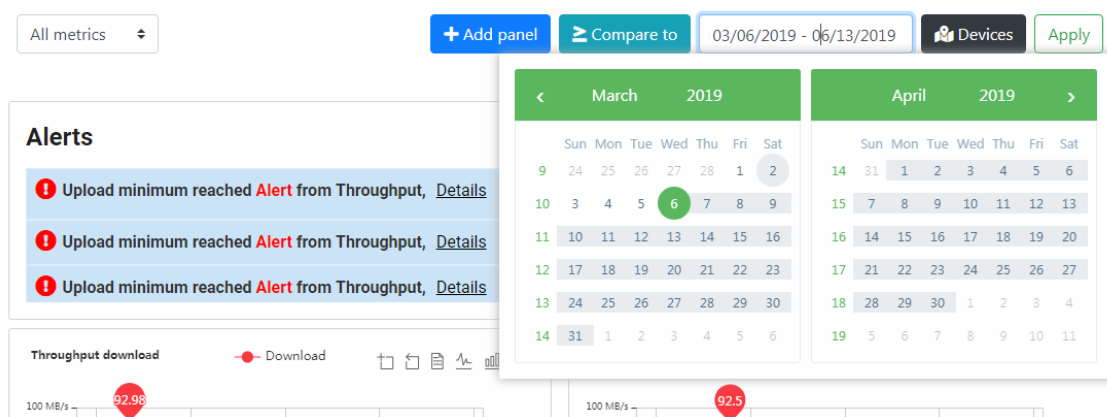


Figure 4.17: Time period filter

This filter specifies the period of time that we used to generate the statistics according to it. We use for this a time range picker. With this filter we can get relevant statistics about the QoS.

The following figure (fig.4.18) presents the zone filter:



Figure 4.18: Device position filter

As mentioned before, we have the information about the devices' positions. Using this information, we can locate the probes on the map. The network supervisor can select a zone area. SMAQ Probes generates the statistics using the devices that are located in the specified area. This filter allows the network supervisor to evaluate the QoS in a specified area.

The following figure (fig.4.19) shows the comparison filters.

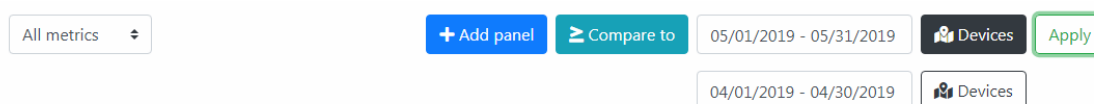


Figure 4.19: Comparison filters

By clicking the “Compare to” button, an additional filters are shown. Using these filters, the system generates charts that includes a comparison between two different results. This feature gives the possibility to the network supervisor to compare between different QoS information, this helps him to improve the network quality of service.

The following screen (fig.4.20) shows a sample of a triggered alert information:

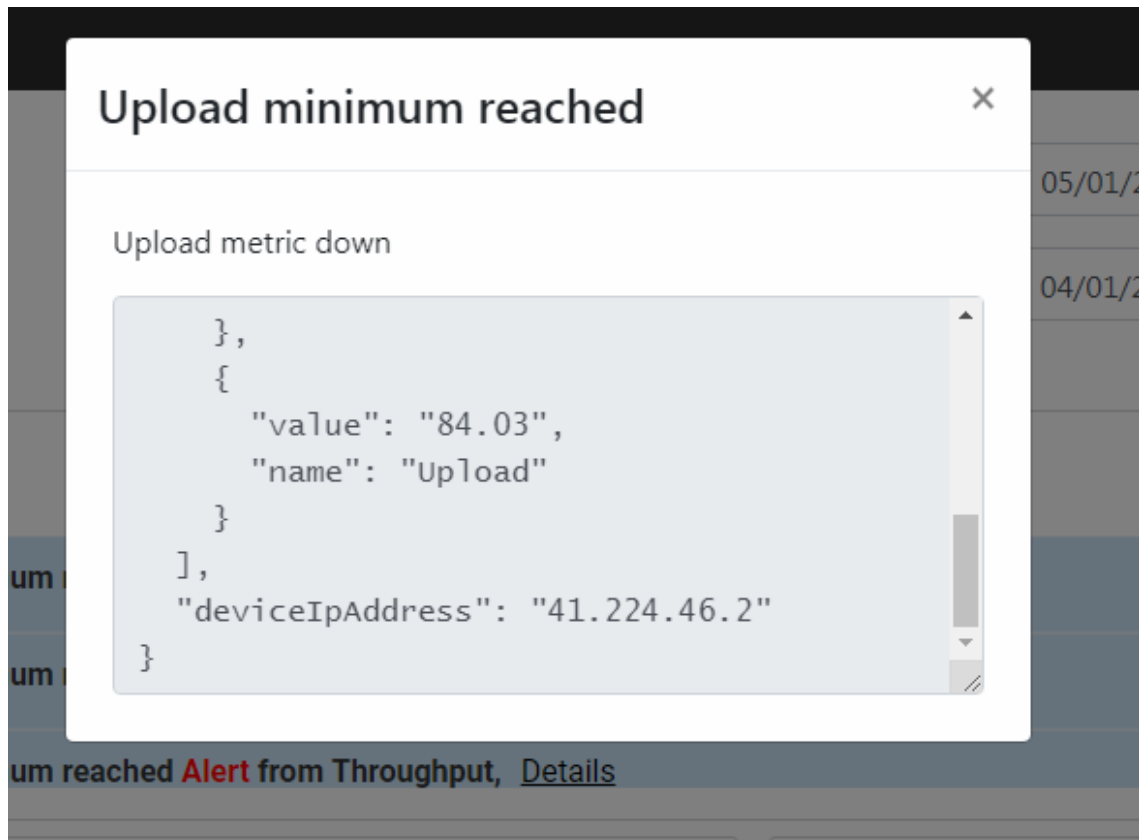


Figure 4.20: Triggered alert detail screen

In the real-time processing, when an alert is triggered, it holds within it the detailed information about the metric value that satisfied the constraint. This feature helps the network supervisor to get quick information about the broadband network problems.

The charts that we are using in the dashboard have several advanced options like data zooming, data visualizing and bar chart support[9]. The following figure (fig.4.21) is a sample. This figure presents the upload speed metrics values during a specified period of time, the upload speed is a metric from the throughput test:

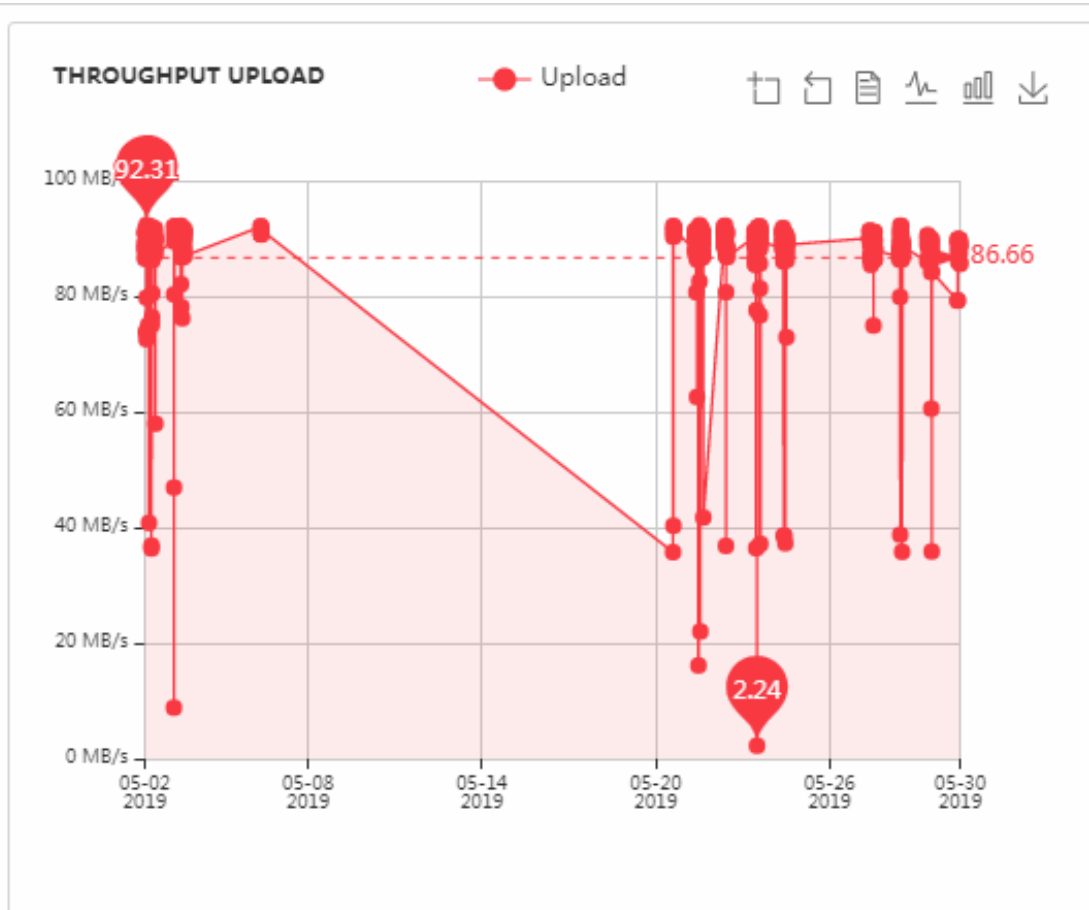


Figure 4.21: Upload speed chart

4.3 Conclusion

During this chapter, we presented the technologies that were chosen to implement SMAQ Probes features and we argued that choice for the different components of the project. We also presented the used frameworks and tools, and we finished by displaying the main features and interfaces from SMAQ Probes application.

General Conclusion and Future Work

In our project, we focused on solving the broadband supervision and monitoring problem by designing and developing a flexible and easy-to-use platform. We designed and developed SMAQ Probes, a web application that helps network supervisors assess the quality of service of their networks. The information about quality of service is provided through data analytics coming from widespread probes. We designed a platform that gives users the flexibility they need to manage their tests and devices online.

For the graphical user interface, we used many advanced options to provide users with better experience using our application.

To assure data security, we implemented many security features, such as secure sockets layer (SSL) implementation between the different components of the solution.

As every project can be improved and expanded, various perspectives are imminent for SMAQ Probes. We can mention the enhancement of the real-time processing component using machine learning algorithms to predict the quality of experience. Some other technologies can be integrated with our platform in order to achieve these enhancements, such as KSQL[10] for Apache Kafka.

Bibliography

- [1] Samknows.com. (2019). SamKnows. [online] Available at: <https://samknows.com/technology/testsS> [Accessed 19 Jun. 2019].
- [2] Docs.confluent.io. (2019). librdkafka: librdkafka documentation. [online] Available at: <https://docs.confluent.io/2.0.0/clients/librdkafka/index.html> [Accessed 19 Jun. 2019].
- [3] Docs.confluent.io. (2019). Documentation — Confluent Platform. [online] Available at: <https://docs.confluent.io/current/streams/index.html>. [Accessed 19 Jun. 2019].
- [4] V7.angular.io. (2019). Angular. [online] Available at: <https://v7.angular.io/docs> [Accessed 19 Jun. 2019].
- [5] PyPI. (2019). python-crontab. [online] Available at: <https://pypi.org/project/python-crontab/> [Accessed 19 Jun. 2019].
- [6] Docs.mongodb.com. (2019). Indexes — MongoDB Manual. [online] Available at: <https://docs.mongodb.com/manual/indexes/> [Accessed 19 Jun. 2019].
- [7] Mark Pollack, J. (2019). Spring Data MongoDB - Reference Documentation. [online] Docs.spring.io. Available at: <https://docs.spring.io/spring-data/mongodb/docs/current/reference/html/> [Accessed 19 Jun. 2019].

-
- [8] Paraschiv, E. (2019). Using JWT with Spring Security OAuth | Baeldung. [online] Baeldung. Available at: <https://www.baeldung.com/spring-security-oauth-jwt> [Accessed 19 Jun. 2019].
- [9] Echarts.apache.org. (2019). ECharts. [online] Available at: <https://echarts.apache.org/en/index.html> [Accessed 19 Jun. 2019].
- [10] Waehner, K. (2019). Machine Learning with Python, Jupyter, KSQL and TensorFlow | Confluent. [online] Confluent. Available at: <https://www.confluent.io/blog/machine-learning-with-python-jupyter-ksql-tensorflow> [Accessed 20 Jun. 2019].