Data Science Project: CaaS

0120000304
ROMEO
VELVI

Notes

The system proposed has been ideated according to these papers:

[1] Yin, C., Xiong, Z., Chen, H., Wang, J., Cooper, D., & David, B. (2015). A literature survey on smart cities. Science China. Information Sciences, 58(10), 1-18.

[2] Kaur, M. J., & Maheshwari, P. (2016, March). Building smart cities applications using IoT and cloud-based architectures. In 2016 international conference on industrial informatics and computer systems (CIICS) (pp. 1-5). IEEE.

As far as it concerns, the whole analysis divides the CaaS in three different topics:

1- Smart People
   a. For the use-case description click here
   b. For the analysis click here
2- Smart Logistic
   a. For the use-case description click here
   b. For the analysis click here
3- Smart Environment
   a. For the use-case description click here
   b. For the analysis click here

Contents

Smart City, Platform CaaS: City-as-a-Service

This project also aims to extend the Smart City definition and characteristic in order to build all the tools which are most likely able to represent and handle the idea of CaaS Platform: City-as-a-Service Platform.

The paper "*A literature survey on smart cities*" introduce a way to identify smart city utilities and application domain scenario which can be considered the ideal system plan:

Domain	Sub-Domain	Description
Government (more efficient)	E-government Transparent government Public service Public safety City monitoring Emergency response	Improving the internal and external efficiency of the government; enabling citizens and other relevant organizations to access official documents and policies; ensuring that public services work efficiently; monitoring and managing public safety; responding quickly and effectively in emergency situations.
Citizen (happier)	Public transport Smart traffic Tourism Entertainment Healthcare Education Consumption Social cohesion	Traveling and moving more efficiently; accessing contextualized, precise, real-time information in daily life; high-quality essential public services such as education, healthcare and sport; enriching spare time activities, communicating and sharing more with others.
Business (more prosperous)	Enterprise management Logistics Supply chain Transaction Advertisement Innovation Entrepreneurship Agriculture	Improving inter-management efficiency and quality; using more efficient logistics and supply chain platforms and methods; advertising more widely and accurately; expanding trade partners and customers; facilitating entrepreneurship and investment; upgrading the business activity in a city, such as production, commerce, agriculture and consulting; fostering innovation
Environment (more sustainable)	Smart grid Renewable energy Water management Waste management Pollution control Building Housing Community Public space	Delivering more sustainable, economic and secure energy and water supplies by taking into account citizens' behavior; using more green or renewable energy; recycling and treating waste efficiently and safely; reducing and preventing pollution in the city; offering mobility, telecommunication, information and all other facilities in different city spaces.

Meantime, the paper "*Building smart cities applications using IoT and cloud-based architectures*" introduce some classical usages of some system which ideas are currently being developed and applied in a practical scenario (Dubai Smart City). Here some examples:

Platform concept	Description
STS: Smart Transportation System	A proposal for the provision of Smart Transportation System through the formation of a common control to improve transportation and traffic systems.

My Window	My Window, which will allow residents and institutions, easy and shared access to data and information about schools, roads, hospitals, buildings, transport systems, energy, etc.
SED: Smart Electrical Grid	The design and development of Smart Electrical Grid to encourage homeowners and buildings in Dubai to use solar energy.
Smart-Meters	Smart meters would be introduced with the target of regulating the use of electricity and water in Dubai.
Smart Parks	Smart Parks and Smart Beaches projects, which would provide detailed information about weather conditions, temperatures and safety guidelines.
Municipality	Municipality of Dubai has also implemented mandatory green building regulations for the private sector to become a Green City.
5D Control Room	5D Control Room, world's largest of its kind will monitor government projects, service indicators and monitoring roads, weather conditions and emergency situations.

The ideological and practical point of views in smart city applications are strictly based on three/four essential classes:

- Smart People.
- Smart Logistics.
- Smart Environment.
- Eco Solution.

For the sake of this project, we are now introducing some practical use-case examples in order to design some solutions.
Note, we do not mention any kind of ecological or green choice, so the Eco Solution part is not analyzed.

Smart People

The "Smart People" initiative within the smart city framework is dedicated to enhancing the interaction between citizens and their urban day-life experience through a specialized smartphone application. This system envisions each individual as an integral component of the city, enabling proactive participation in the smart ecosystem.

Through this application, citizens have access to a wealth of resources and information. They can explore descriptions of key landmarks such as monuments, artworks, statues, and architectural sites, which enriches their experience, especially in tourism. Additionally, users can consult official government directives and documents, ensuring they are well-informed about public policies. The application also provides quick access to information on public services like buses and trains, making commuting more efficient. In case of emergencies, the app delivers real-time alerts and allows users to make rapid emergency calls. Moreover, users can access interactive maps and environmental control data, integrating the functionalities of Smart Logistics and Smart Environment.

The data that fuels this system comes from various sources. Government entities regularly publish documents and directives on the platform. Cultural and artistic entities contribute by providing descriptions of artworks and cultural sites every six months the data can includes street locations, coordinates, images, and descriptions, and is stored indefinitely.
Public transportation services, such as buses and trains, update their status daily, adding another layer of information.
Emergency entities play a critical role by using an ad-hoc system to intercept emergency calls and broadcast alerts in real-time. These alerts are small XML messages, stored indefinitely, and the volume of data is minimal, just a few megabytes per year per source.

The system's analytical component is minimal, primarily functioning as a classic Publish-Subscribe (PubSub) repository system. This means it efficiently disseminates information to users without extensive data analysis, focusing instead on real-time data distribution and interaction. This approach ensures that citizens are continuously engaged and informed, playing an active role in the smart city's dynamic environment.

Smart Logistic

The "Smart Logistic" initiative within the smart city framework focuses on enhancing transportation control, managing traffic flow, and alleviating congestion.
The system involves multiple entities working together to ensure efficient urban mobility.

Central to this subdomain are the smart citizens who interact with the system through a mobile application. They can query maps, tailored for either vehicle or pedestrian use, to navigate the city. By transmitting their location data every five minutes while the app is active, users contribute to a more accurate prediction of traffic flow. This data, structured with a timestamp, device identifier, and coordinates, is transmitted securely via HTTPS over TCP. Importantly, the data is not stored long-term; once processed, it is discarded for privacy law.

Traffic signals equipped with sensors, including cameras and radar guns, play a vital role in this system. These sensors regulate traffic intelligently and manage traffic flow by actuating the status of traffic lights (green, yellow or red). They also transmit data about street video captures and vehicle crossing speeds. The video captures are transmitted in real-time and stored using a weekly roll-in roll-out approach, with cold data archived for six months to serve as evidence in case of incident. The real-time videos are transmitted via RTP over UDP. The vehicle crossing speeds are recorded whenever there is a violation, with the data stored for a week. Additionally, this information is valuable to local security departments like the police.

The analytical component of this subsystem processes sensor data in real-time. For instance, it analyzes street video captures to determine the number of vehicles waiting at a traffic light, enabling dynamic adjustment of the light's status to improve traffic flow. All the sensing data are transmitted via MQTT over TCP/IP. In the case of speeding violations, the system automatically processes the data and, if a violation is detected, sends the timestamp, location, and vehicle plate image to the local security services for further actions. This analysis occurs weekly, with the violation data stored for six months and transmitted securely via HTTPS over TCP/IP.

Furthermore, the system processes users' location data to predict traffic conditions, indicating areas with different levels of trafficability (white, orange, red) on user maps. This prediction is updated every five minutes.
Daily, a cron-job executes the roll-in roll-out mechanism to archive video data, ensuring efficient storage management.

Through these coordinated efforts, the smart city subdomain effectively manages transportation, enhances traffic flow predictions, and improves urban mobility.

Smart Environment

The "Smart Environment" initiative within the smart city framework facilitates direct analysis and visualization of environmental situations and weather conditions. The key participants "Smart People", have access to real-time information about their local environment. They can view day-time predictions and receive alerts for poor air quality, prompting them to use face masks if necessary.

Smart Lamps in the city adjust their brightness based on the ambient light, thanks to data from light sensors. These sensors measure and transmit sunlight levels every 20 minutes, storing one sample per hour for long-term analysis. This data is structured with timestamps, locations, device information, and telemetry, transmitted via the MQTT protocol over TCP/IP.

Air quality sensors monitor pollutants such as CO2, NOx, and particulate matter, sending data every 10 minutes. Like the light sensors, they store one sample per hour. Temperature and humidity sensors operate on the same frequency and storage schedule. All this structured data, including timestamps, locations, device information, and telemetry, is transmitted using the MQTT protocol over TCP/IP.

The analytical subsystem processes data from these sensors to provide predictions and correlations, potentially utilizing advanced models like LSTM or ARIMA. Predictions for light sensors are updated every 10 minutes, while air quality, temperature, and humidity sensors are updated every 10 minutes. Special attention is given to temperature and air quality data for safety threshold analysis and environmental risk assessment. These critical updates are sent directly to the Smart People every 10 minutes, ensuring they remain informed and can take necessary actions.

Use-Case Analysis

In this section are listed and analyzed the various field mentioned before.

Smart People Analysis

It is a smart city branch related to the citizen application. Each person can be seen as a part of the city allowing them to interact with public and artistical topics. This makes each user a proactive part of the whole informational part of the smart city.

The entities in this part are:

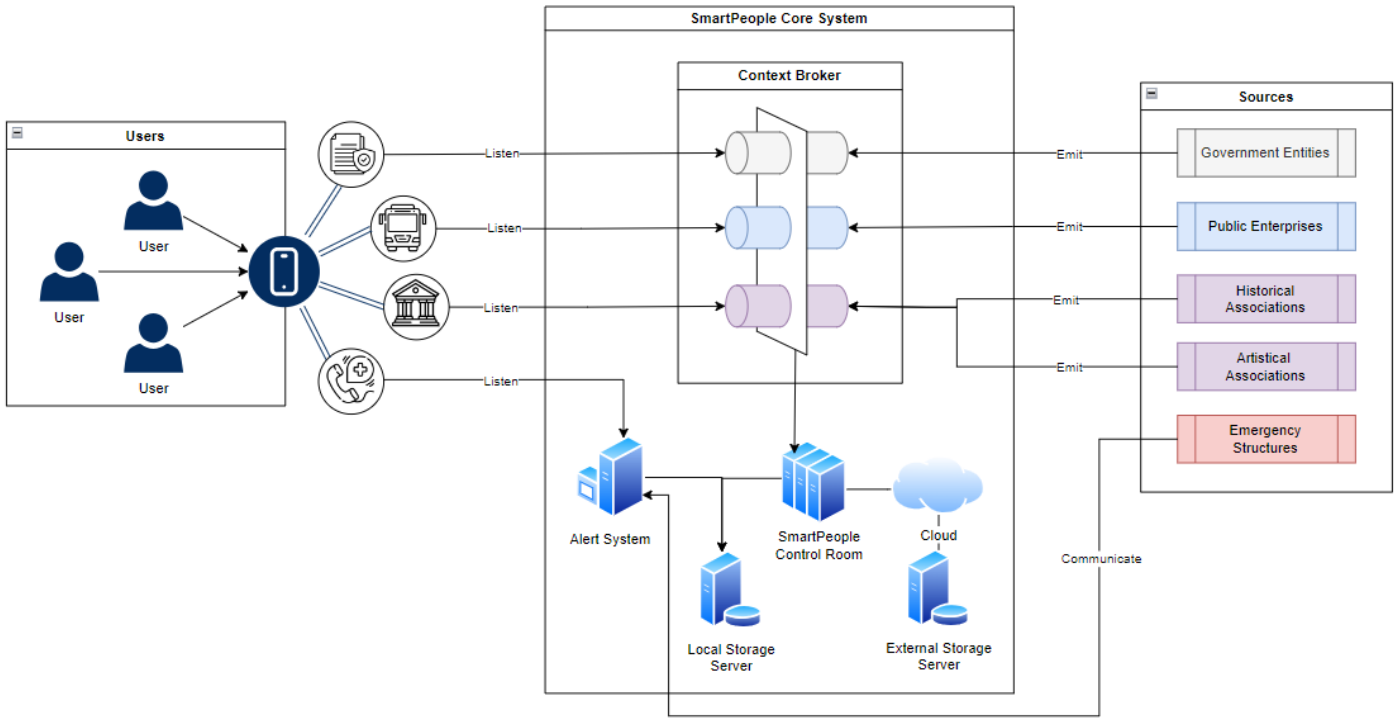
- People have access to the platform (application) which allows them to:
    - Hot points (monuments, arts, statues, architectures) descriptions essentially related with tourism activities.
    - Consult official government directives and papers
    - Fast access to public services information (bus, trains, etc.)
    - Fast access to alerts and emergency situations and rapid emergencies call

  Moreover, one can consult maps (see Smart Logistic) and environmental controls (see Smart Environment).

- Source of data:
    - Government Entities which publish the documents and directives
        - [Frequency] Once each month
        - [Storage] Indefinitely
        - [Volume] 100 MB/month ∀ Source
        - [Hypothetical structure] timestamp, source, document, OCR
        - [Protocol] HTTPS - TCP/IP
    - Cultural and Artistical Entities which publish the art's description
        - [Frequency] Once a month
        - [Storage] Indefinitely
        - [Volume] 500 MB/Semester ∀ Source
        - [Hypothetical structure] street, coordinates, images, description
        - [Protocol] HTTPS - TCP/IP
    - Other public information i.e. Bus, Trains which communicate their statuses.
        - [Frequency] Once a day
        - [Storage] Monthly
        - [Volume] 100 MB/day ∀ Source
        - [Protocol] HTTPS - TCP/IP
    - Emergencies Entities which have an ad-hoc system for intercept emergencies calls
        - [Frequency] Real-time
        - [Storage] When an alert has been (broadcasted) sent, the event is saved indefinitely
        - [Volume] few MB/year ∀ Source
        - [Hypothetical structure] timestamp, XML i.e. CLOB but note: the XML alert message are small (K,V)
        - [Protocol] CAP (reference: all6-it-alert-io-capit.pdf (protezionecivile.gov.it))

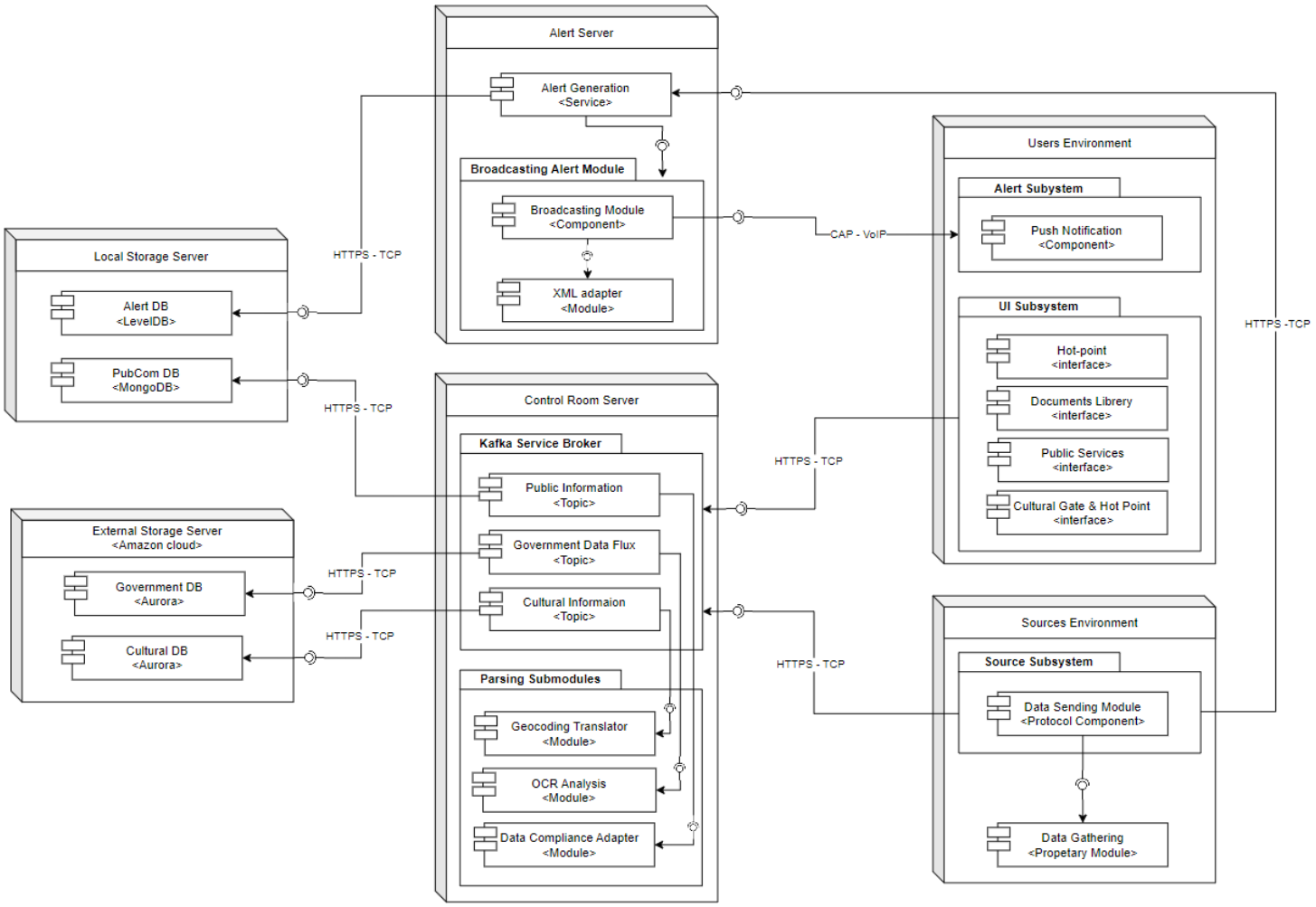
The Analytical part of this subsystem is practically absent (it's classical PubSub "repo" system env).

General Idea Smart People



The idea behind Smart People is that:
Users are able to query the application and navigate through various sources of information, which can be government public documents, artworks and even information about public entities.
Each topics mentioned before is generated and published by particular entities (which have the authority and knowledge to produce the information) on a certain channel controlled by a context broker which is handled by the control room system that analyzes and store the data for an effective and fast service.
The control room also stores the data in two different environments: one is local, and the other one is accessible via cloud. The emergency situation alert is handled ad-hoc with a system directly linked to the emergencies structures for a real-time data communication to the people.

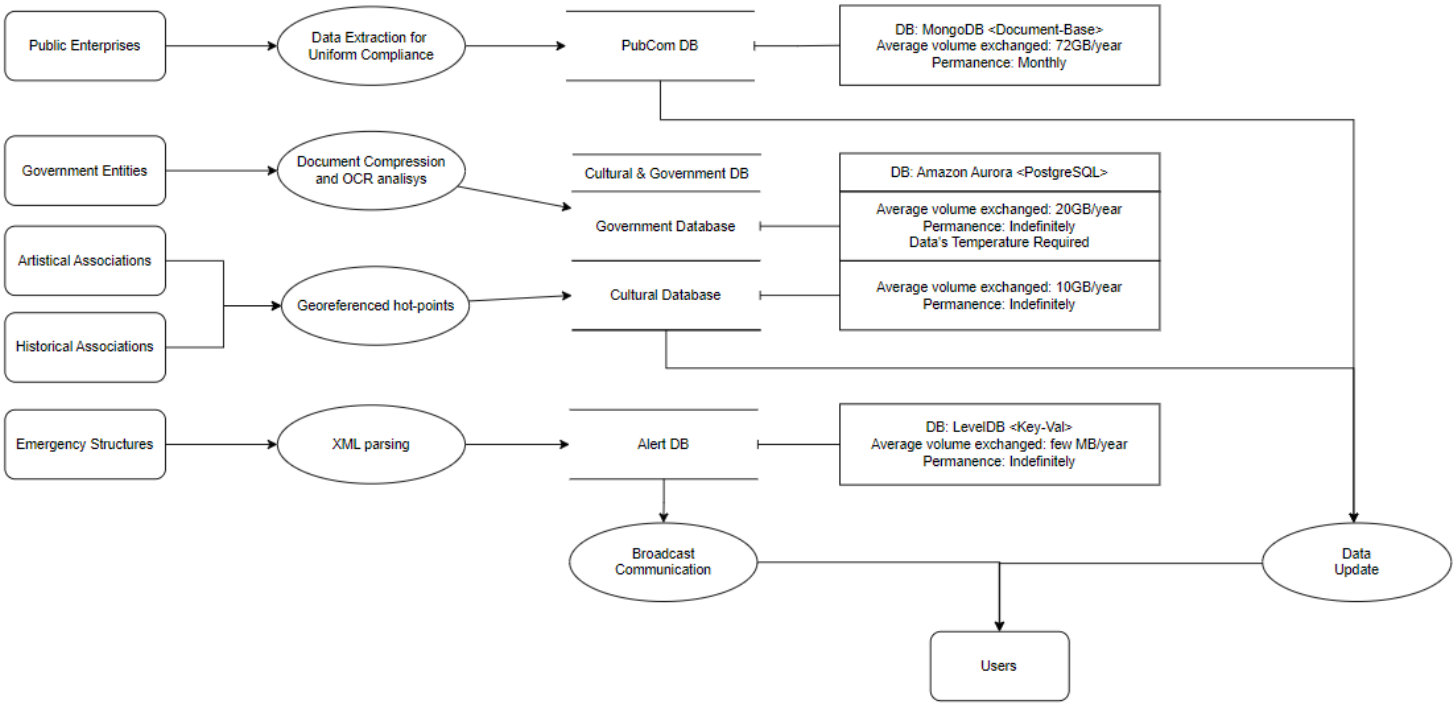
C.D.D. & D.F.D. Smart People



The following system reflects the general idea proposed:

- The **Users Environment** has two different subsystems: one is dedicated to the emergency (classical messaging system) already provided by the cellphone HW specs; the other part is the UI application which enable the user to navigate different topics in order to interact with different data.
- The **Sources Environment** can be different due to the fact that each public entity has its own proprietary system. Anyway, for representative purposes we divide the data sending module which should rely on the communication protocol and defined by apposite APIs exposed by the system.
- The **Alert Server** has two different modules: the Alert Generator which can be activate when the emergency source invokes it; and the second part is the Broadcast Alert Module which handle the broadcast communication part with all the users in order to send the emergency alert.
- The **Control Room** handles the various communications channels i.e. topic using a Kafka service broker module. This module relies on two parts: the first part is the parsing submodules that applies some data compliant and corrections; and the second part is the actual data storage (each topic saves data in different databases). It is worth to mention that the cultural and artistical data can be geocoded in order to make it available also with POI on the eventual map navigation system.
- The **Local Storage Server** handles the data storage for alert and other public service.
- The **External Storage Server** (Amazon) handles the storage for government and cultural data.

The D.F.D. is shown below



Analysis of costs

In order to build this kind of system the following choices are made for a polyglot DB environment:

1- For the public entities, due to the variety of information and type of data (which can be defined) we made these choices:
   a. Store all sources in a MongoDB situated in a local subsystem.
      i. MongoDB provides a Community Edition in our local infrastructure, which offers:
      ii. Powerful distributed capabilities with self-managing operations.
      iii. Allows to secure and encrypt data and gives access to an advanced in-memory storage engine.
      iv. Referred link: MongoDB Community Edition - MongoDB Manual v7.0
   b. The volume of the data is supposed to be handle a total of 100GB at year, but in overall, the expected monthly permanence will be more or less 10 GB at moth. Eventually we can also apply some subsystem which handles data's temperature (maybe applying in-memory solution w/t roll-in roll-out weekly strategies).

2- For government data and cultural associations, we must rely on system which grant safe and reliability solution. A global estimation is made Amazon Aurora PostgreSQL-Compatible DB (calculator.aws):

   a. Unit conversions.
      i. Storage amount: 10 TB x 1024 GB in a TB = 10240 GB
      ii. Baseline IO rate: 22 per day / (24 hours in a day x 7 days in a week) = 0.92 per hour
      iii. Peak IO rate: 22 per day / (24 hours in a day x 7 days in a week) = 0.92 per hour
   b. Pricing calculations
      i. 10,240 GB x 0.10 USD = 1,024.00 USD (Database Storage Cost)
      ii. 730 hours in a month - 10 peak hours per month = 720.00 baseline hours per month
      iii. 0.92 baseline IOs x 720.00 baseline hours per month = 662.40 (Monthly Baseline IOs)
      iv. 662.40 (Monthly Baseline IOs) x 0.0000002 USD = 0.00013248 USD (Baseline IOs Cost)
      v. 0.92 peak IOs x 10 peak hours per month = 9.20 (Monthly Peak IOs)
      vi. 9.20 (Monthly Peak IOs) x 0.0000002 USD = 0.00000184 USD (Peak IOs Cost)
      vii. 0.00013248 USD (Baseline IO Cost) + 0.00000184 USD (Peak IO Cost) = 0.00 USD (IOs Rate Cost)

Total Storage Cost (monthly) on average: 1,024.00 USD

3- For the emergencies storage we prefer the usage of local LevelDB which is an efficient open-source on-disk key-value store written by Google in order to handle few MB each month.
    a. Check information about on this link: LevelDB
    b. Also check the public repo: google/leveldb.

Smart Logistic Analysis

It is a smart city subdomain which helps the transportation control, traffic flux and manage congestion.

The entities in this scenario are :

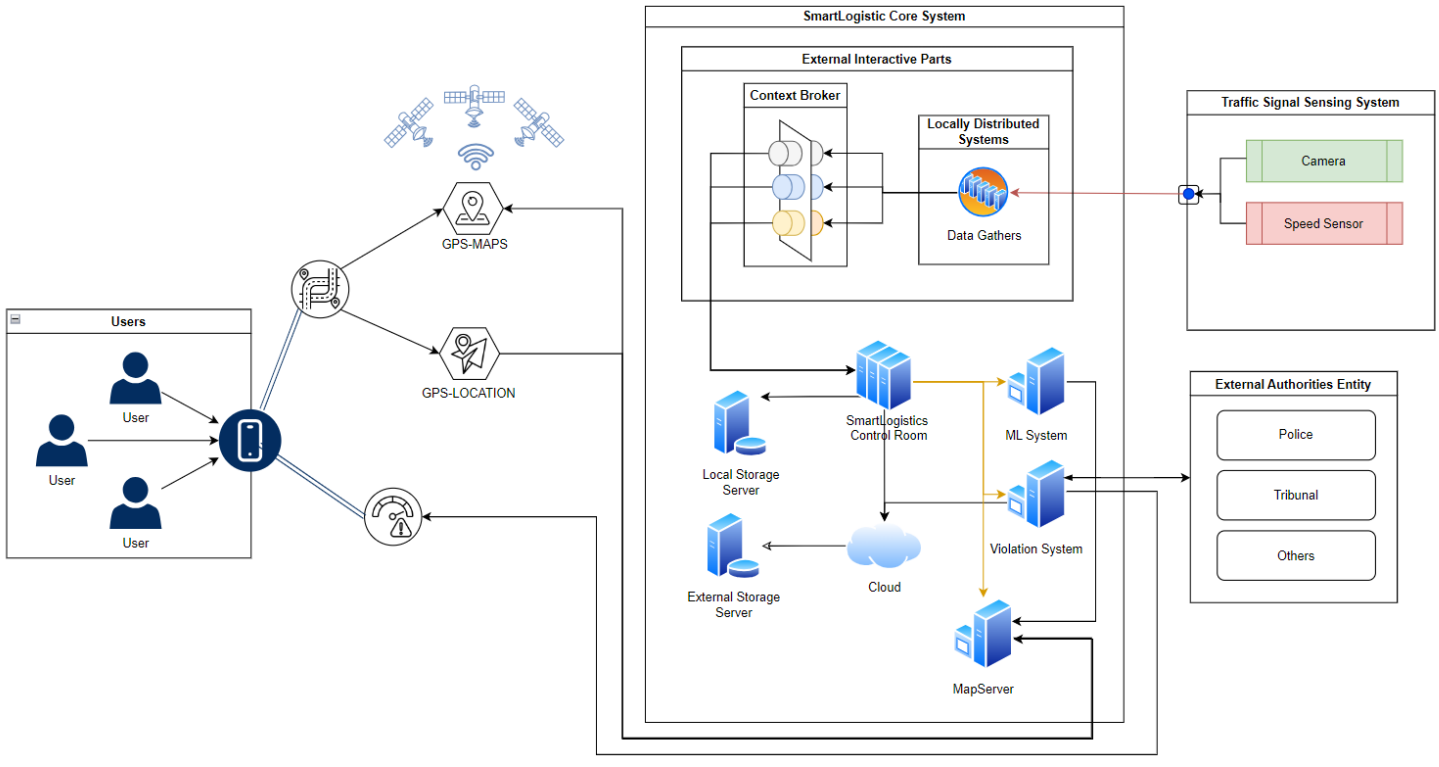
- The Smart People that:
    - Interrogate the maps (for vehicle, bicycle or pedestrian). The maps status are updated via Map-Server that computes the shortest path in order to reduces congestion and regulate flux/affluence.
    - The user position is sent (via Map-Server) in order to process the data and eventually make a more accurate traffic prediction. Moreover, the users have to choose which transport method they have to use (car, bicycle, or just walk) in order to get and transmit more accurate information.
        - [Frequency] the user location is updated once each 5 minutes
        - [Storage] None i.e. once the data has been processed, they are discarded
        - [Volume] 5 MB/day ∀ user
        - [Hypothetical structure] timestamp, session key (privacy policy), means of transport, coordinates
        - [Protocol] HTTPS - TCP
- Each traffic signals have a block sensing system (composed by cameras and radar guns) which handle an intelligent flux via traffic-regulation. The data sent from the sensing system can be of two possible kinds:
    - *Street video capture*
        - [Frequency] Real-time
        - [Storage] Hot data are weekly settled using a *weekly* roll-in roll-out approach and the cold data are kept for 1 year (for incidents/accident proof or similar).
            - Note: in this week, the video can be fast-consulted at any time for any reason.
        - [Volume] 10 GB/week ∀ Source (compressed data)
        - [Protocol] SRTP - TCP
    - *Road speed violation*
        - [Frequency] Whenever there is a violation
        - [Storage] 1 year
        - [Volume] 1 MB/week ∀ Violation
        - [Hypothetical structure] timestamp, coordinates, speed reached, frame
        - [Protocol] MQTT – TCP/IP

The Analytical part of this subsystem is related to:

1) take information from the *street video capture* and:
    - [***car-count submodule] the map status is updated considering the street-intersection flux in order to gives a hint/alternative of what path a user should take for reaching its destination avoiding too much congestion on the same road for each user.
        - [System Update Frequency] Real-time analysis
        - [User Update Frequency] In order to avoid excessive users-server communication, once each 5 minutes is updated.
    - [road accident submodule] the data are weekly analyzed in order to find if some accident happened, and, in the case, a short-video is cut/taken and kept in case of request from the local authorities.
        - [Frequency] The analysis is weekly computed
        - [Information Storage] 1 year
        - [Volume generated] 100 MB/day ∀ Violation
        - [Hypothetical structure] timestamp, camera who record the accident, coordinates, short-video
- [violation module] Take information from the and process, from *road inflation speed* and:
    - [plate recognition submodule] process the frame sent by the sensor and retrieve the plate and store the event into a database.

        - [Frequency] Real-Time
        - [Storage] 1 year (starting from signal timestamp)
        - [Volume] 1 MB/day ∀ Violation
        - [Hypothetical enriched structure] timestamp, location, frame, plate string
        - [Protocol] HTTPS – TCP/IP
    - [violation message submodule] when a violation has been committed and stored, the next step is using some authority channel to retrieve the owner of the plate and send a message directly to owner of the vehicle that committed a violation.
        - [Frequency] Real-Time
        - [Protocol] HTTPS – TCP/IP
- [path hint module] Takes information from the users' location (and *street video capture* ***), for:
    - [Channel recommendation] this makes prediction for the white, orange and red trafficability area and intersection such that it sends alternative/different path in order to avoid congestions . This information can be seen via user application when consulting maps.
        - [System Update Frequency] Real-time analysis
        - [User Update Frequency] In order to avoid excessive users-server communication, once each 5 minutes is updated.
    - [Blocking Access system] this is purely defined on that areas that are known for supporting a limit amount of people. When a certain number of peoples are located on a given area, this module can lock/advice against that particular section.
        - [System Update Frequency] Real-time analysis
        - [User Update Frequency] In order to avoid excessive users-server communication, once each 5 minutes is updated.
- [roll-in-roll-out module] Once a day, a service (via cron-job) executes the roll-in roll-out of the video in order to archive them.

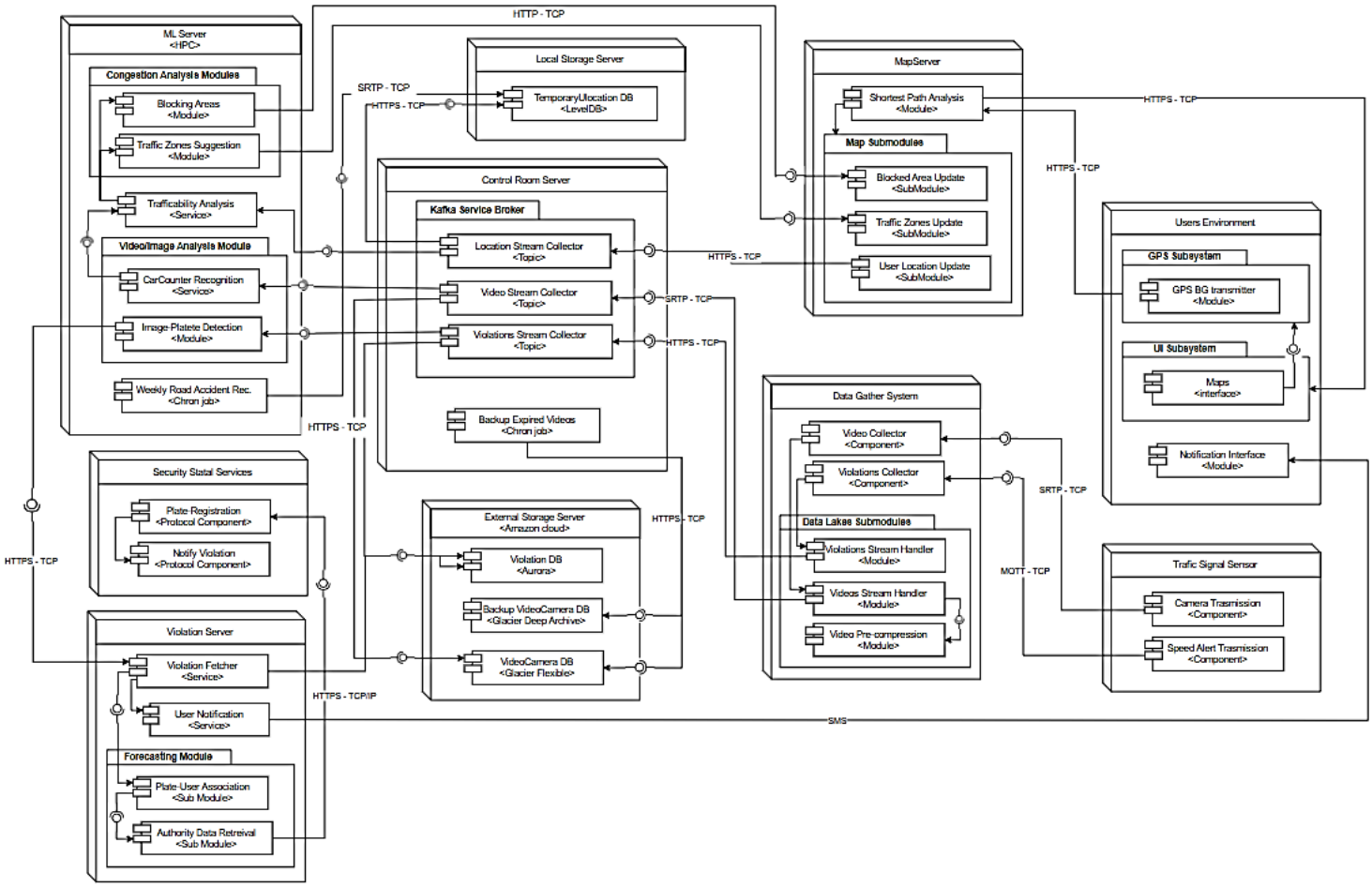
General Idea Smart Logistic



The idea behind Smart Logistic is that:

- The people, access to the application (interface via map-server) and can open the maps, which is updated with hot-point and congestion analyzed considering the means of transport chosen.
- Meantime, the map server, sends the users positions to the control room which is analyzed and processed in order to update the trafficability status of the city in combination to the camera sensor.
- The traffic flux is regulated with the smart signal system which:
    - Sends real-time video of the interception in order to be:
        - Real-time analyzed by the control room which updates the congestion status over the interception channels i.e. update the trafficability status.
        - Weekly block-analyzed such that identify road accident in order to store the video and make it available for further investigations.
    - Moreover, the traffic signals are equipped with speed sensor that sends data about the cars speed violations. When the control room got this kind of messages, it analyzes the snapshot taken for retrieving via plate-recognition the owner in order to send a violation message directly to him/her. This data are also shared to the local authorities (e.g. police department) in order to be proceed with law processes.
- Also, in this case the control room works with two different storage environments: one local, one on cloud.
- All the analytical and recognition part are handled with an apposite machine learning methods which are preformed using a HPC server.
- Since the large number of sensors available on the city, all the communication are done with various data gather system and actuator spread throughout the city that collect, preprocess and send the data in order to speed up the elaboration.

C.D.D. & D.F.D. Smart Logistic



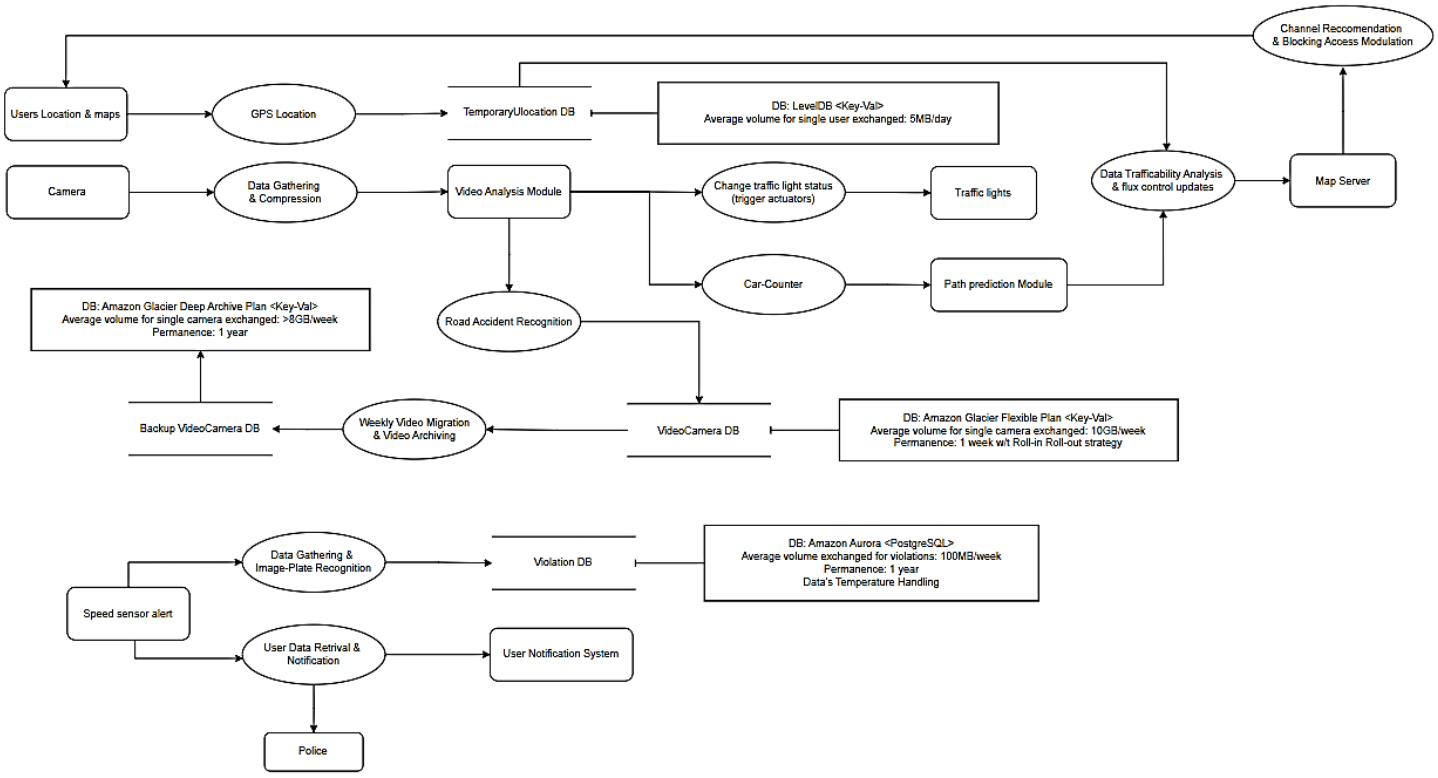
The following system reflects the general idea proposed:

- The **Users Environment** has three different subsystems:
    - the first part sends the user position (the GPS HW is supposed to be preinstalled into the smartphone)
    - the second part is the UI application which enable the user to consult the maps after choosing the means of transport (like Google Maps).
    - the third part is the message subsystem that is just responsible to show the (violation) messages.
- The **Traffic Signal Sensor** which can be seen as a unique block sensor composed by various components:
    - a camera that sends the real-time video.
    - a speed radar which sends the information about speed violations.
- The **Data Gather System** is responsible for getting the vast amount of data coming from the cameras and the speed sensor violations and applies a first layer of data preprocessing (compression) before sending it to the control room.
- The **Control Room** handles the various communications channels i.e. topic using a Kafka service broker module. This module relies on three parts:
    - the first part is the user location that gathers the users' data location.
    - the second and third part is dedicated to the video collection.
    - the third part is dedicated for the violations.

  Moreover, the control room communicates with a ML server that:
    - sends the user position in order to predict the people affluence.
    - sends the video-capture in order to process the channel's interceptions congestion.
    - sends the violation such that via image recognition can automatically discern the vehicle plate.
- The **ML Server** is designed to be an HPC which purpose is the:
    - video and image recognition and detection.
    - traffic analysis.
    - weekly road accidents detection.

  Moreover, it directly communicates to the Map Server in order to updates the city's street information flus, saturation and congestion.
- The **Map Server** is designed to:
    - Handles user via shortest weighted path. The "weighted" is due to the fact that each intersection and street is updated with the flux conditions directly discerned from the ML Server.
    - For each user connected to it, the mapserver communicate each 5 minutes the user position to the control room, that updates the real-time ML prediction system and re-updates the affluence and congestion.
        - Auto-regulation strategy.
- The **Local Storage Server** handles the data coming from the user location (which has to be fast processed and after eliminated).
- The **External Storage Server** (Amazon) handles the storage of violations, cameras (and their backups) in different solution plan.

The D.F.D. is shown below



Analysis of costs

In order to build this kind of system the following choices are made for a polyglot DB environment:

1- For the temporary storage we prefer the usage of local LevelDB which is an efficient open-source on-disk key-value store written by Google in order to handle an amount of data of 5 MB each user. If we think that 100'000 users are contemporary connected, the system must have the capacity of maximum 500GB but since once each 5 minutes the data from users is elaborated and discarded (for privacy reason) using a sort of roll-in roll-out strategy on 5 minutes window. Long story short: we expect the volume never exceed that 500GB. Moreover, the system will be able to discard expiry location position.

   a. Check information about on this link: LevelDB
   b. Also check the public repo: google/leveldb.

2- For the video system, we have two kinds of different supports:

   a. Amazon Glacier Flexible Plan (hot/warm video-data) for "weekly video" eventually requested by institutional entities and so we need a storage system which handles a large amount of data with the possibilities to retrieve data in minutes or hour if the requested data is in week-lifespan (a use-case scenario are the accident video proof). The amount of data is supposed to be 10 GB at week for single camera. If we expect a number of sensors equals to 2000 the volume of the data is 20TB each week. Applying some data preprocessing we can save up to 2/3 of the space, so the total volume will be approximately 15TB.
   A global estimation is made Amazon Simple Storage Service (S3) (calculator.aws):

      i. Unit conversions
         1. S3 Glacier Flexible Retrieval storage: 15 TB per month x 1024 GB in a TB = 15360 GB per month
         2. S3 Glacier Flexible Retrieval Average Object Size: 15 MB x 0.0009765625 GB in a MB = 0.0146484375 GB
         3. Data retrievals (Standard): 15 TB per month x 1024 GB in a TB = 15360 GB per month
      ii. Pricing calculations
         1. 15,360 GB per month / 0.0146484375 GB average item size = 1,048,576.00

unrounded number of objects
2. Round up by 1 (1048576.0000) = 1048576 number of objects
3. 1,048,576 number of objects x 32 KB = 33,554,432.00 KB overhead
4. 33,554,432.00 KB overhead / 1048576 KB in a GB = 32.00 GB overhead
5. 32.00 GB overhead x 0.0036 USD = 0.1152 USD (Glacier Flexible Retrieval storage overhead cost for metadata)
6. Glacier Flexible Retrieval storage overhead cost for metadata: 0.1152 USD
7. 1,048,576 number of objects x 8 KB = 8,388,608.00 KB overhead
8. 8,388,608.00 KB overhead / 1048576 KB in a GB = 8.00 GB overhead
9. Tiered price for: 8.00 GB
10. 8 GB x 0.023 USD = 0.18 USD
11. Total tier cost = 0.184 USD (S3 Standard storage overhead cost for metadata)
12. S3 Standard storage overhead cost for metadata: 0.184 USD
13. 15,360 GB per month x 0.0036 USD = 55.296 USD (Glacier Flexible Retrieval storage cost)
14. Glacier Flexible Retrieval storage cost: 55.296 USD
15. 0.1152 USD + 0.184 USD + 55.296 USD = 55.5952 USD (Total Glacier Flexible Retrieval storage cost)
16. 4 requests x 0.00003 USD = 0.0001 USD (PUT, COPY, POST, LIST requests to S3 Glacier Flexible Retrieval cost)
17. 4 requests x 0.00003 USD = 0.0001 USD (Lifecycle transitions cost)
18. 4 requests x 0.00005 USD = 0.0002 USD (Cost for Restore requests (Standard))
19. 4 requests x 0.01 USD = 0.04 USD (Cost for Restore requests (Expedited))
20. 4 requests x 0.00 USD = 0.00 USD (Cost for Restore requests (Bulk))
21. 15,360 GB per month x 0.01 USD = 153.60 USD (Cost for Data retrievals (Standard))
22. 1 Provisioned Capacity Units x 100.00 USD = 100.00 USD (Cost for provisioned capacity)
23. 55.5952 USD + 0.0001 USD + 0.0001 USD + 0.0002 USD + 0.04 USD + 0.00 USD + 153.60 USD + 0 USD + 0 USD + 100.00 USD = 309.24 USD (Total S3 Glacier Flexible Retrieval cost)

Total Storage Cost (monthly) on average: 309.24 USD

b. Amazon Glacier Deep Archive (cold video-data) for “oldest videos” essentially used as semestral backups.
If we think to have more or less 15TB each week, it brings us the total semestral capability of 360TB.
A global estimation is made Amazon Simple Storage Service (S3) (calculator.aws):

i. Unit conversions
   1. S3 Glacier Deep Archive storage: 360 TB per month x 1024 GB in a TB = 368640 GB per month
   2. S3 Glacier Deep Archive Average Object Size: 16 MB x 0.0009765625 GB in a MB = 0.015625 GB

ii. Pricing calculations
   1. 368,640 GB per month / 0.015625 GB average item size = 23,592,960.00 unrounded number of objects
   2. Round up by 1 (23592960.0000) = 23592960 number of objects
   3. 23,592,960 number of objects x 32 KB = 754,974,720.00 KB overhead
   4. 754,974,720.00 KB overhead / 1048576 KB in a GB = 720.00 GB overhead
   5. 720.00 GB overhead x 0.00099 USD = 0.7128 USD (Glacier Deep Archive storage overhead cost for metadata)
   6. Glacier Deep Archive storage overhead cost: 0.7128 USD
   7. 23,592,960 number of objects x 8 KB = 188,743,680.00 KB overhead
   8. 188,743,680.00 KB overhead / 1048576 KB in a GB = 180.00 GB overhead
   9. Tiered price for: 180.00 GB
   10. 180 GB x 0.023 USD = 4.14 USD
   11. Total tier cost = 4.14 USD (S3 Standard storage overhead cost)

12. S3 Standard storage overhead cost: 4.14 USD
13. 368,640 GB per month x 0.00099 USD = 364.9536 USD (Glacier Deep Archive storage cost)
14. Glacier Deep Archive storage cost: 364.9536 USD
15. 0.7128 USD + 4.14 USD + 364.9536 USD = 369.8064 USD (Total Glacier Deep Archive storage cost)
16. 4 requests x 0.00005 USD = 0.0002 USD (Cost for PUT, COPY, POST, LIST requests)
17. 4 requests x 0.00005 USD = 0.0002 USD (Cost for Lifecycle transitions)
18. 4 requests x 0.0001 USD = 0.0004 USD (Cost for Restore requests (Standard))
19. 5 GB per month x 0.02 USD = 0.10 USD (Cost for Glacier Deep Archive Data Retrieval (Standard))
20. 5 GB per month x 0.0025 USD = 0.0125 USD (Cost for Glacier Deep Archive Data Retrieval (Bulk))
21. 369.8064 USD + 0.0002 USD + 0.0002 USD + 0.0004 USD + 0.10 USD + 0.0125 USD = 369.92 USD (S3 Glacier Deep Archive cost)

Total Storage Cost (monthly) on average: 369.92 USD

3- For the Violation data, since we do not expect to have much data, we prefer an on-cloud solution based on Amazon Aurora via PostgreSQL. The amount of data is supposed to be 100MB/week which is 400MB/month approximately since we must store it for one year.
Considering the smallest amount of data generated, we could also rely on internal storage form which has zero-costs. But, since it is an average estimation, the data could grows, and, moreover, it useful to have all the cloud storage advantages.
The total semestral value will be Amazon Aurora PostgreSQL-Compatible DB (calculator.aws):

a. Unit conversions
   i. Storage amount: 500 MB x 0.0009765625 GB in a MB = 0.48828125 GB
   ii. Baseline IO rate: 22 per day / (24 hours in a day x 7 days in a week) = 0.92 per hour
   iii. Peak IO rate: 22 per day / (24 hours in a day x 7 days in a week) = 0.92 per hour
b. Pricing calculations
   i. 0.48828125 GB x 0.10 USD = 0.05 USD (Database Storage Cost)
   ii. 730 hours in a month - 10 peak hours per month = 720.00 baseline hours per month
   iii. 0.92 baseline IOs x 720.00 baseline hours per month = 662.40 (Monthly Baseline IOs)
   iv. 662.40 (Monthly Baseline IOs) x 0.0000002 USD = 0.00013248 USD (Baseline IOs Cost)
   v. 0.92 peak IOs x 10 peak hours per month = 9.20 (Monthly Peak IOs)
   vi. 9.20 (Monthly Peak IOs) x 0.0000002 USD = 0.00000184 USD (Peak IOs Cost)
   vii. 0.00013248 USD (Baseline IO Cost) + 0.00000184 USD (Peak IO Cost) = 0.00 USD (IOs Rate Cost)

Total Storage Cost (monthly): 0.05 USD

Smart Environment Analysis

It is a smart city subdomain which allows a direct analysis and visualization of environmental situation and weather conditions.

The entities in this scenario are :

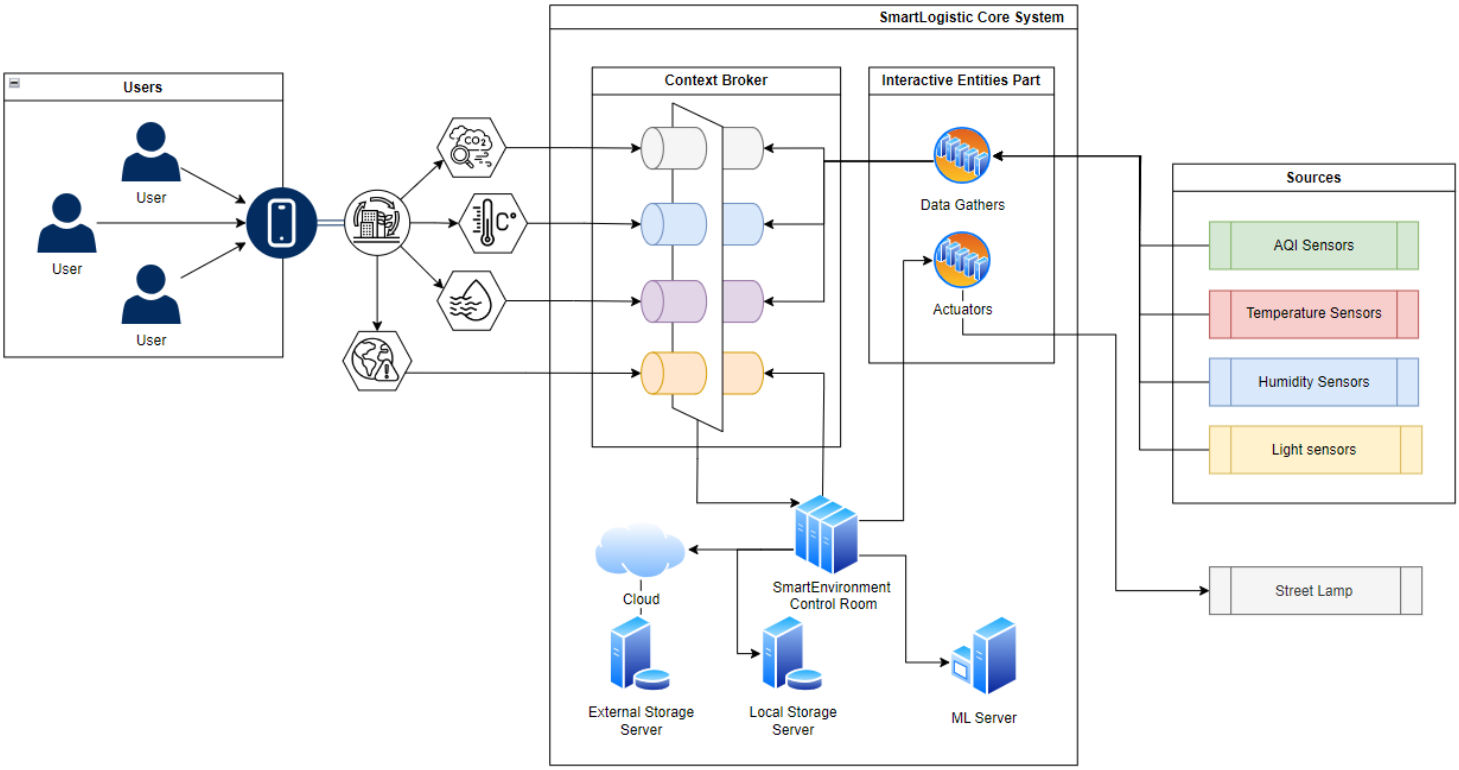
- Smart peoples:
  - which can watch the local environment situation with some day-time predictions.
  - watch alert in case of bad air quality (eventual uses of face masks).
- Smart Lamps that can:
  - adapt the luminosity based on how much light there is during the day.
  - Listen to the actuator directives
- Light sensors: sensor that measures and send the amount of light coming from the sun.
    - [Frequency] Once each 10 minutes
    - [Storage] For analytical purposes, just 24 sample ∀ Source throughout the day are stored (1/h) for 5 years
    - [Volume] 1 MB/week ∀ Source
    - [Hypothetical structure] timestamp, location, device, telemetry (K-V)
    - [Protocol] MQTT – TCP/IP
- Air Quality Sensors: sensors which monitor and send pollutants such as CO2, NOx, and particulate matter.
    - [Frequency] Once each 10 minutes
    - [Storage] For analytical purposes, just 24 sample ∀ Source throughout the day are stored (1/h) for 5 years
    - [Volume] 3 MB/week ∀ Source
    - [Hypothetical structure] timestamp, location, device, telemetry (K-V)
    - [Protocol] MQTT – TCP/IP
- Temperature sensors: sensors which monitor and send the temperature of a given area.
    - [Frequency] Once each 10 minutes
    - [Storage] For analytical purposes, just 24 sample ∀ Source throughout the day are stored (1/h) for 5 years
    - [Volume] 1 MB/week ∀ Source
    - [Hypothetical structure] timestamp, location, device, telemetry (K,V)
    - [Protocol] MQTT – TCP/IP
- Humidity sensors: sensors throughout the city which monitor and send the humidity percentage of a given area.
    - [Frequency] Once each 10 minutes
    - [Storage] For analytical purposes, just 24 sample ∀ Source throughout the day are stored (1/h) for 5 years
    - [Volume] 1 MB/week ∀ Source
    - [Hypothetical structure] timestamp, location, device, telemetry (K-V)
    - [Protocol] MQTT – TCP/IP

The Analytical part of this subsystem produce these effects:

- For the data that comes out from sensor we can apply some prediction and correlation (maybe using LSTM or ARIMA):
  - Light sensors
    - [Frequency] Forecasted and updated (current vs predicted) each 10 minutes
  - Air Quality Sensors
    - [Frequency] Forecasted and updated (current vs predicted) each 10 minutes
  - Temperature sensors
    - [Frequency] Forecasted and updated (current vs predicted) each 10 minutes
  - Humidity sensors:

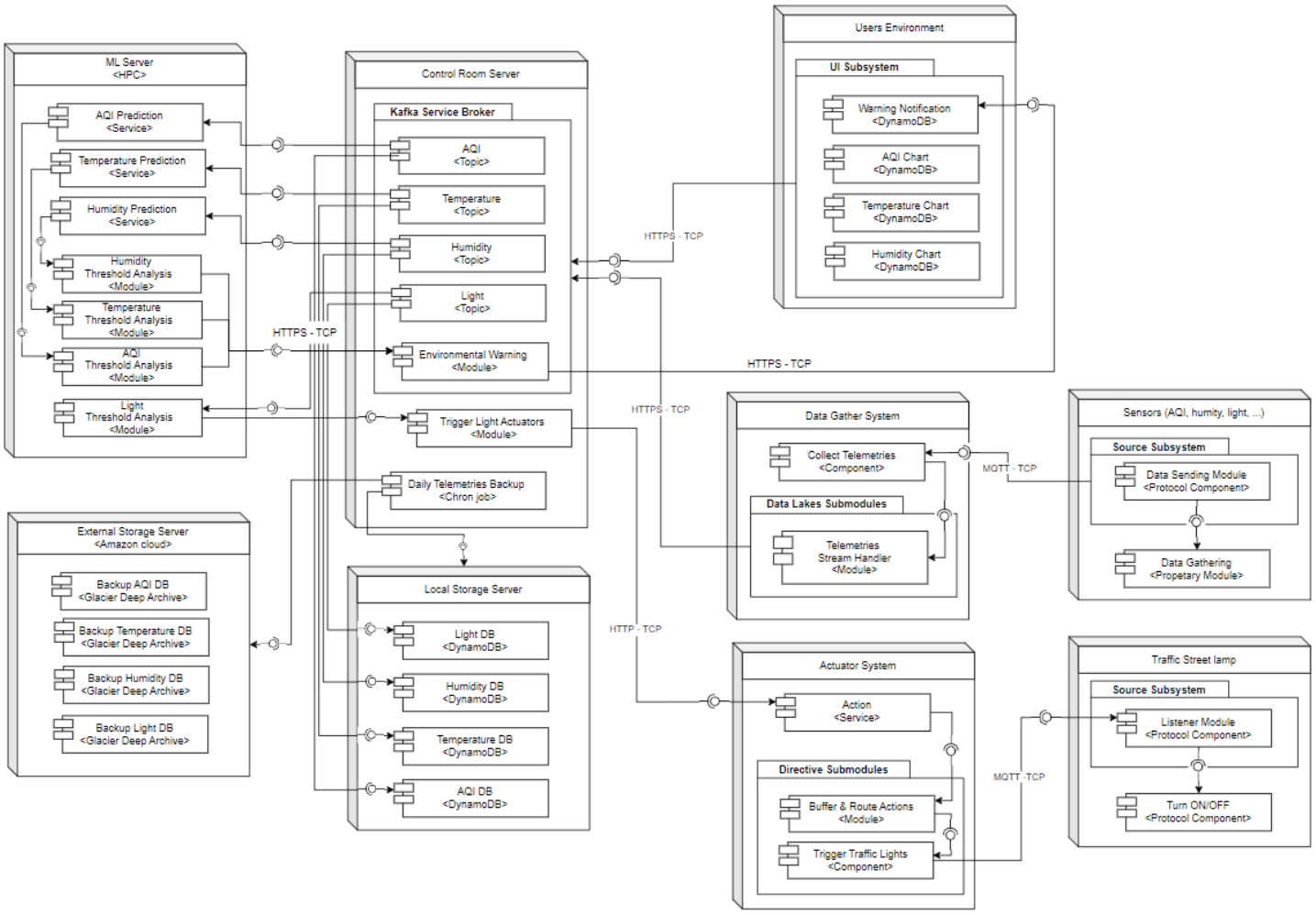
    - [Frequency] Forecasted and updated (current vs predicted) each 10 minutes
- Particular attention on the threshold system and that can analyze each environmental variable and if some elements are above certain limits, the system automatically send the warning to the users.
    - [Frequency] Once each 10 min

General Idea Smart Environment



The idea behind Smart Environment is that:
Peoples can watch the environmental situation (both actual and predicted) like AQI, temperature, etc. directly in the app.
The data in order to accomplish this are generated by various sensor spread throughout the city. Due to the large amount of sensor, some data gathers and actuators are necessary to collect and sand data. The actuator is needed because the streetlamps are automatically (locally) triggered. The triggering is made by a threshold analysis, which is directly made by the ML server also during predictions.
Moreover, when some variables are above certain threshold, it automatically sends the warning information at the users.
The whole intercommunication system is handled by the control room which has also the responsibility to store and flush the data in order to retrieve them in the future (ML training purposes or statistics).

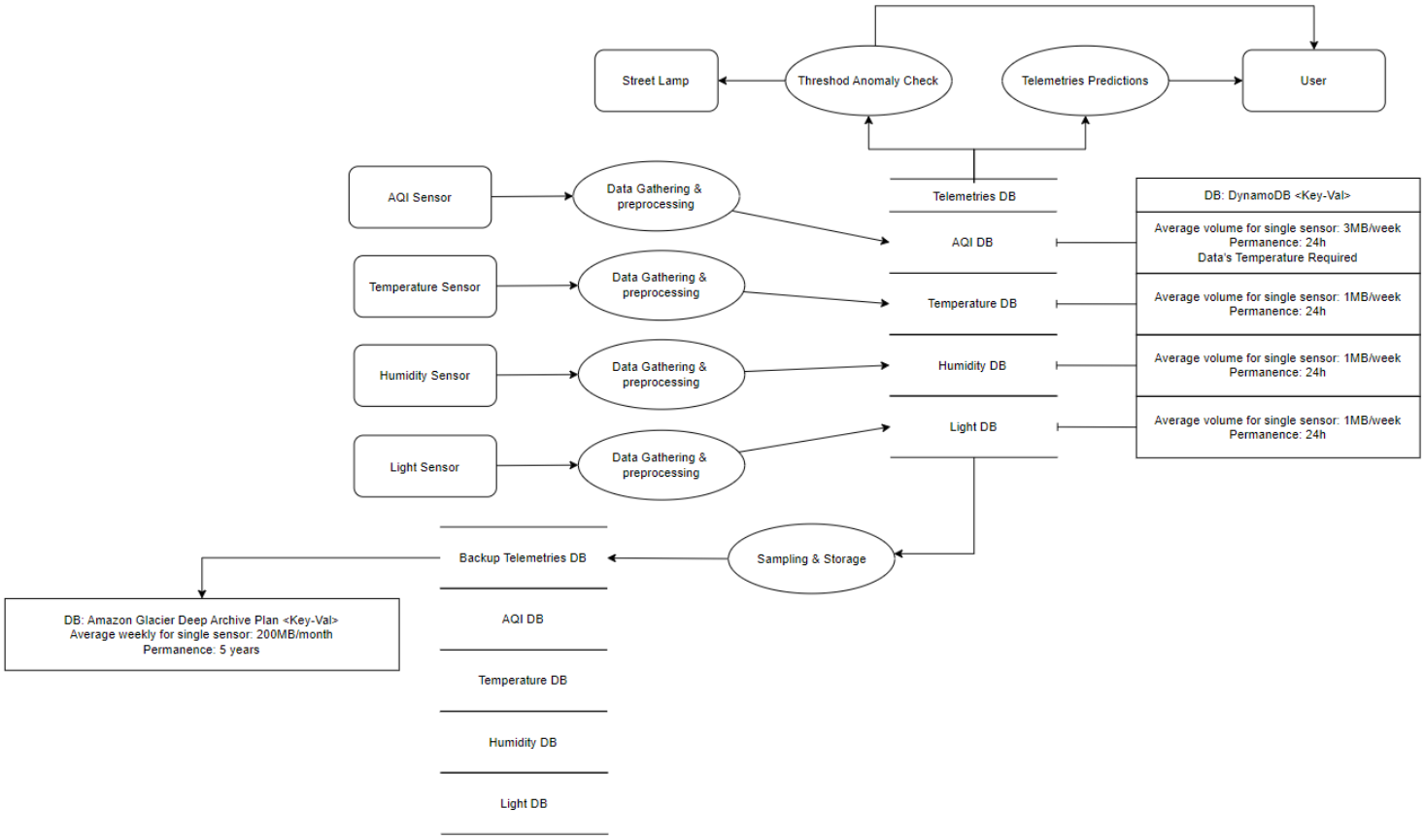
C.D.D. & D.F.D. Smart Environment



The following system reflects the general idea proposed:

1- The **Users Environment** can access through the application at the environmental situation about current and predicted variables.
2- The **Sensor** (general way) can just send the data to the data gather system.
3- The **Data Gather System** is responsible for getting the vast amount of data coming from the sensors and send it to the right topic in the control room.
4- The **Control Room** handles the source channels topic using a *Kafka service broker* module. This module relies on four different topics: AQI, Temperature, Humidity, Light and Warnings. The data that comes from data gather are sent to the ML server in order to make some predictions. If the ML system notes some variable above the threshold, it sends the event to the control room which writes to its warning topic the event in order to notify the user. Moreover, this system can also be applied for the streetlamps that can be (locally) activated when the light sensors of some areas note changing below some thresholds. In this scenario, the triggering action is sent to the local actuator. Another feature is the daily telemetries backup which pick the daily data from the local system, takes the daily average of the sensor's measurement and push into the external storage like sort of backup (which can be used for further investigation).
5- The **Actuator System** handles the triggering action from the control room through the streetlamps on the streets.
6- The **ML Server** is designed to be an HPC which purpose is the data forecasting for each one of the telemetries. Meantime it also equipped with some modules that executes the threshold analysis.
7- The **Local Storage Server** handles the daily sensor data.
8- The **External Storage Server** (Amazon) is used as backup agglomerative sensor service.

The D.F.D. is shown below



Analysis of costs

In order to build this kind of system the following choices are made for a polyglot DB environment:

1- For the telemetries we choose to have a DynamoDB in order to store the latest weekly telemetries measured by each sensor. The cumulative weekly data volume is on average 2MB each sensor if we think to have more or less 200 sensor, we will have a weekly data stream of 400MB. We are interested to store the latest daily telemetry so each day the amount of data will be more or less 60MB a day.

   a. DynamoDB chosen is the one available on LocalStack which can easily be settled in our local environment (this allows to apply custom storage criteria and we do not rely on amazon external service). For further information check this link: DynamoDB | Docs (localstack.cloud)

   b. It is always possible to apply some Amazon external services since the data are not so big, but the major pricing quota depends on total throughout required and how mani I/O operations (which can be millions per months). For this reason, we prefer a custom handled architecture system.

2- The telemetries have also some backup criteria for analytical purposes. We can store this data in an external S3 system, which can be Amazon Glacier Deep Archive. The amount of data has to be pre-sampled in order to save just the average hour telemetry, this tells us that for each sensor we must keep just 24 values. Since we supposed to have 200 sensors, we would get a volume of 4800 information, if we supposed each value of one value is 280KB, we have that the total daily amount is of 1.3GB at most. This bring us with the idea that each month the data generated are more or less 50GB. This means that for 5 years the amount of data to store is less than 4TB.
   A global estimation is made Amazon Simple Storage Service (S3) (calculator.aws):

   a. Unit conversions
      i. S3 Glacier Deep Archive storage: 4 TB per month x 1024 GB in a TB = 4096 GB per month
      ii. S3 Glacier Deep Archive Average Object Size: 16 MB x 0.0009765625 GB in a MB = 0.015625 GB
   b. Pricing calculations
      i. 4,096 GB per month / 0.015625 GB average item size = 262,144.00 unrounded number of

objects
ii. Round up by 1 (262144.0000) = 262144 number of objects
iii. 262,144 number of objects x 32 KB = 8,388,608.00 KB overhead
iv. 8,388,608.00 KB overhead / 1048576 KB in a GB = 8.00 GB overhead
v. 8.00 GB overhead x 0.00099 USD = 0.00792 USD (Glacier Deep Archive storage overhead cost for metadata)
vi. Glacier Deep Archive storage overhead cost: 0.00792 USD
vii. 262,144 number of objects x 8 KB = 2,097,152.00 KB overhead
viii. 2,097,152.00 KB overhead / 1048576 KB in a GB = 2.00 GB overhead
ix. Tiered price for: 2.00 GB
x. 2 GB x 0.023 USD = 0.05 USD
xi. Total tier cost = 0.046 USD (S3 Standard storage overhead cost)
xii. S3 Standard storage overhead cost: 0.046 USD
xiii. 4,096 GB per month x 0.00099 USD = 4.05504 USD (Glacier Deep Archive storage cost)
xiv. Glacier Deep Archive storage cost: 4.05504 USD
xv. 0.00792 USD + 0.046 USD + 4.05504 USD = 4.10896 USD (Total Glacier Deep Archive storage cost)
xvi. 30 requests x 0.00005 USD = 0.0015 USD (Cost for PUT, COPY, POST, LIST requests)
xvii. 30 requests x 0.00005 USD = 0.0015 USD (Cost for Lifecycle transitions)
xviii. 1.30 GB per month x 0.02 USD = 0.026 USD (Cost for Glacier Deep Archive Data Retrieval (Standard))
xix. 30 GB per month x 0.0025 USD = 0.075 USD (Cost for Glacier Deep Archive Data Retrieval (Bulk))
xx. 4.10896 USD + 0.0015 USD + 0.0015 USD + 0.026 USD + 0.075 USD = 4.21 USD (S3 Glacier Deep Archive cost)

S3 Glacier Deep Archive cost (monthly): 4.21 USD

Final Consideration

For the processing units we have made these choices:
For the control rooms, each Smart-Topic has its own main server that handles the communications with different kind of protocols.

The baseline HW requirements for each control room will be:

- Number of Support clone servers (w/t replicas): 5
- CPU Cores per Server: 64
- RAM per Server (GB): 256
- Storage per Server (TB): 10
- Network Bandwidth (Gbps): 40
- Power Supply (kW): 2.5
- Cooling Capacity (kW): 3.0
- Backup Power (UPS) Capacity (kVA): 20
- Physical Space (Rack Units): 4

For the other support system spread throughout the city (data gathers and actuators) will have:

- CPU Cores: 8
- RAM per Server (GB): 24
- Storage per Server (GB): 200
- Network Bandwidth (Gbps): 50
- Backup Power (UPS) Capacity (kVA): 60

For the part data-related i.e. storage criteria we have preferred:

- For the disposable data a local DB solution
- For backup data, we proceed to have the systems as most compliant as possible (choosing AWS solution)

Both are done following as (monthly) low-cost strategy whenever is possible or required.