CO2 Sensor Application

Table of Contents

[1. Prerequisite: 1](#_Toc15833455)

[2. Building: 1](#_Toc15833456)

[3. Configuration: 1](#_Toc15833457)

[4. Output Artifacts: 1](#_Toc15833458)

[5. Design Diagram 2](#_Toc15833459)

[a. Module Dependency Diagram 2](#_Toc15833460)

[b. co2-sensor-domain Design Diagram 2](#_Toc15833461)

[c. Co2-sensor-worker Design Diagram 3](#_Toc15833462)

[d. Co2-sensor-server Design Diagram 3](#_Toc15833463)

[6. Api Design: 4](#_Toc15833464)

[a. Collect Sensor Measurement 4](#_Toc15833465)

[b. Sensor Status Api: 5](#_Toc15833466)

[c. Listing Alerts 5](#_Toc15833467)

[d. Sensor metrics 6](#_Toc15833468)

[7. Database Design: 7](#_Toc15833469)

## Prerequisite:

**Software**

|  |  |
| --- | --- |
| Software | Version |
| OS | Mac OS Mojave |
| Java | 8 or above |
| Mongo DB | v4.0.3 |
| RabbitMQ | 3.7.12 |
| Maven | Apache Maven 3.6.0 |
| Application Server | Tomcat 9.0 |

**Hardware:**

|  |  |
| --- | --- |
| Memory | 8GB |
| Hard disk | 50GB |
| Processor | 2.6 GHz I7 |

This code has been tested on the above software/hardware. IDE used to test is Eclipse and tool used is JMeter and Postman

## Building:

1. Download the code from github
2. Build the code using the following command “mvn install”

## Configuration:

Configuration is required for MongoDB and RabbitMQ. Default configuration is:

**RabbitMQ:**

Host: localhost

**MongoDB:**

mongoUrl: mongodb://localhost:27017/local

To Change the configuration, following files needs to be modified.

**RabbitMQ:**

Co2-sensor-worker: src/main/resources/workerapplication.properties

Co2-sensor-server: src/main/resources/application.properties

**MongoDB:**

Co2-sensor-domain: src/main/resources/domainapp.properties

Co2-sensor-server: src/main/resources/application.properties

## Output Artifacts:

On successfully building the source code, output shall be two jar files in their respective target folder. Both this jars needs to be deployed to test the service.



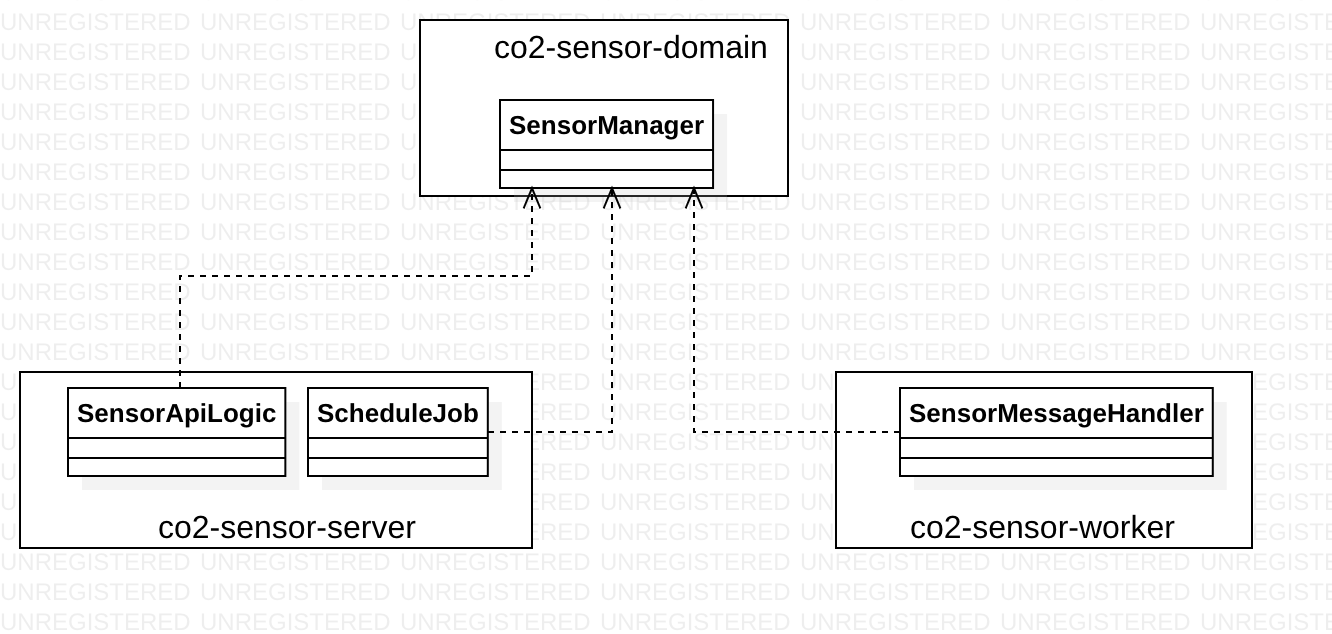
## Design Diagram

**Co2-sensor-domain:** It contains the core logic of the service. It interacts with DB and also contains the data model.

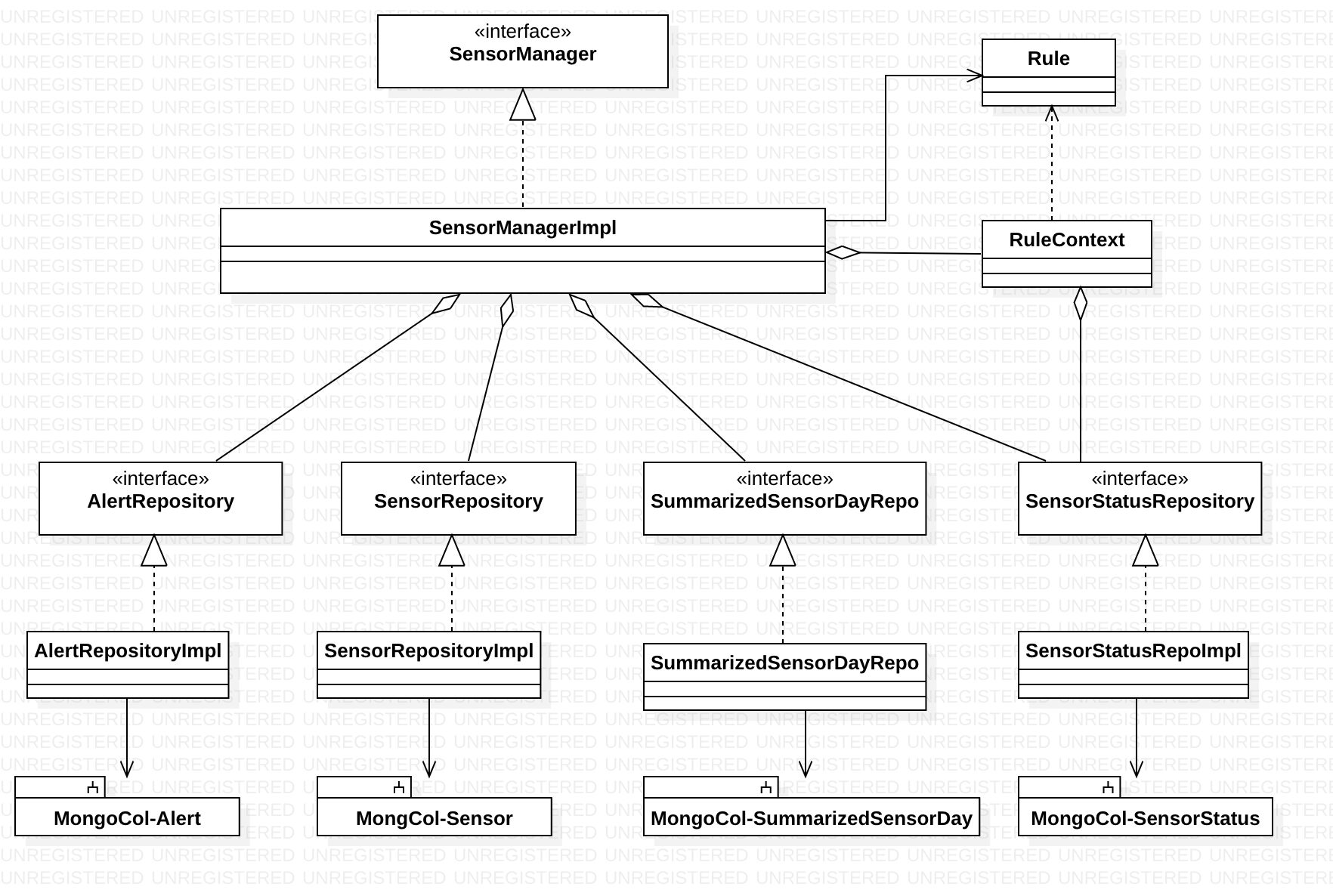
**Co2-sensor-server:** It contains the controller class for all the api’s

**Co2-sensor-worker:** It contains the handler class to listen to rabbit Queue.

## Module Dependency Diagram



## co2-sensor-domain Design Diagram



## Co2-sensor-worker Design Diagram



## Co2-sensor-server Design Diagram



## Api Design:

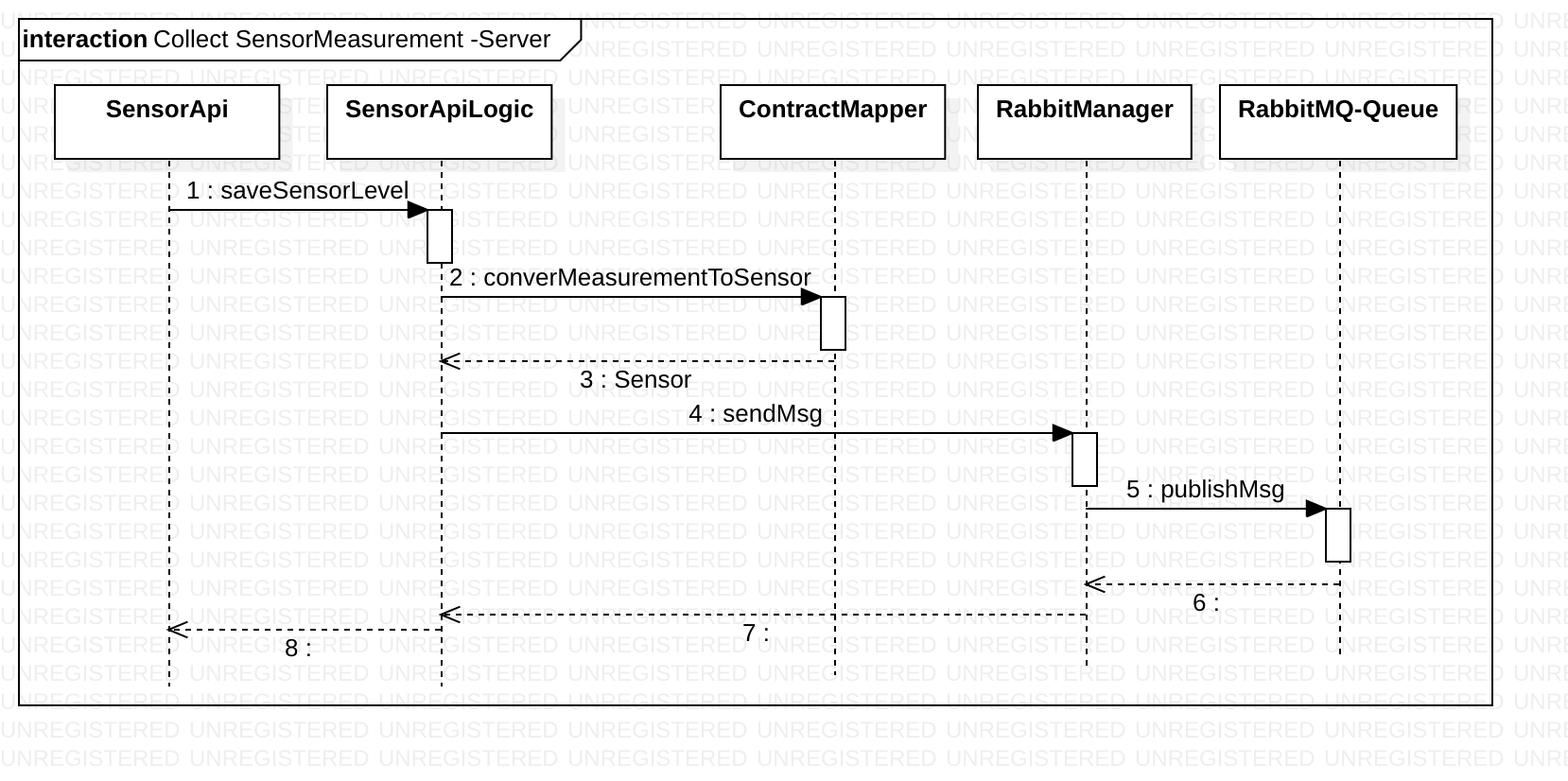
## Collect Sensor Measurement

This api has two parts. When the request hit the co2-sensor-server it passes the request to RabbitMQ. Co2-sensor-worker listens to the rabbit queue and when the message arrives it picks the message execute the rule to determine the sensor status and stores it internally memory. It keeps on storing the messages till it receives a msg from a sensor for which data is already been stored in internal memory, it then does a **bulk insert** to mongoDB.

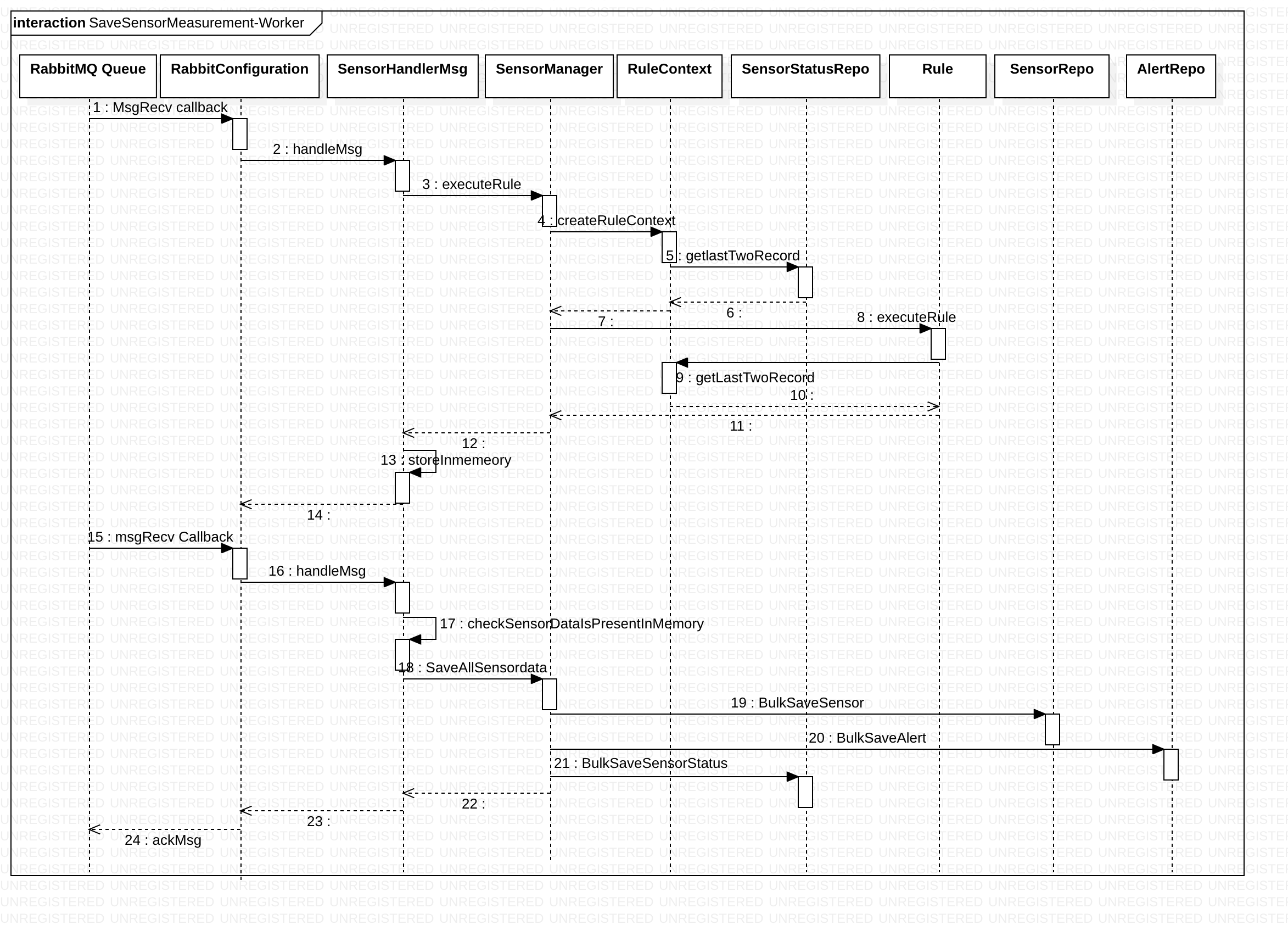
Following collections are updated due to the bulk operation.

1. Sensor: Bulk Insert operation
2. SensorStatus: Bulk Update operation
3. Alert: Bulk insert operation for all the sensors that have alert data

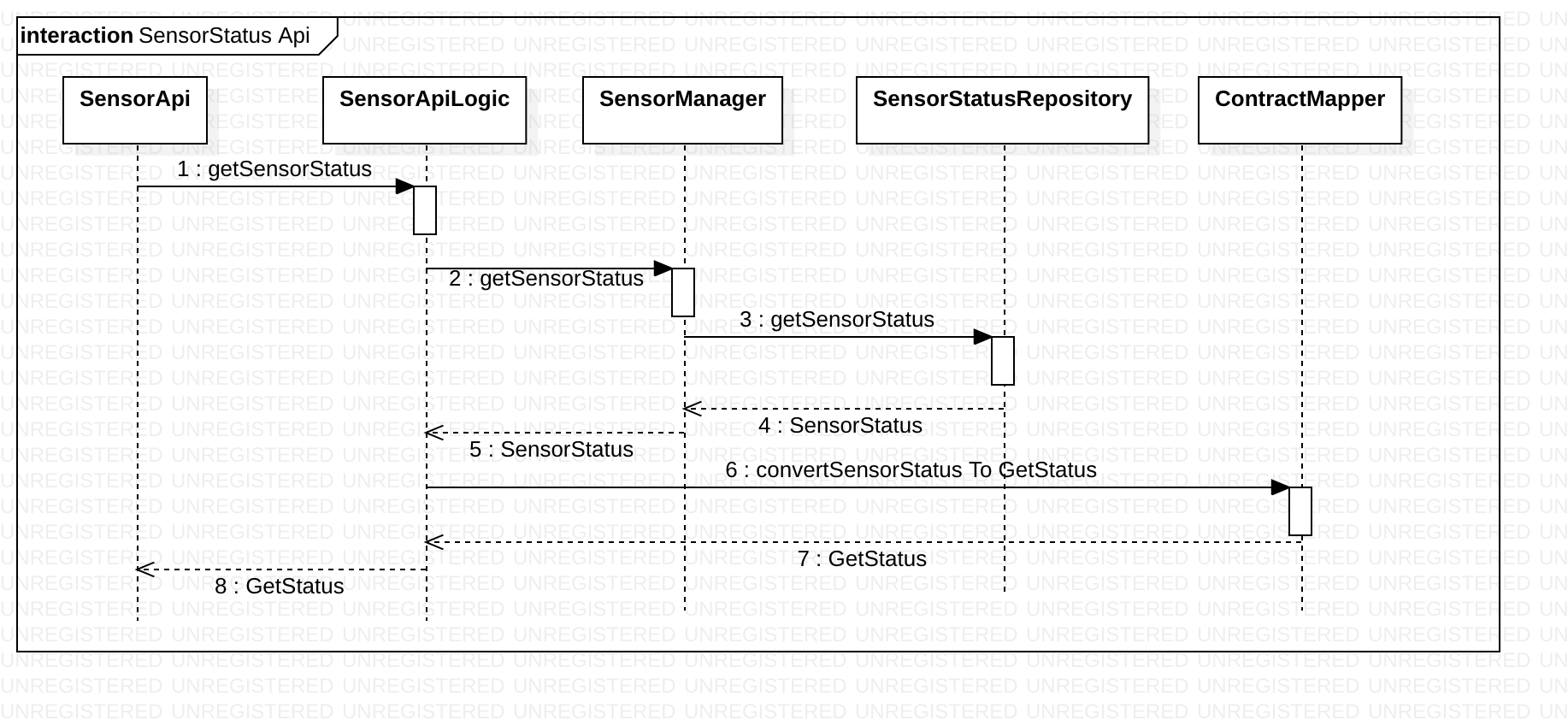
Api sequence flow for co2-sensor-server:



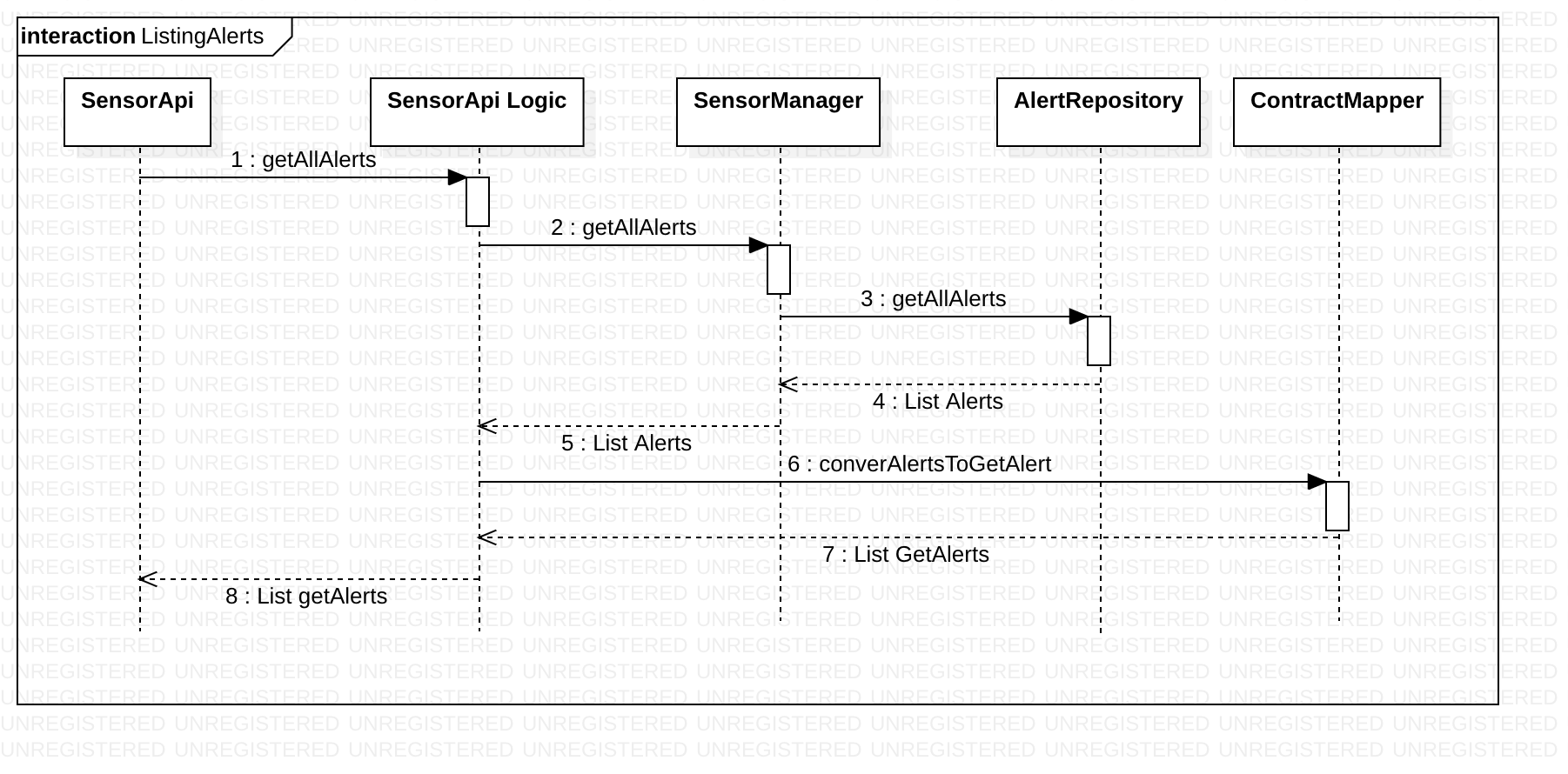
Sequence flow for co2-sensor-Worker:



## Sensor Status Api:



## Listing Alerts

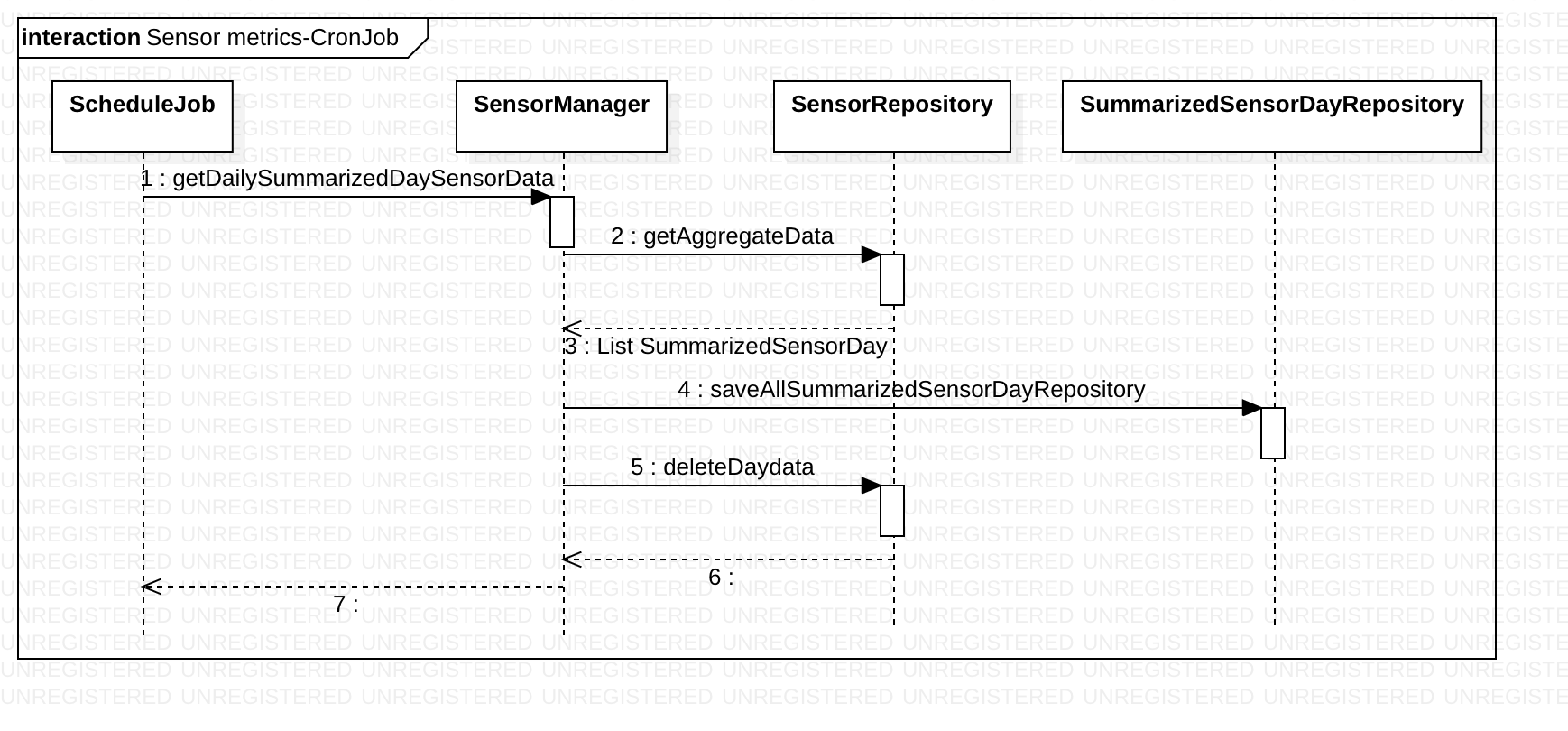


## Sensor metrics

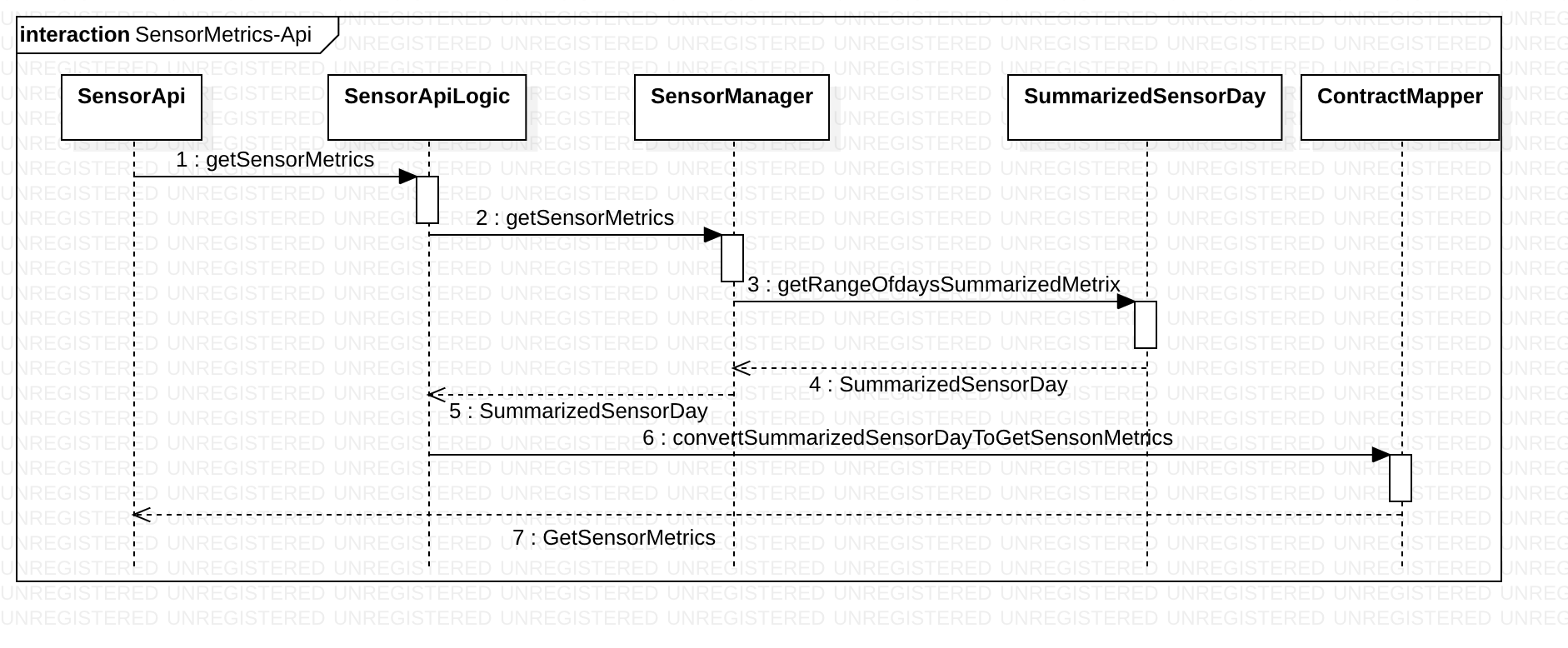
A Cron Job is configured to run every day at 00.00 am. This job aggregates previous day data from sensor collection and **bulk inserts** it in “SummarizedSensorDay “collection. It also a bulk deletes the collected data from “Sensor” collection.

Sensor metrics api aggregates one month data from “SummarizedSensorDay” collection return it back.

Cron Job Sequence:



Sensor metrics api flow:



## Database Design:

MongoDb is selected to store data. There has been no evaluation done to select the MongoDB. It is selected as it was readily available. To select an optimized database performance evaluation is required.

MongoDB collections are:

Sensor:

{

"\_id":"qUBGECPhcvUM+LSzVqQCgA==",

"\_class":"com.assessment.co2.sensor.domain.model.Sensor",

"sensorId":"I0yAyR43PpNjOEQkdJuFnA==",

"level":2500,

"recordingDateTime":"2019-08-02T19:11:47.000Z",

"status":"ALERT",

"isAlert":true

}

SensorStatus:

{

"\_id":"I0yAyR43PpNjOEQkdJuFnA==",

"\_class":"com.assessment.co2.sensor.domain.model.SensorStatus",

"status":"WARN",

"lastValue":2300,

"lastMinusOneValue":1800

}

Alert:

{

"\_id":"xEWFtD4iv8NGWDmmKHpRiQ==",

"\_class":"com.assessment.co2.sensor.domain.model.Alert",

"sensorId":"I0yAyR43PpNjOEQkdJuFnA==",

"status":"ALERT",

"level":2300,

"recoredDate":"2019-08-03T12:09:51.000Z"

}

SummarizedSensorDay

{

"\_id":"gk+IwKI7vy4aebmMB1xYhQ==",

"\_class":"com.assessment.co2.sensor.domain.model.SummarizedSensorDay",

"sensorId":"I0yAyR43PpNjOEQkdJuFnA==",

"date":"2019-08-01T18:30:00.000Z",

"max":2400,

"average":2160

}