* **[G1]** Provide a form of unique identification (registration/login) of all clients of the services.

(Requirements **[R1], [R2], [R3]**)

* + LoginServiceImpl
  + SigninServiceImpl
* **[G1.1]** Provide a form of unique identification (registration/login) of all users using the services.

(Requirement **[R4]**)

* + LoginServiceImpl
  + SigninServiceImpl
* **[G1.2]** Provide a form of unique identification (registration/login) of all third parties using the services

(Requirement **[R5]**)

* + LoginServiceImpl
  + SigninServiceImpl
* **[G2]** Allow the user to avoid being associated to his data without his permission. (Requirements **[R6], [R7], [R8], [R9]**)
  + RequestManager
  + RequestServiceImpl
  + DatabaseService(Impl?)
  + DataElbaorationServiceImpl
  + DataCollectionService?
  + MarketingComponent?
  + Router
* **[G3]** Allow third parties to request access to data of some specific individuals. (Requirements **[R6], [R10]**)
  + DataReadingServiceImpl
  + DataCollectionServiceImpl
  + RequestManager
  + RequestServiceImpl
  + DatabaseService
  + DataElbaorationServiceImpl
  + Router
* **[G4]** Allow third parties to request access to anonymized data of groups of individuals.

(Requirements **[R9], [R11], [R12]**)

* + DataReadingServiceImpl
  + DataCollectionServiceImpl
  + RequestManager
  + RequestServiceImpl
  + DatabaseService
  + DataElbaorationServiceImpl
  + Router

* **[G5]** Allow third parties to subscribe to new data.

(Requirements **[R6], [R7], [R8], [R9]**)

* + DataReadingServiceImpl
  + DataCollectionServiceImpl
  + RequestManager
  + RequestServiceImpl
  + DatabaseService
  + DataElaborationServiceImpl
  + NewDataSubcriptionServiceImpl
  + Router
* **[G6]** Allow third parties to obtain the most adapt data for their needs. (Requirements **[R15], [R16], [R17]**)
  + RequestManager
  + DataElaborationServiceImpl
  + RequestServiceImpl
  + DatabaseService
  + Router
* **[G7]** Third parties are alerted, whenever a user is in danger of life (personalized), the application is working properly and there is internet connection.

(Requirements **[R18], [R19]**)

* + DataReadingServiceImpl
  + EmergencyServiceImpl
  + ThresholdsServiceImpl
  + Router
* **[G8]** If the user’s health status is not clear due to malfunctions, an Emergency number is alerted within an hour.

(Requirements **[R4], [R20], [R21]**)

* + DataReadingServiceImpl
  + MalfunctionServiceImpl
  + EmergencyServiceImpl
  + Router

* **[G9]** The user can temporarely suspend AutomatedSos in special cases (non-intrusive).

(Requirements **[R22]**)

* + ThresholdsService
  + UserStatusServiceImpl

SEQUENCE DIAGRAMS DESCRIPTIONS

1. Create Data

This diagram shows the creation and the storing in the database of a single data.

The components mainly involved in this process are DataReadingService and DataCollectionService. The first receives the information from the MarketingComponent and creates a data based on the given parameters.

Then it adds the data to the data’s list of the user and when the data is saved in user, is deleted from DataReadingService.

Then DataReadingService asks the DatCollectionService to store data in the database, by passing the user id as a parameter. So DataCollectionService asks the user to send the data and once received, sends it to DataBaseService, in order to store it. Then it sends an asynchronous request of new data.

In this diagram is modelled the creation of a single data, but actually DataReadingService receives data from the device continuously, then creates new data objects and sends them to the user. Also DataCollectionService always asks for new data, picks them from the user and sends them to the DatabaseService.

1. Emergency Handling

This diagram shows how an emergency is handled by the application. the components most involved in this operation are EmergencyService and ThresholdService.

We supposed that ThresholsService uses a State Pattern or whatever it is characterized by two different states: on and off, set according to the user status (see UserStatus sequence diagram).

So DataReadingService asks ThresholdService to compare the data with the thresholds every time it received data from the device, then the ThresholdService makes different operations depending on the State in which it is situated.

If its State is “OFF”, it simply returned the call, without doing anything.

If its State is “ON”, it compares the data received from DataReadingService. If the value is okay, nothing happens, otherwise the collected data are sent to EmergencyService.

Then EmergencyService creates an alert containing all the data received from ThresholdService, which is forwarded to all the Third Parties.

When a third party takes charge of the emergency, EmergencyService edits the alarm status accordingly.

1. Make Group Request

This diagram shows the procedure to create a request for a group of data by a third party. Shows the interaction between the different requests and data management services that are provided by the application, for this is shown only the case in which there isn’t request for new data. (this alternative is shown in the “New Data Subcription Diagram”).

The first step passes from the RequestManager, which creates the grouprequest with the parameters received and forwards it to the RequestService. It sends the request to the DataElaborationService.

The DataRequestService makes a query to the database with the information contained in the request and creates a datapool object with tuples received as answer. Then it sends the datapool to the RequestService.

If the data is properly anonymized (it involves more than 1000 users) and there are no statistical tools in the parameters of the request, the RequestService sends it to the RequestManager , that forwards the data to the costumer app of the Third Party. Otherwise the RequestService ask the DataElaborationService to elaborate the data and receives a new datapool, containing also the tools.

If the data cannot be anonymized, the RequestService refuses the request as the RequestManager.

1. Subcribe New Data

This diagram shows the process to follow to subscribe to new data. It then shows the case where in the request for a group of data is subscribed to new data. ( Is the alternative [new\_data==TRUE] of the “Make Group Request” diagram).

In this case then, after creating the request, the RequestManager notifies the NewDataSubscriptionService (and not the RequestService) that registers itself as observer of the users that match the request parameters.

Then starts a timeout and there are two different options: the timeout expired without anything to happen or the NewDataSuptionService receives a notify.

The first option means that there are no new data for the subscription and so the request’s refuse is sent to the costumer app.

The second option means that there were updates about the data required. Indeed, the notify is sent to NewDataSubscriptionService only when the DataCollectorService asks to the user data to update the database. In this case the data are stored on the Databae, the RequestManager is notified that data is ready and the process continues as in a request for a group of data until the datapool does not arrive to the costumer app. (see “Make Group Request” diagram).

1. Make Individual Request

This diagram shows the data of a specific user request management. We decided to model only the case where the client allows access to the data because more meaningful for the interaction between components. (In the other case simply a rejection message is sent in response to the request.)

As for the group request, is the RequestManager that creates the individualrequest with the parameters sent from the third party costumer app. Then the RequestManager forwards the request to the RequestService, which sends it to the client costumer app (MarketingComponent). When the user gives his/her permission to access data, the RequestManager can escalate to RequestService who forwards it to the DataElaborationService that sends the query to the DatabaseService.

Then DataElaborationService creates datapool that returns back up the costumer app and as for requests for group may be present or not statistical tools.

1. Malfunction Detection

This diagram shows the management of a malfunction: the component mainly involved is MalfunctionService.

If DataReadingService does not receive data for some time (2 secondi???), it reports the malfunction to MalfunctionService, sending the userID and the timestamp of the last data received.

Once received the notice, the MalfunctionService creates a malfunction with the information received and forwards it to the costumer app, then there are three possible cases.

In the first case the client responds that everything is ok, so nothing more happens.

In the second case the client responds that an emergency is occurred , so the MalfunctionService sends an alert to EmergencyService, without information.

in the third case does not get any response, so the MalfunctionService sends a message to the emergency number.