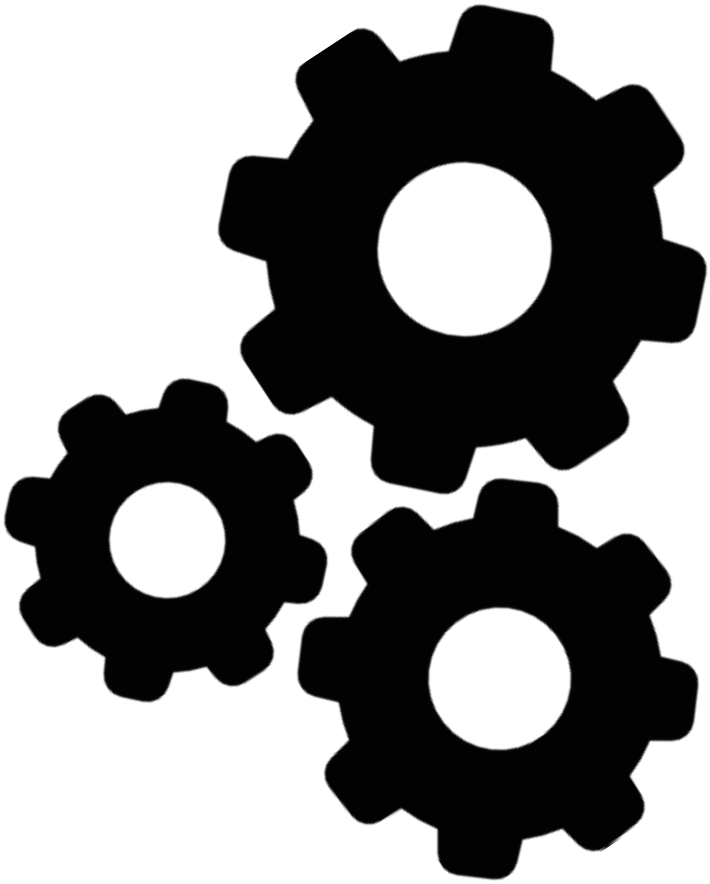
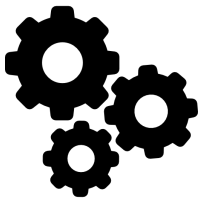
INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES LYON

Software Engineering & UML : Specifications



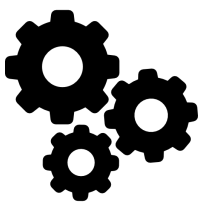
Group : 3IF-3

Pairs : B3320 - B3331



Introduction

In this project we're controlling the interactions between the user and two databases. The database with all the measured information and the database that stocks information created by the interactions. We need to create a terminal-based interface that allows you to call functions, and that shows the data in a readable manner. We also need the data analysis to be as fast as possible and make sure that safety issues are considered.

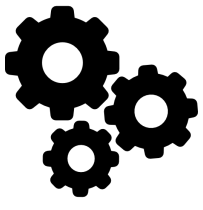


User Requirements Definition

General

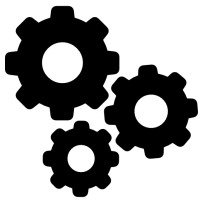
Functionality F1	Authentication (Log In)
Description	Authentication of any type of user
Required Data	Username & password
Precondition	
Postcondition	Access to the functionalities accessible to the role of the user
Output data	(User)
Borderline case	• the couple (username,password) cannot be matched to any existing user in the database
Side effects	
Priority (from 1 to 5)	5

Functionality F2	Log out
Description	Disconnexion of any type of user
Required Data	
Precondition	• User is logged in
Postcondition	The functionalities are no longer accessible.
Output data	
Borderline case	
Side effects	
Priority (from 1 to 5)	5



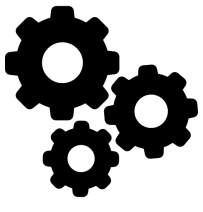
Functionality F3	Detect similarities
Description	Compares the data provided by the specified sensor to all other sensors during a specified period of time and ranks them based on this similarity.
Required Data	(sensor,start,end)
Precondition	• User is logged in
Postcondition	
Output data	An array of sensors ranked by similarity
Borderline case	• Sensor doesn't exists in database • period_end < period_start
Side effects	Awards points to users that possesses sensors which data is used
Priority	3

Functionality F4	Assess air quality at a point
Description	Returns an estimation of the air quality at the specified coordinates and the specified time.
Required Data	(longitude,latitude, time_start, time_end)
Precondition	• User is logged in
Postcondition	
Output data	A value representing air quality
Borderline case	• coordinates are invalid • time_start > time_end
Side effects	Awards points to users that possesses sensors which data is used
Priority	4



Functionality F5	Assess air quality in area
Description	Returns an estimation of the air quality in the area specified by the coordinates of its center and its radius using the data measured in the specified time interval.
Required Data	(longitude,latitude,radius, time_start, time_end)
Precondition	• User is logged in
Postcondition	
Output data	A value representing air quality
Borderline case	• coordinates are invalid
Side effects	Awards points to users that possesses sensors which data is used
Priority	4

Functionality F6	Statistics
Description	Compute several indicators and returns them
Required Data	
Precondition	• User is logged in
Postcondition	
Output data	(rankings, top_5, etc.) where all objects are lists or dictionaries
Borderline case	
Side effects	Awards points to users that possesses sensors which data is used
Priority	2

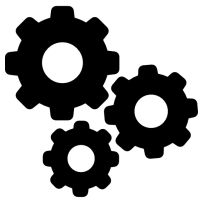


Cleaner owners

Functionality F7	Analysis of cleaner efficiency
Description	Computes several indicators and statistics about the specified cleaner
Required Data	(Cleaner)
Precondition	• User is logged in • User has the role “Cleaner Owner”
Postcondition	
Output data	(level_improvement, cleaned radius,etc.)
Borderline case	• Cleaner does not exist in database
Side effects	Awards points to users that possesses sensors which data is used
Priority	3

Private

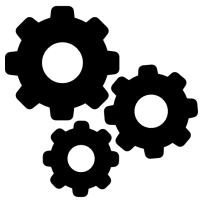
Functionality F8	View reward points
Description	Access to the number of reward points earned by a user
Required Data	
Precondition	• User is logged in
Postcondition	
Output data	(points)
Borderline case	
Side effects	
Priority	1



Government

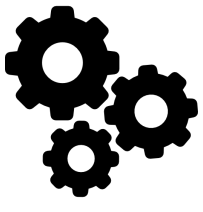
Functionality F9	Analysis of sensors
Description	Access to all the measurements of a specific sensor
Required Data	(sensor)
Precondition	• User is logged in • User has the role “Government”
Postcondition	
Output data	(measurements)
Borderline case	Sensor doesn’t exist in database
Side effects	Awards points to the user that possesses the sensor
Priority	4

Functionality F10	Assess algorithm performance
Description	Executes the algorithm(s) and outputs its (their) runtime(s) in milliseconds.
Required Data	
Precondition	• User is logged in
Postcondition	
Output data	(runtimes)
Borderline case	
Side effects	
Priority	2

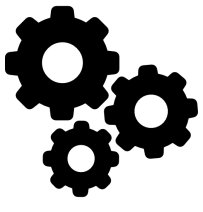


Functionality F11	Flagging sensor
Description	Flags all measurements of a sensor, and marks it as defective
Required Data	(sensor)
Precondition	• User is logged in • User has the role “Government”
Postcondition	
Output data	
Borderline case	• Sensor doesn’t exist in database
Side effects	Measurements are marked as non reliable
Priority	4

Functionality F12	Flagging malicious user
Description	Flags all measurements of all sensors of a user, and marks him/her as malicious
Required Data	(malicious_user)
Precondition	• User is logged in • User has the role “Government”
Postcondition	
Output data	
Borderline case	• User doesn’t exist in database
Side effects	Measurements are marked as non reliable, User is marked as malicious
Priority	4



Functionality F13	Compare sensor
Description	Compares the specified sensor to all other sensors and returns the results of the comparison with each sensor
Required Data	(sensor)
Precondition	• User is logged in • User has the role “Government”
Postcondition	
Output data	(results)
Borderline case	• sensor does not exist in database
Side effects	Awards points to users that possesses sensors which data is used
Priority	3



System requirements specification

This should describe the functional and nonfunctional requirements in more detail. If necessary, further detail may also be added to the nonfunctional requirements. Interfaces to other systems may be defined.

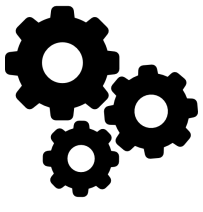
The description of the project allowed us to define the non-functional requirements of the system that are listed below.

Performance

The government agency wants the algorithms of the system to be efficient. Thus, the time and space complexity should be as good as possible and should also be visible. The user should be able to measure the performance of those algorithms, for example by displaying the duration of execution of an algorithm in milliseconds.

Security

The functionalities provided by the app should be limited for each user type. The DataBase access should be safe.



User Case Diagram

Analysis of security risks

Attackers

- People who want to sell the data for the air purifiers to competitors to disrupt the market
- Leak the air quality data of the government agency to undermine the integrity of the government in the eyes of the people
- User who want to falsify or spam data to increase the rewards they can reap from the system

Assets

- Air quality data that has been influenced by the air purifiers
- Location of the air purifiers and sensors (hardware which can be stolen or tampered)
- Personal information of the users

Vulnerabilities

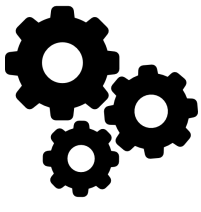
- Passwords that have low password entropy
- Ability to check for falsified or spam data against secure and reliable data sources
- Data submitted by users is transmitted and stored without proper encryption

Type of attacks

- Usage of password cracking tools to access user accounts
- The attacker enters invalid or falsified data
- The attacker intercepts the communication to obtain the data
- The attacker uses malicious code: Virus, Worm, Trojan to paralyse the system to prevent access to the data
- Man-in-the-middle attack
- Denial-of-service attack

Risk & Impact

- Attacker get access to air quality data (medium)
- Attacker get access to sensor or cleaner information (low)
- Attacker get access to personal data (high)
- Attacker pollutes the data (medium)
- Attacker delete the data (high)
- Attacker disables the service (low)



Countermeasures

- Back-up the data to a standalone server
- Anonymise the data submissions by users while also using encryption so as to prevent attackers from gaining full access to user information
- Use 2 step verification for logins so as to prevent password cracking
- Only allow strong passwords
- CAPTCHA for too many requests
- Cross-referencing of submitted data so as to prevent falsification

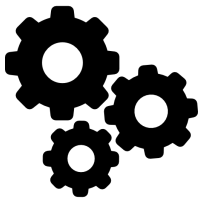
Evaluation of Assets

Asset	Security value	Financial value	Business impact	Overall
Air quality data	3	2	3	3
Sensor & cleaner information	3	4	4	4
Personal information	5	3	2	3

Evaluation of Risks per asset

Air quality data

Threat action (frequency)	Vulnerability (severity)
Natural • Purifier or sensor malfunction or network error causing corrupted data (4)	• Lack of protection from corrupted data (4)
Professional Criminals • Data theft (3) • Data falsification (2) • Data deletion (2)	• Lack of data protection (5) • Lack of data validation (4) • Lack of backup (6)
Users • Data entry errors (5) • Usage falsification (5)	• No cross-referencing of data entered (2) • Misrecording of data access (3)



$$Risk(A) = \left(\sum_{Threat \in A} (Threat * Vulnerability) \right) * Asset\,Value(A)$$
$$Risk = (4 * 4 + 3 * 5 + 2 * 4 + 2 * 6 + 5 * 2 + 5 * 3) * 3$$
$$Risk = 228$$

Sensor & cleaner information

Threat action (frequency)	Vulnerability (severity)
Professional Criminals • Data theft (3) • Data falsification (2) • Data deletion (2)	• Lack of data protection (5) • Lack of data validation (4) • Lack of backup (6)

$$Risk = (3 * 5 + 2 * 4 + 2 * 6) * 4$$
$$Risk = 140$$

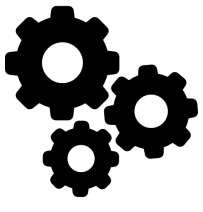
Personal information

Threat action (frequency)	Vulnerability (severity)
Professional Criminals • Data theft (3) • Data falsification (2) • Data deletion (2)	• Lack of data protection (5) • Lack of data validation (4) • Lack of backup (6)
Users • Data entry errors (5)	• No cross-referencing of data entered (2)

$$Risk = (3 * 5 + 2 * 4 + 2 * 6 + 5 * 2) * 3$$
$$Risk = 135$$

Total evaluation of risks

Asset	Risk value
Air quality data	228
Sensor / cleaner information	140
Personal information	135



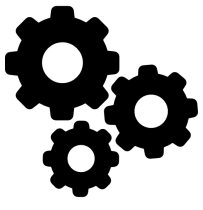
Validation tests

Validity Tests

- Test that the data can be accessed from the interface without missing data
- Data is sorted and filtered based on the conditions proposed by the user
- The program is able to count the number of times each data entry has been used
- The program is able to conclude the air quality based on the data
- The program is able to cross-reference the data to surrounding sensor data to check reliability

Fault tests

- The program is able to detect anomalous data by cross referencing based on the coordinates of the sensors
- The program is able to detect repeated entries in the dataset
- The program is able to detect that the type of user and hence only grant access to the necessary level



User manual

The format is :

```
functionName {option1/option2} [variable]
```

The {/} and [ ] are just to help in reading clarity and shouldn't be typed in.

Log in

```
LogIn [Username] [Password]
```

`Username` and `password` are two strings with no breaks.
If the username and password is registered you'll receive a confirmation of connection otherwise you'll receive an error message.

Log out

```
LogOut
```

Disconnects the user.

Analyse the sensor's data

```
AnalyseSensor [SensorId]
```

`SensorId` : a string that will have a corresponding sensor (the general format is Sensor followed by a number).
If a sensor with that Id exists the sensor's data is shown. Otherwise you'll receive an error message.

Mark user

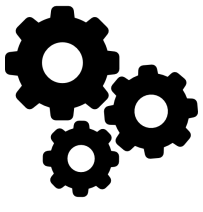
```
MarkUser [UserId]
```

`UserId` is a string that will have a corresponding sensor (the general format is User followed by a number). If a user has that Id then they will be marked as untrustworthy. Otherwise an error will be sent.

Compare all sensors

```
CompareSensors [SensorId]
```

`SensorId` : a string that will have a corresponding sensor (the general format is Sensor followed by a number).
If a sensor with that Id exists a list of sensors is posted ranked by similarity to the chosen sensor. Otherwise an error message is sent.



Mean air quality in an area

```
MeanAirQuality {-instant/-period} [lat] [lon] [radius] [date1] [date2]
```

`lat` : a float that gives the latitude of the center point, it has to be within the [-180°;180°] interval.
`lon` : a float that gives the longitude of the center point, it has to be within the [-180°;180°] interval.
`radius` : a float in meters that gives the radius in meters of the circle used.
`-instant` : an option that calculates the mean air quality at a given instant. `date1` is a date that gives the instant, `date2` is unused.
`-period` : an option that calculates the mean air quality for a given period. `date1` is a date for the start of the time period, `date2` is a date for the end of the time period. `date2` cannot be before `date1`.
Returns a float if the call to the function is correct (i.e. if types are respected and values are coherent).

Air quality at one point

```
PonctualAirQuality [lat] [lon] [instant]
```

`lat` is a float that gives the latitude of the center point, it has to be within the [-180°;180°] interval.
`lon` is a float that gives the longitude of the center point, it has to be within the [-180°;180°] interval.
`instant` : a date at which the air quality shall be measured
Returns a float if the call to the function is correct (if types are respected and values are coherent).

Study air cleaner

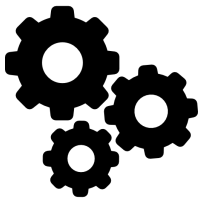
```
StudyCleaner [CleanerId]
```

`CleanerId` is a string that will have a corresponding sensor (the general format is Cleaner followed by a number).
If a cleaner with that Id exists the sensor's data is shown. Otherwise an error message is sent.

Check private data

```
CheckData
```

Lists all private sensors ranked by reliability estimates.



Get points

Points

Returns the number of points earned by the private user.