CitiObs

CitiObs - ENHANCING CITIZEN OBSERVATORIES FOR HEALHTY, SUSTAINABLE, RESILIENT, AND INCLUSIVE CITIES

D2.1 TECHNICAL ARCHITECTURE

AUTHORS: JOAN MASÓ AND NÚRIA JULIÀ (CREAF)

DISCLAIMER

This document contains material, which is the copyright of certain CitiObs beneficiaries, and may not be reproduced or copied without permission.

The information appearing in this document has been prepared in good faith and represents the views of the authors. Every effort has been made to ensure that all statements and information contained herein are accurate; however, the authors accept no statutory, contractual or other legal liability for any error or omission to the fullest extent that liability can be limited in law.

This document reflects only the view of its authors. Neither the authors nor the Research Executive Agency nor European Commission are responsible for any use that may be made of the information it contains. The use of the content provided is at the sole risk of the user. The reader is encouraged to investigate whether professional advice is necessary in all situations.

No part of this document may be copied, reproduced, disclosed, or distributed by any means whatsoever, including electronic without the express permission of the CitiObs project partners. The same applies for translation, adaptation or transformation, arrangement or reproduction by any method or procedure whatsoever.

COPYRIGHT MESSAGE

ACKNOWLEDGEMENT

Funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Research Executive Agency (REA). Neither the European Union nor the granting authority can be held responsible for them

DOCUMENT DESCRIPTION

Delivery date:	15/11/2024		
Type*:	Other	**Dissemination Level**:**	PU - Public
Contributing WP:	WP2		
Lead Partner Organisation:	CREAF		
Lead author(s):	Joan Masó (CREAF) Núria Julià (CREAF)		
Contributor(s):	Andreas Matheus (SECD) Mirjam F. Fredriksen (NILU) Oscar Gonzalez (IAAC) Sjoerd van Ratingen (RIVM) Philipp Schneider (NILU) Nikos Kekatos (DRAXIS)		
Reviewer(s):	Nuria Castell (NILU)		
Abstract:	This deliverable is a collection of UML diagrams with the general architecture of the project. That is the reason why the deliverable is of the “other” type. The purpose of this document is to accompany the diagrams with an explanation of their meaning. Standard APIs for exchanging air quality data are used to communicate data between levels. The technical architecture illustrated in the UML diagrams and explained in this document focuses on the components used or developed during the project and the way they connect and interact with each other. It introduces the overall architecture in a core diagram. The architecture is comprised of different levels that are explained section by section in consecutive chapters. The technical architecture prioritizes the use of standard protocols and is based on replaceable building blocks. A final chapter describes how this		

	approach increases the sustainability of the CITIOBS architecture and the exploitability of the individual building blocks in other initiatives like GEOSS.

VERSION LOG

Version	Date	Partner	Content and changes
0.1	09/06/23	Joan Masó (CREAF)	Document created
0.2	10/09/23	Joan Masó (CREAF)	Definition of the content and the sections of the document
0.3	10/11/23	Joan Masó and Núria Julià (CREAF)	Filling the content of sections and acronyms
0.4	23/11/23	Núria Julià (CREAF)	Figures, diagrams and references
0.5	07/12/23	Andreas Matheus (SECD)	Revision and update
0.6	08/12/23	Joan Masó (CREAF)	Clarification that this document accompanies a set of UML diagrams, and the latter is the actual deliverable
0.7	11/12/23	Joan Masó and Núria Julià (CREAF)	Revision, update and improving some UML diagrams
0.8	12/12/23	Nuria Castell (NILU)	Review
1.0	15/12/23	Nuria Castell (NILU)	Upload to the EC platform
1.1	13/11/24	Joan Masó and Núria Julià (CREAF), Philipp Schneider (NILU), and	Revision and update of version 2: clarification of some sections based on reviewers request and

		Nikos Kekatos (DRAXIS)	improving the general architecture with lessons learnt and new views
1.2	14/11/24	Nuria Castell (NILU)	Review
2.0	15/11/24	NILU	Upload to the EC platform

CITIOBS

CitiObs is a four-year project funded under Horizon Europe by the European Commission. CitiObs will consolidate and apply tools and practice-based knowledge for co-creating data, knowledge and local action via Citizen Observatories (COs): these tools will enhance existing and new citizen observatories to engage people from a diverse range of communities, add value to environmental observations in the urban context, increase and validate citizen observations of the urban environment as part of the existing in-situ Earth Observation systems, co-create inclusive local actions for sustainability and ensure that CO data contributes to research and policy development towards the objectives of the European Green Deal. To ensure broad use, the CitiObs tools and approaches will be developed in co-creation with COs in 5 Frontrunner cities, finetuned with 30 Implementer cities and showcased to 50 Fellow cities.

CitiObs will support citizen observatories in distinct cities to create/enhance/or scale up inclusive and diverse citizen observatories, fostering in particular an active role of citizens in the observation of the urban environment using low-cost sensor technologies and wearables, with a particular focus on air quality and related environmental measures. CitiObs will formalise, valorise and legitimise the role of citizen observations.

The CitiObs methodology of using a large-scale demonstration, co-design and coaching approaches with CO stakeholders (citizens, scientists, policy/decision makers) in 5+30+50 cities in Europe explicitly builds on the Responsible Research & Innovation (RRI) dimensions as founding principles. Ethics consideration will be addressed consistently across all Work Packages.

- WP1. Social dimensions of Citizen Observatories for transition governance
- WP2. Tools, Technologies, and Data Services for Citizen Observatories
- WP3. Co-creation of data and actions for healthy, sustainable and resilient cities with Citizen Observatories
- WP4. Impact creation, Communication, Dissemination and Exploitation
- WP5. Project management
- WP6. Ethics

EXECUTIVE SUMMARY

This deliverable is a collection of UML diagrams with the general architecture of the project. That is the reason why the deliverable was initially defined as “other” type. The purpose of this document is to accompany the diagrams with an explanation of their meaning. The collection of UML diagrams and this document describe the overall CitiObs technical architecture and standard APIs for connecting the components. The technical architecture includes already existing components, and components to be developed during the project. The overall architecture is separated in several way. On one hand there is a separation between the FrontEnd and the BackEnd. On the other hand, different data levels of processing are explained to better illustrate the components and their interconnections.

The architecture core UML diagram introduces the overall architecture which is broken into parts that provide further details. The technical architecture prioritizes standard protocols and is based on replaceable building blocks.

The main objective of the architecture components is to produce data of incremental added value. The level 1 organizes and stores raw data coming from Citizen Observatories. The level 2 (ValAir) introduces data quality, semantic interoperability and homogenizes the data resulting in Analysis Ready Data. The level 3 (VirtualAir) provides access to some data providers in a unified way (several ValAir providers in one end point) and the level 4 (MapAir) transforms observational data and ancillary data into maps for decision making (resulting in Decision Ready Information).

The architecture also includes additional components that manage data. There is an asynchronous notification system (implemented as a PubSub service) for generating alerts and distributing data updates. Another system is a Geospatial User Feedback Service that implements a way for the users to comment about the data and share experiences using it. When necessary, an authentication service (Authenix) can be used to manage user information.

The architecture also shows the different possibilities for the FrontEnd specifically targeting user types such as decision makers (Decision Support System) or local communities (TAPIS and Python libraries for Orange). The architecture ensures that components developed by different participants and the FrontEnd and the BackEnd in the project can work together based on open standards such as OGC SensorThings API and OGC SensorThings API plus (STAplus).

The architecture increases the sustainability of CitiObs and the exploitability of the individual building blocks in other initiatives such as GEOSS, the Green Deal Data Space and the Copernicus Services.

TABLE OF CONTENTS

INDEX OF FIGURES

INDEX OF TABLES

ACRONYMS

Acronym	Full name
AQI	Air Quality Index
API	Application Program Interface
ARD	Analysis Ready Data
CAMS	Copernicus Atmosphere Monitoring Service
CO	Citizen Observatories
CS	Citizen Science
DRI	Decision Ready Information
DSS	Decision Support System
FROST-Server	FRankhofer Opensource SensorThings-Server
HTML	HyperText Markup Language
HTTP	Hypertext Transfer Protocol
IoT	Internet of Things
ISO	International Organization for Standardization
JSON	JavaScript Object Notation
LCS	Low-Cost Sensors
MQTT	Message Queuing Telemetry Transport
OData	Open Data Protocol
OGC	Open Geospatial Consortium

REST	REpresentational State Transfer
RDF	Resource Description Framework
SPARQL	SPARQL Protocol and RDF Query Language
STA	Sensor Things API
STAplus	Sensor Things API Plus
SWE	Sensor Web Enablement
UML	Unified Modeling Language
URL	Uniform Resource Locator
WMS	Web Map Service
WP	Work package

INTRODUCTION

1.1 Purpose of the document

The purpose of this document is to accompany and describe the deliverable D2.1 that is a collection of UML diagrams with the general architecture of the project. That is the reason why the deliverable is of the "other" type. The collection of UML diagrams and this document introduce the CitiObs architecture, components and their accessibility for external applications and services; the "outside" view.

The main architecture structures the components in four levels that reflect the evolution of the data from raw data from sensors to Decision Ready Information (DRI). It describes the data flow between the levels and how to external users can access the CitiObs data at the different levels.

1.2 Scope of the document

This document presents how external users can access data and services. It does not provide full details about the internal architecture and its individual components.

This document reflects the current view of the project based on the current state of the art. The technical architecture may evolve during the project timeline.

For better evaluation of the use of open standards, this document also provides a brief introduction to the relevant standards that are used to exchange data between components, and external applications and services.

1.3 Overview of the technical architecture

The aim of the architecture is to harmonize and integrate air quality data sources from different providers in a system that can apply data quality filters and add semantics to the data for providing access to raw sensor data, Analysis Ready Data (ARD) and Decision Ready Information (DRI). The CitiObs data is combined with other sources to provide data services and create and show Decision Ready Information in a Decision Support System (DSS) which is expected to be the main FrontEnd system serving the users.

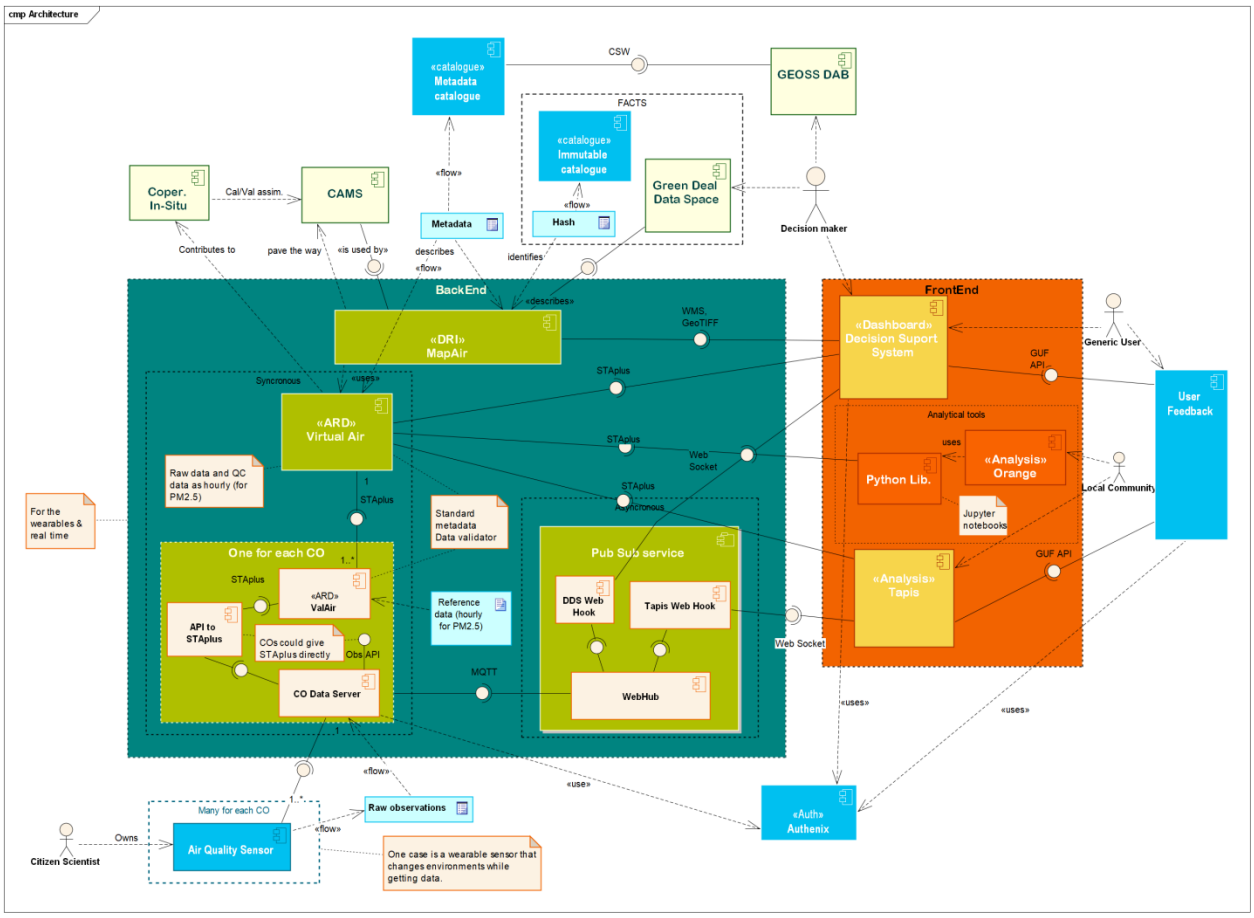

Figure 1: Overall architecture diagram

The overall architecture as illustrated in Figure 1 comprises a CitiObs BackEnd that collects and provides the data to the CitiObs FrontEnd. The BackEnd includes separate components from different Citizen Observatories (COs) that undertake the data collection as input for the BackEnd that is composed by a synchronous dataflow and an asynchronous notification service, as well. For illustrating the direct use of OGC STAplus standard for sensor data collection, the BackEnd also contains STAplus compatible data components.

The CitiObs BackEnd is used by the CitiObs FrontEnd components such as the Decision Support System, the TAPIS, and analytical tools implemented in Phyton libraries that can be used e.g. in Jupyter Notebooks.

Low-Cost Sensors (LCS) can address gaps in official air quality networks, especially in urban settings. However, it is necessary to provide a harmonized framework for validating air quality data collected by diverse citizen-operated LCS networks. The framework proposed in CitiObs is a novel, scalable solution for harmonizing, standardizing, applying quality control, and correcting pollutant measurements across varied LCS networks. As one of the first continental-scale

implementations of rigorous quality control and correction processes for citizen-sourced data, CitiObs sets a strong foundation for integrating crowd-sourced data into formal air quality assessments. In that respect, focusing on the data flow, the architecture can be divided into levels of data processing, from the raw data (level 1) to Decision Ready Information (level 4). At each level above level 1, data is transformed, harmonized and improved. The overall data flow is presented in Figure 2.

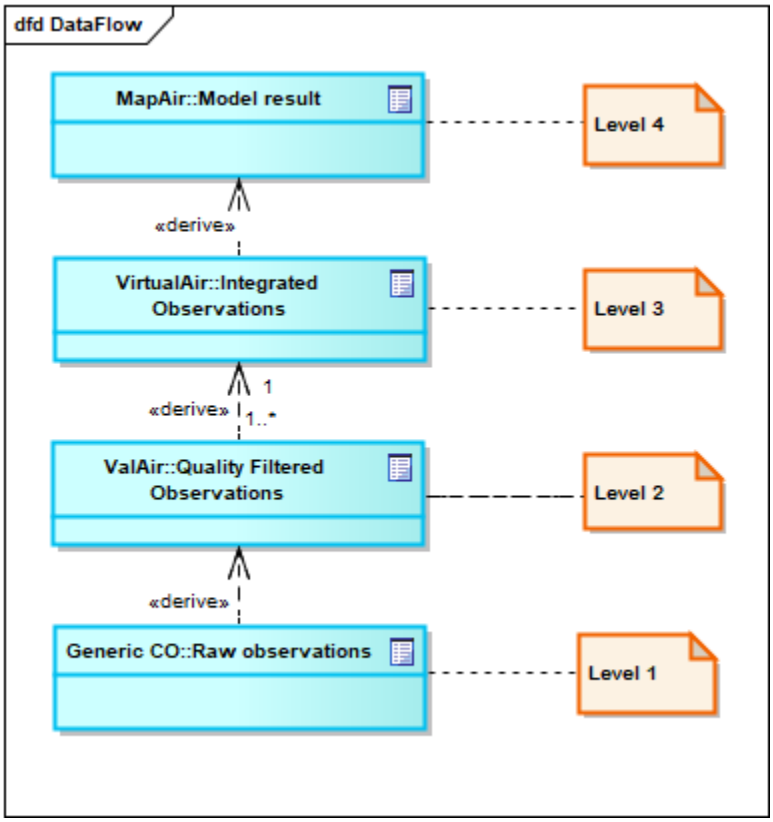

Figure 2: Data flow diagram and components levels

At level 1, the raw sensor data is collected via existing APIs operated by participating COs. For further processing, the stored data from level 1 can be accessed via the OGC STAplus standard. For evaluating the direct use of STAplus for sensor data collection and storage, SECD operates an additional Data Server as a deployment of FROST-Server.

At level 2, the ValAir component produces quality controlled, calibrated and validated data, allocating a data quality level to the data according to their “plausibility”, this is, the assessed data quality. The resulting data is made available to the next processing level.

At level 3, the VirtualAir component leverages level 2 data from the different COs and the SECD FROST-Server and harmonizes ARD data.

At level 4, the MapAir component produces gridded DRI by combining level 3 with external data from CAMS and other sources, like satellite data and reference data.

The PubSub Service provides a system to generate notifications (a.k.a. alerts or data updates) to the FrontEnd.

The CitiObs FrontEnd (DSS-Graphical Interface, TAPIS and analytical tools for local communities) takes information from several levels and presents it in a Graphical User Interface.

1.4 Structure of the document

The document is organised as follows:

- Section 1 - Introduction: Description of the purpose and scope of the document and its structure
- Section 2 - Level 1: From sensors to CO services
- Section 3 - Level 2: Quality Control and Validation of the Data (ValAir)
- Section 4 - Level 3: Data integration in a single-entry point (VirtualAir)
- Section 5 - Level 4: Creation of Decision Ready Information (MapAir)
- Section 6 – Asynchronous flow: Notification Service: STA Pub Sub Service
- Section 7 - CitiObs FrontEnd: Graphical User Interfaces
- Section 8 - Connection to other initiatives
- ANNEXES: Introduction to OGC SensorThings API and SensorThings API plus standards

SECTION 2: Level 1: From sensors to CO services

Commonly, each Citizen Science (CS) project or CO typically develops their own platforms in a way to be most efficient and convenient for their community. This approach indirectly introduces the creation of data silos. Even if each CS or CO offers their data on the Internat using their API, this is done in a way that maximizes the interest and priorities of their use. Consequently, this results in a myriad of data formats and data sharing protocols. Commonly, the individual sensor measures air quality parameters and sends measurements in raw data format to a CO service that stores these measurements. The services hosted by the COs can be a simple storage facility or may include some degree of quality control to reduce spatial and temporal uncertainties. The CO offers access to historical and current measurements via an API. The project recognizes the resource limitation of some of the individual COs, so that, it is unrealistic to request every individual CO to change their systems into supporting a common API.

To overcome the data isolating via different APIs and data formats, the use of standards is applied. For CitiObs, we have identified three possible approaches:

- The hardware of the sensors themselves could be reprogramed to send the data to the storage service. This communication could be done by STA or STAplus using MQTT.
- The sensor uses a proprietary communication schema to communicate with the COs data services, but the stored data is offered by the service in an open standard, e.g. STA or STAplus.
- Neither the sensor nor the service offers access to the data using an open standard. Instead, the data is imported by a bigger system that can offer data via STA or STAplus.

For the CitiObs architecture, we are facing all three approaches. The CitiObs Data Server offers the direct use of STAplus for uploading raw sensor data, data calibration and querying of the data. NILU develops a system called NILU Sensor Data Platform (NSDP) (previously called iFLINK) that falls into the second approach where the raw data is collected via an established system, but the stored sensor data will be made available via STAplus (see Figure 3). RIVM has a system that also falls into the second approach where established systems collect the sensor data but a STA or STAplus API allows querying the data. Finally, FabLab operates a system accordingly to the third approach: the data is collected and processed internally but made available via the STAplus API developed by NILU.

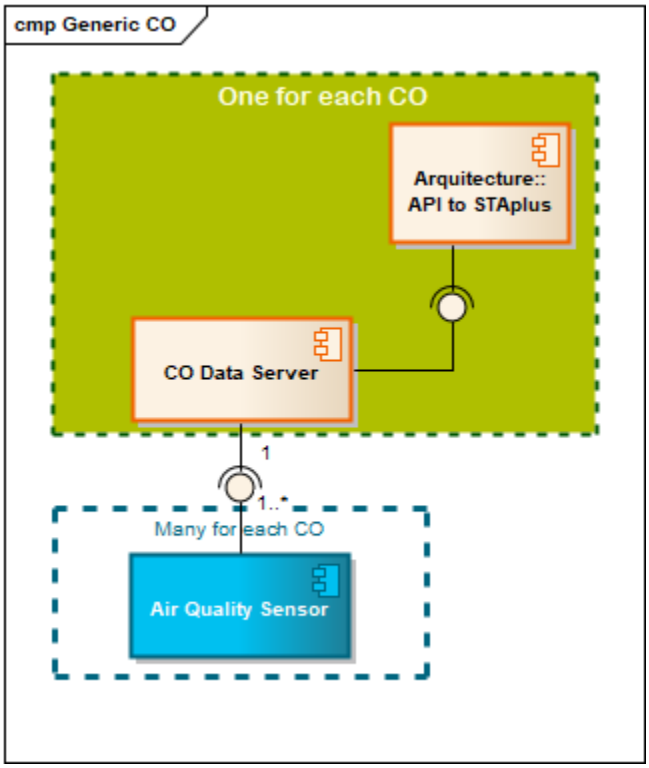

Figure 3: In level 1 sensor data is recorded in independent COs services

The adoption of a single API (STA) makes the creation of aggregated datasets easier at the next level.

SECTION 3: Level 2: Quality Control and Validation of the Data (ValAir)

As currently many COs do not use OGC STA standards, CitiObs has a dedicated task to provide guidelines, and code for illustrating how to translate the APIs to provide sensor data to STAplus API. At the same time and in this level, CitiObs will provide tools for the application of standard metadata containing information related to:

- data quality level,
- processing level,
- semantic meaning of the variables measured and its units of measure,
- context under which the data was collected,
- what information can and cannot be gained from the data,
- other details about the data collection process, processing steps, and quality control process.

This will facilitate both the discovery and the semantic interoperability among datasets and will make possible later integration in external infrastructures such as GEOSS, INSPIRE or Copernicus. The data collected through the sensors and wearables will be stored in the local COs databases when available (in compliance with INSPIRE) and latter included into STAplus compatible data server or, when the COs do not have their own data repository, it will be stored in the NILU Sensor Data Platform (NSDP), a distributed and scalable computing/storage infrastructure for sensors hosted at NILU. NSDP will be compatible with STAplus so metadata and semantic interoperability among datasets is ensured, and data will be offered with a single data sharing and access protocol.

Since every CO has a different data quality control system in level1, level 2 will execute a set of common routines that will be personalized for each CO: ingestion, cleaning and calibration. Once the data has been through the different steps, it will become part of a good quality dataset and with the semantics harmonized with other datasets.

There will be one ValAir component detailed in Figure 4 for each CO platform (assuming that some COs will not share a common platform) such as NSDP from NILU. Exceptions are: For Smart Citizen from IAAC, adoption is not necessary as their sensor data is made available via

NILU system. The RIVM system is already serving STA. The output of each ValAir component is always Analisys Ready Data that can be queried using STAplus.

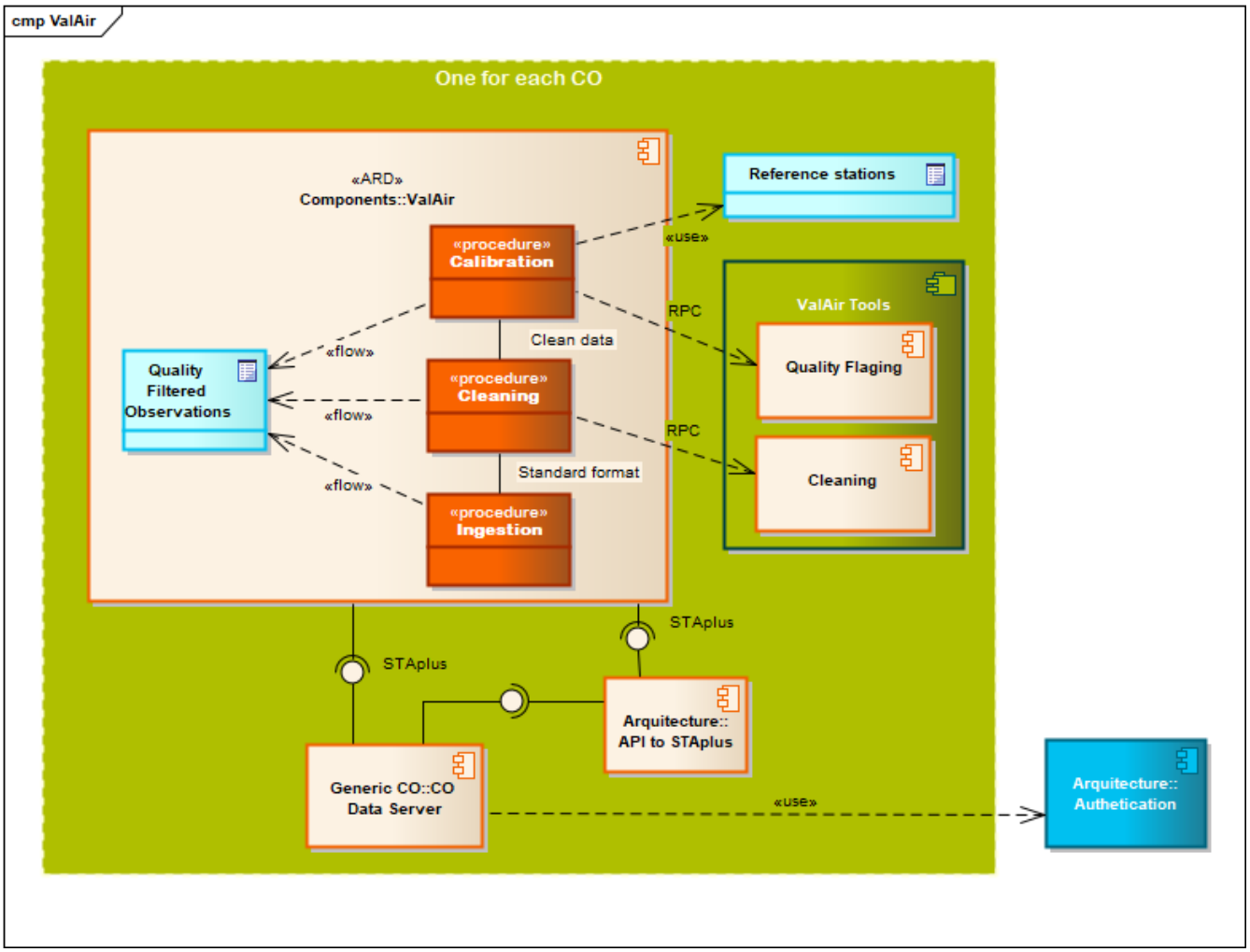

Figure 4: In the level 2, the COs data is filtered and validated

One important aspect of level 2 is that data is semantically enriched. To achieve that, all outputs of a level 2 component are tagged with the right vocabulary. The concepts that need to be part of this activity are:

- A set of URIs that define the variables that were captured.
- A set of URIs of the Units of Measure corresponding to those variables.
- A set of URIs corresponding to quality flags associated to the data.

For example, these are variable URIs used in the Smart Citizen Kit:

"description": "Air Temperature"
"definition": http://vocabs.lter-europe.net/EnvThes/22035

"description": "Relative Humidity"
"definition": http://vocabs.lter-europe.net/EnvThes/21579

"description": "Ambient Light"
"definition": https://qudt.org/vocab/quantitykind/LuminousExposure,

"description": "Noise Level"
"definition": https://qudt.org/vocab/quantitykind/SoundExposureLevel

"description": "Barometric Pressure"
"definition": https://vocabs.lter-europe.net/EnvThes/22060

"description": "Particulate matter with an average aerodynamic diameter of up to 1 micrometer"
"definition": https://www.iqair.com/us/newsroom/pm1

"description": "Particulate matter with an average aerodynamic diameter of up to 2.5 micrometers"
"definition": https://www.eea.europa.eu/help/glossary/eea-glossary/pm2.5

"description": "Particulate matter with an average aerodynamic diameter of up to 10 micrometers"
"definition": https://www.eea.europa.eu/help/glossary/eea-glossary/pm10

SECTION 4: Level 3: Data integration in a single-entry point (VirtualAir)

Currently there is no data service that allows querying data from different COs in Europe or worldwide to create an extended analysis, e.g. air quality maps at the European level, or merging data from COs in air quality with COs in biodiversity to research, for example the connection between pollution levels and the declining of bee population. The VirtualAir component uses STAplus, common metadata and data validation terminologies (applied in level 2) to deliver a service that aggregates data from different COs in a single endpoint. Instead of creating permanent consolidated datasets, the component will implement a distributed query among the integrated COs in their original repositories.

The virtual aggregated dataset is respectful with the original data license. Trust in the data is possible by propagating data validation levels (data quality), accumulating geospatial user feedback and added recognition of the originators.

There is only one VirtualAir component that integrates data from different COs called ValAir in the previous level 2 diagram. VirtualAir will make use of STAplus as an input and output standard protocol. The output of the ValAir is Analisys Ready Data queryable using STAplus.

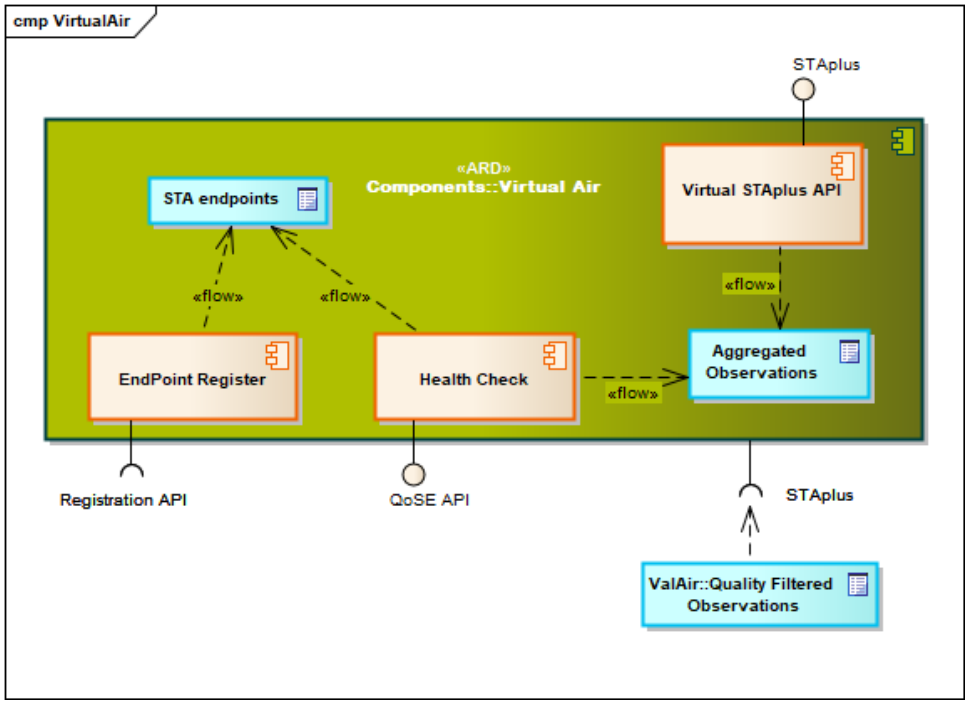

Figure 5: In the level 3, data is integrated in a single-entry point

As illustrated in Figure 5, the interaction between level 2 (ValAir) and level 3 (VirtualAir) is all standards-based levering STAplus.

It is important to keep in mind that STAplus supports views or collections of observations and enables thereby the efficient structuring of the level 2 data to be consumed by level 3. Furthermore, the support for licensing of collections and individual observations ensures proper re-use of the observations.

SECTION 5: Level 4: Creation of Decision Ready Information (MapAir)

MapAir comprises of a suite of algorithms and code designed for generating spatially complete gridded maps from point-based observations provided by COs. As input, this system integrates sensor readings with auxiliary information, including model data from the Copernicus Atmosphere Monitoring Service, satellite data, and air quality monitoring stations. The goal is to overcome the limitations of point measurements in the sensors, allowing for the creation of continuous spatial maps of air quality. The project aims to achieve this at two scales: firstly, covering Europe at a 1 km resolution, subject to sufficient sensor data coverage; secondly, focusing on individual cities with higher resolution. To address the issue of spatial scales, MapAir employs a flexible approach to adapt its resolution based on the availability of sensor data and the specific needs of the target area.

1. For large-scale mapping across Europe, a 1 km resolution is selected to balance the broad coverage with computational feasibility, making the system effective even with varying sensor densities across regions.
2. For individual cities or smaller areas, where a sufficient density and number of CitiObs sensors provide richer data, the set of tools in MapAir can in principle increase the spatial resolution, potentially down to tens or hundreds of meters.

The high-resolution mapping is particularly valuable for localized scenarios, such as:

- pollution hotspots,
- traffic-related emissions, or
- wildfire impacts,

where more detailed spatial information is critical. Maps for these specific cases will be generated only on demand and when sufficient sensor information is available, integrating the sensor data with relevant auxiliary sources to provide targeted, accurate assessments of air quality in those regions.

The output from MapAir is Decision Ready Information consisting of geospatial datasets in gridded formats such as GeoTIFFs, NetCDFs, or other geospatial representations, providing the best estimates of PM2.5 concentrations. The practical application of MapAir involves

demonstrating its capabilities by generating European-scale PM2.5 maps at a 1 km spatial resolution (e.g. annual or monthly averages) and ad-hoc demonstrator maps for specific scenarios like individual cities or wildfire-affected areas. It is important to note that the intended use of MapAir is primarily for demonstration purposes that can inform a future procedure for the CAMS product (see Figure 6). However, there are no plans for operational deployment within the project timeframe.

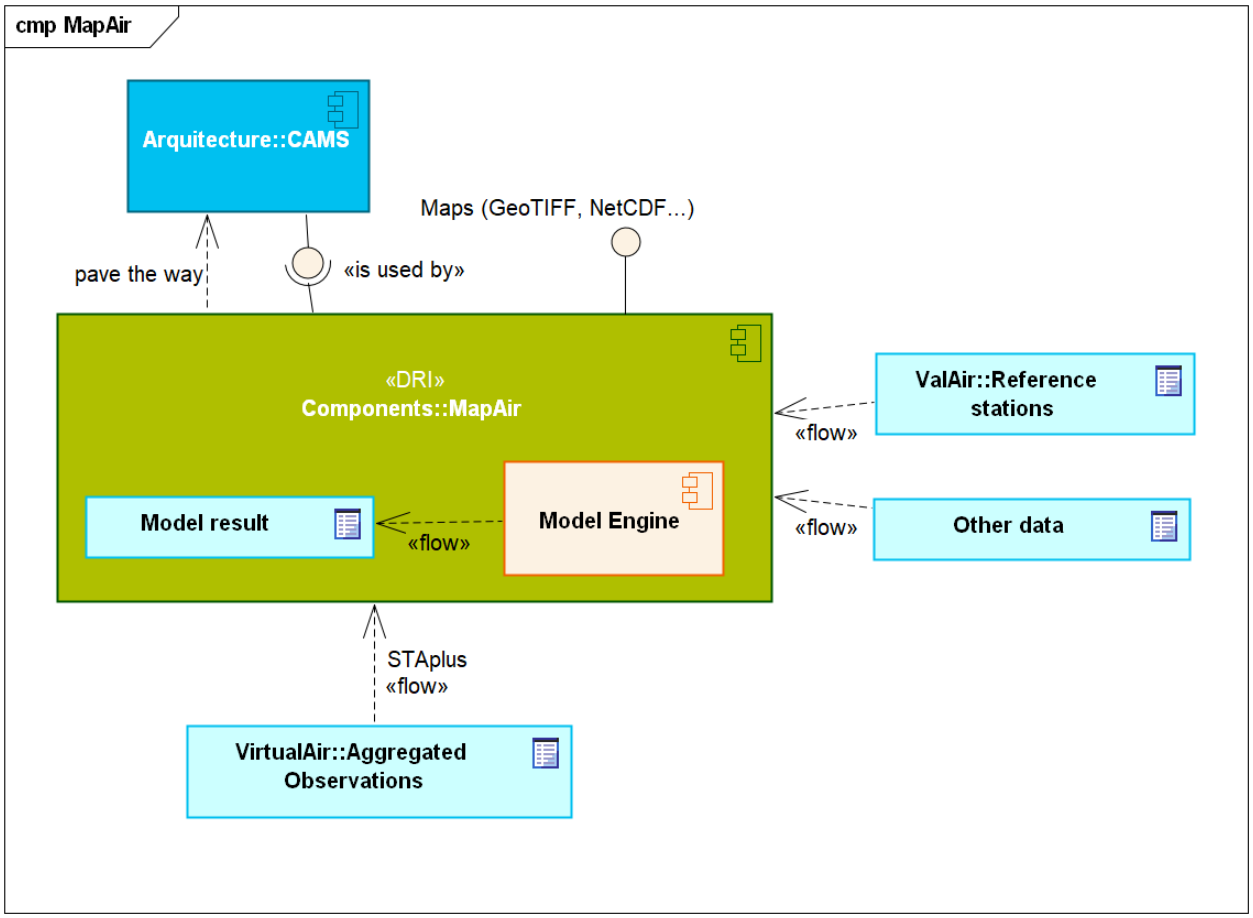

Figure 6: In the level 4, Decision Ready Information is provided

Like in level 3, the structuring of observations into collections with appropriate licensing provided by VirtualAir ensures the proper reuse of the observations in MapAir when producing the DRI products available to the users.

SECTION 6: Notification Service: STA Pub Sub Service

Sections 2 to 4 describes a linear synchronous flow that gives access to time series of that can be queries and sub-setted by space and time intervals. These queries work in the “pull” mode, where the FrontEnd requests data to the BackEnd, as a result of user actions on the GUI (this is how the classical web interactions between web browsers and web servers happen). In this section, we describe a different dataflow that uses the “push” mode. The protocol can be decomposed in two actions:

- Subscription (Sub): As a request from the user, the FrontEnd subscribes to a particular “topic” (that is a subset of the new observations arriving two the system).
- Publication (Pub): Every time that a new observation is received and if is part of the subscribed topic, the subscriber receives a message with the new observation.

For that reason, these kinds of protocols are called PubSub.

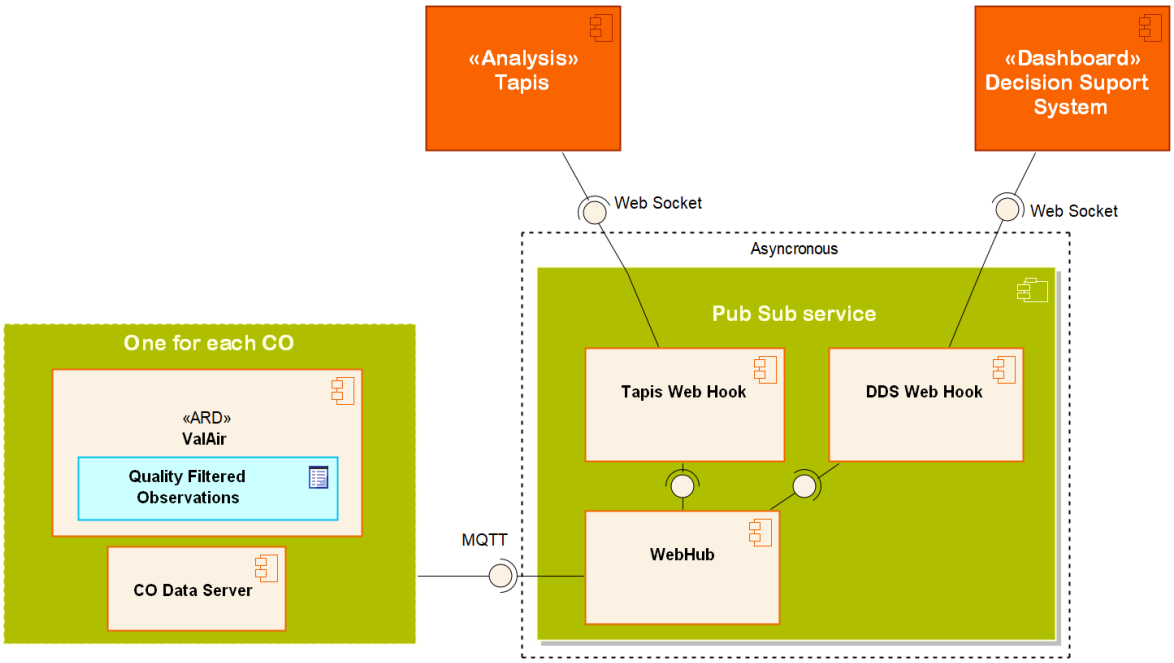

Figure 7: Asynchronous data flow for a notification service.

In case the protocol is implemented in a web application, the protocol requires a new component called WebHook that keeps a permanent connection between the Web-app and the subscriber that is maintained with WebSockets (a new bidirectional protocol adopted in HTML5

that supports “push” mode). The WebHook relays messages to the Web-app when new observations are notified from another component called WebHub. The WebHub acts as publisher of new observations. While the Web-app and the WebHook communicates via HTTP and WebSockets, the WebHook and the WebHub communicate via HTTP. The WebHub and the STAplus service communicate via MQTT. Actually, both the WebHook and the WebHub are only necessary because the web browsers are not capable to directly communicate via MQTT. As a bonus, the WebHub and the WebHook provide scalability as they can act as a proxy for common subscriptions (see Figure 7).

Web-App and WebHook are the client side of the system, and they act as a Subscriber. The Hub and the STAplus Service are the server side of the system, and they act as a Publisher.

This new architecture requires some experimentation and, for that reason, CitiObs made two implementations for the client side. Figure 8 shows the protocol of one of these implementations, where the Web-App establishes a WebSocket connection to a WebHook for each subscription that it needs to do. During the establishment of the WebSocket connection (handshake), the Web-App sends all the information needed for the Subscription directly to the WebHub. In this case a different WebSocket connection is needed for each subscription.

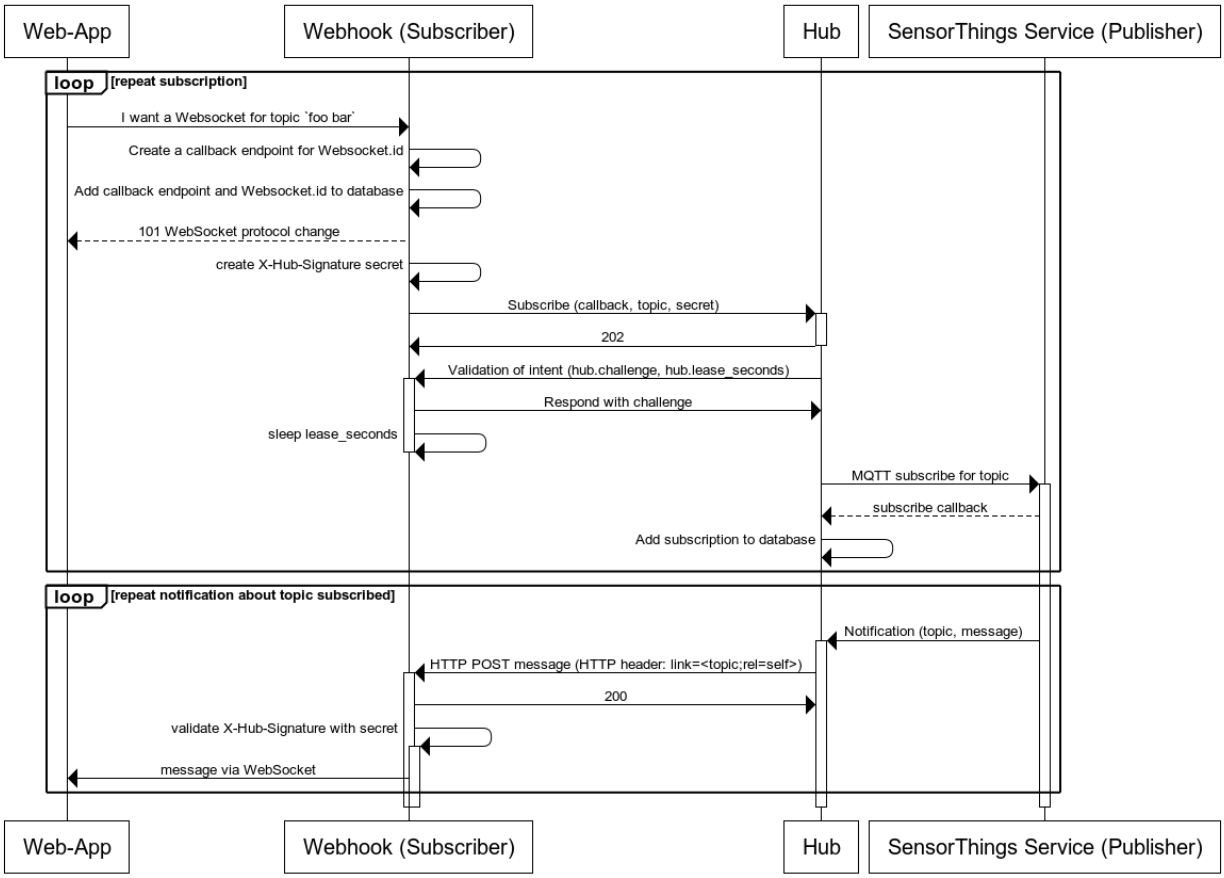

Figure 8: Protocol for a generic Web-App to display notifications (WebHook as a Subscriber).

Figure 9 shows the second implementation where the Web-App establishes a WebSocket connection first. Then, the Web-App sends the subscription request directly to the WebHub.

In both implementations there is a process of Validation of intent between the WebHub and the WebHook to validate the intention of the topic subscription. After that, the WebHub establishes a MQTT subscription to the Publisher (STAplus service).

When a new observation is produced that is compatible to a particular topic, the WebHub receives the notification for the topic with the new information and sends a POST message to the WebHook. The WebHook checks the information and sends it to the Web-App through the WebSocket connection established at the beginning of the process.

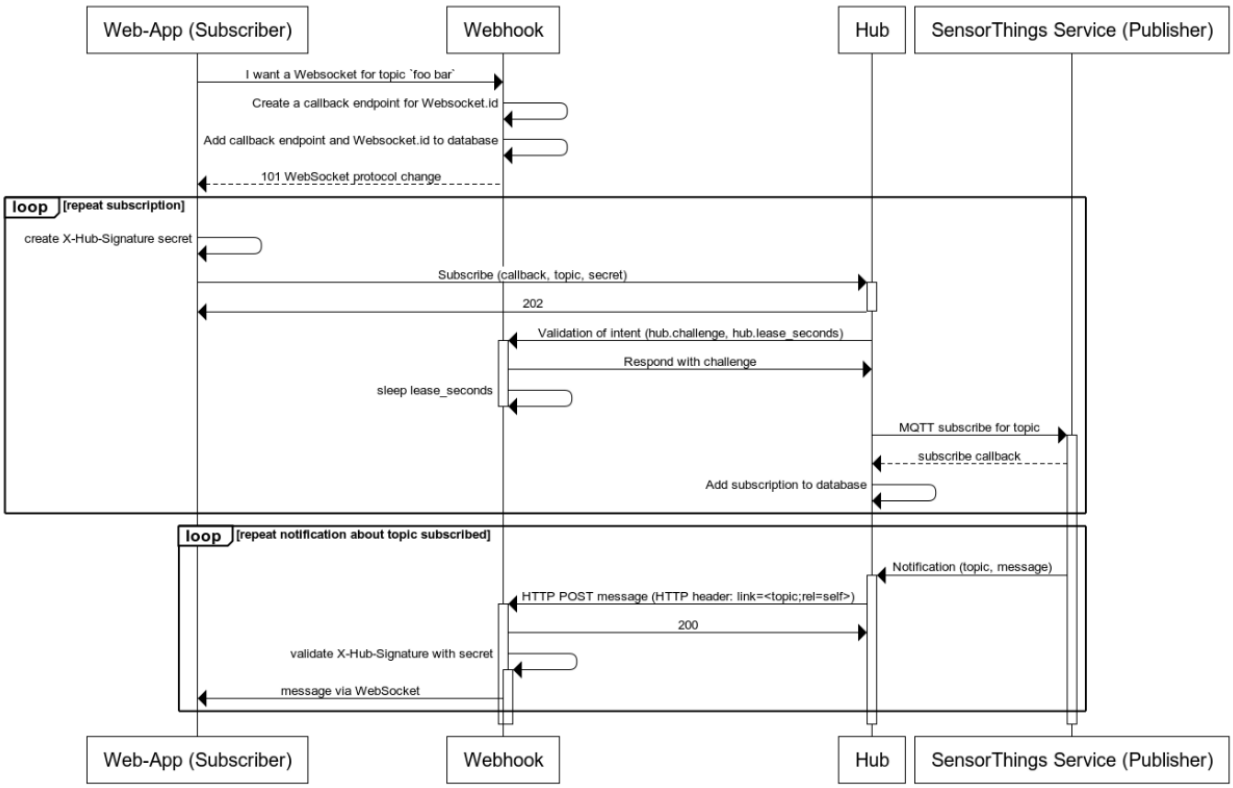

Figure 9: Protocol for a generic Web-App (as a Subscriber) to display notifications.

SECTION 7: Graphical User Interfaces

The separation of responsibilities between the BackEnd and the FrontEnd and the use of internation standard protocols and formats allow for multiple GUIs to be provided to different user profiles. CitiObs provides 3 different GUIs for the data (see Figure 10). The main tool for visualizing the ARD and DRI is the Decision Support System (DSS). In addition to that, CitiObs is developing two other tools targeted at local communities. One is the web app called TAPIS, that allows for easy direct access and query to sensor observations via STAplus, and the second is based on the use of Python libraries that also access STAplus data and that can be directly used in the Orange Data Mining desktop application.

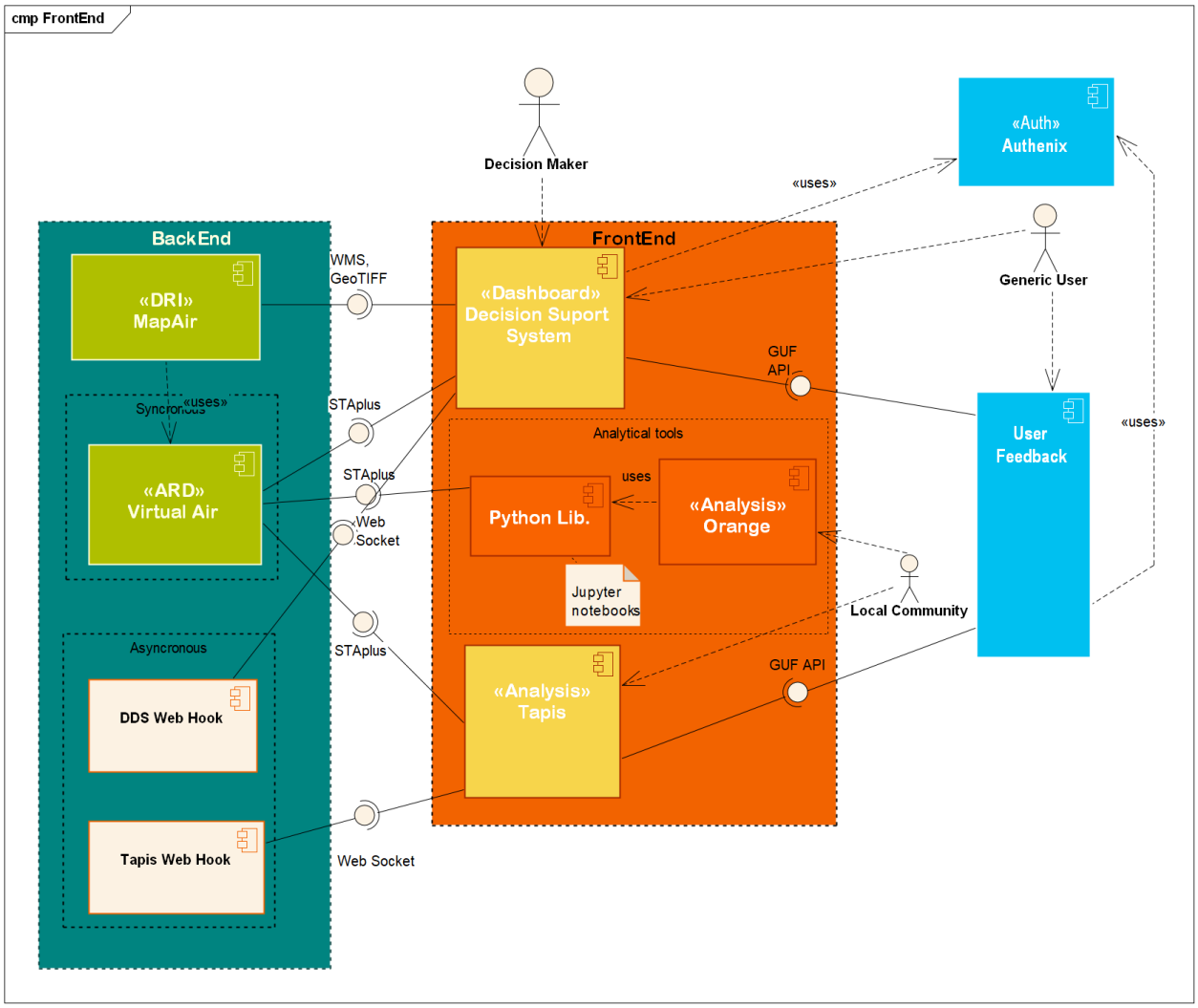

Figure 10: FrontEnd elements and its relation to other components

7.1 Decision Support System and Dashboard

This component is the main user interface foreseen in the project. It allows for the visualization of raw data, Analysis Ready Data and Decision Ready Information. In addition to visualization, it will have tools for data analysis tailored for citizens and professionals. Their internal structure is still not fully decided, as this will be co-designed with the Citizen Observatories participating in the project.

7.1.1 Overview & Design Criteria

To enhance the design of the CitiObs Decision Support System (DSS), we conducted a comprehensive review of existing air quality dashboards and platforms. Our analysis revealed a spectrum of approaches to geospatial air quality visualization, ranging from visually appealing and user-friendly interfaces, such as AirVisual Earth and Plume Labs, to detailed and research-oriented platforms like PurpleAir and OpenAQ.

Each dashboard effectively combines real-time data, interactive maps, pollutant breakdowns, and health advisories, making them invaluable resources for both the general public and experts. Table 1 presents a structured comparison of these air quality dashboards and platforms, highlighting their unique features and strengths. The table emphasizes critical aspects such as community involvement, mobile accessibility, real-time data integration, health guidance, and regional focus. This comparative analysis aims to identify the most relevant features for CitiObs, ensuring that the system meets the diverse needs of its users while providing accurate and actionable air quality information.

Table 1: Comparison of existing dashboards and platforms

Dashboard	Key Features	Strengths	Reference
AirVisual Earth	3D interactive map, real-time Air Quality Index (AQI), pollution movement	Visually engaging, global scope, weather integration	https://www.iqair.com/earth
AirNow (EPA)	Health guidance,	Clear public health advice,	https://www.airnow.gov

	AQI breakdown	U.S.-focused	
Plume Labs	Real-time forecasting, personal air quality monitor	Forecasting insights, mobile-friendly, personalized alerts	https://air.plumelabs.com/en
BreezoMeter	Pollutant details, heatmaps, health/activity suggestions	Health integration, responsive, actionable recommendations	https://www.breezometer.com/air-quality-map/
PurpleAir	Community-based sensors, customizable data, hyper-local	Granular data, community-driven, localized insights	https://www.purpleair.com
OpenAQ	Open-source data, API access, customizable filters	Researcher-friendly, global open-access data	https://explore.openaq.org
World Air Quality Index	Global map, AQI levels, health advice	Easy-to-understand, color-coded AQI, extensive global reach	https://waqi.info
Sensor.Community	Real-time PM10/PM2.5, crowdsourced	Community-driven, highly localized	https://sensor.community/en/

	data		
Clarity Movement	Low-cost sensors, customizable alerts, mobile-friendly	Scalable for urban monitoring, responsive design	https://www.clarity.io/air-quality-monitoring-solution/cloud/dashboard
Air Care (My Earth)	Mobile-focused, AQI forecast, health recommendations	Personalized alerts, mobile-first, location tracking	Google Play
AirCasting	Wearable monitors, heatmaps, crowdsourced data	Citizen science, open-source, collaborative data gathering	http://aircasting.org/
Weather Channel Tracker	AQI + weather, forecast data, health alerts	Accessible, health + weather synergy	https://weather.com
Windy.com	Animated pollution and weather maps, forecast	High interactivity, weather integration	https://www.windy.com
CAMS (Copernicus)	Global AQI, pollutant layers, research-grade data	High-resolution, global forecasting, ideal for research	https://atmosphere.copernicus.eu
BreatheLife	Global health campaigns, air	WHO-backed, health-centric	https://breathelife2030.org

	quality guidance	information	
European Environmental Agency	Air quality analysis & forecasts, European AQ standards, Mobile apps	EU-focused, standards-based approach, animations, time-lapse	https://airindex.eea.europa.eu/AQI
SenseBox	DIY air quality sensors, open data sharing	Educational, hands-on learning	https://opensensemap.org
ClairCity	Citizen engagement, pollution reduction actions, EU project	Community-driven, promotes pollution reduction activities	https://claircity.eu
AirVisual Europe (AIRUSE)	European AQ data, interactive visualization	European scope, visually engaging AQ insights	https://airuse.eu
WecompAIR	Citizen-led monitoring, personalized recommendations	Collaborative data sharing, health-focused alerts	https://wecompair-project.eu Dashboard
Citi-Sense	Urban AQ insights, health-oriented design	Tailored for cities, focuses on health impacts	https://citi-sense.eu

WeCount	Local pollution data, crowdsourced contributions	Localized focus, engages community	https://we-count.net
hackAIR	Community sensors, pollution trends	Easy entry for citizen scientists, visual data	https://platform.hackair.eu
RIVM Samen Meten	Community sensor network, interactive data analysis, open access	Citizen science-driven, open data, customizable analysis	https://analyseren.samenmeten.nl/
RIVM Luchtmeetnet	National network, real-time AQI, pollutant concentration levels	User-friendly, government-backed, detailed pollutant insights	https://aqicn.org/network/luchtmeetnet/
AtmoSud	Regional AQI, real-time and forecast data, pollutant source breakdown	Regional focus, health alerts, pollutant source attribution & Time elapse	https://atmosud.org/
Smart Citizen	Community-based sensors, real-time air quality monitoring, open access data	Empowering citizen science, localized data, easy-to-use interface	https://smartcitizen.me/kits/

After testing and reviewing the aforementioned platforms and tools, we identified several key considerations for designing an effective geospatial dashboard, particularly for air quality monitoring:

1. **Clarity and Simplicity**: Clearly define the dashboard's purpose, emphasize critical metrics, and prioritize readability to enhance user comprehension.
2. **Geospatial Visualizations**: Incorporate interactive maps, layered data, heatmaps, and markers with pop-ups to provide detailed insights into air quality.
3. **Data Visualization**: Offer a combination of real-time and historical data to facilitate informed decision-making.
4. **User Interactivity**: Enable customizable filters, comparison functionality, mobile responsiveness, and user-specific settings to enhance user engagement and personalization.
5. **Color and Design Choices**: Implement intuitive color coding that aligns with air quality standards, while ensuring accessibility for users with different needs.
6. **User Feedback and Guidance**: Provide clear Air Quality Index (AQI) interpretation, health implications, and actionable insights to help users understand air quality levels and their effects.
7. **Performance and Usability**: Ensure the dashboard supports real-time updates, is scalable for high user loads, employs effective caching strategies, and minimizes loading times for optimal performance.
8. **Integration with External Data Sources**: Incorporate weather data, air quality data from official reference stations external, and satellite imagery to enrich the dashboard's context and data comprehensiveness.
9. **Security and Privacy**: Utilize reliable data sources and uphold user privacy, particularly regarding location tracking and data usage transparency.
10. **Cross-Platform Compatibility**: Ensure the dashboard is compatible across various devices and provides offline access where feasible.
11. **User Testing and Feedback**: Conduct user testing to accommodate diverse needs and implement iterative improvements based on real-world usage feedback.

7.1.2 Scope and Functionalities

To effectively meet user needs, CitiObs uses a user-driven approach in designing and refining its dashboard screens. This process includes creating paper sketches, visual mock-ups, and iterative screen revisions to achieve the desired outcomes. By employing User-Centered Design principles, CitiObs balances the technical challenges of System-Centered Design with the social aspects involving users, creating systems that not only support but also motivate users to engage and learn more effectively. This approach improves productivity, enhances work quality, reduces support and training costs, and boosts user satisfaction.

CitiObs places strong emphasis on its unique capabilities and added value. A key feature is its interoperability with STA and STAplus, enabling seamless data exchange and integrating data from multiple sources. The CitiObs DSS will support interoperability by using VirtualAir as a BackEnd instead of using its own one.

Specific functionalities of the DSS will include:

- **Web-Based Dashboard**: Interactive dashboard accessible via the web, featuring real-time air quality data and user-friendly visualizations.
- **Open Data and Data Governance**: Visualize sensor data collected by different communities, always showcasing the origin and license of the data.
- **User-Centric Data Management**: Allow users to interact with the data in different formats, e.g. time series, environmental maps, access to raw data, quality control data and calibrated data, and data aggregated as AQIs.
- **Integration with External Tools**: Support interoperability with other existing CitiObs tools, as the tools for communities, to create a comprehensive suite of functionalities for users.

A more detailed overview of the DSS architecture is shown in Figure 11. The final functionalities to be included in the DSS will be co-designed together with the stakeholders from the participating Citizen Observatories, as well as other interested stakeholders at national or European level.

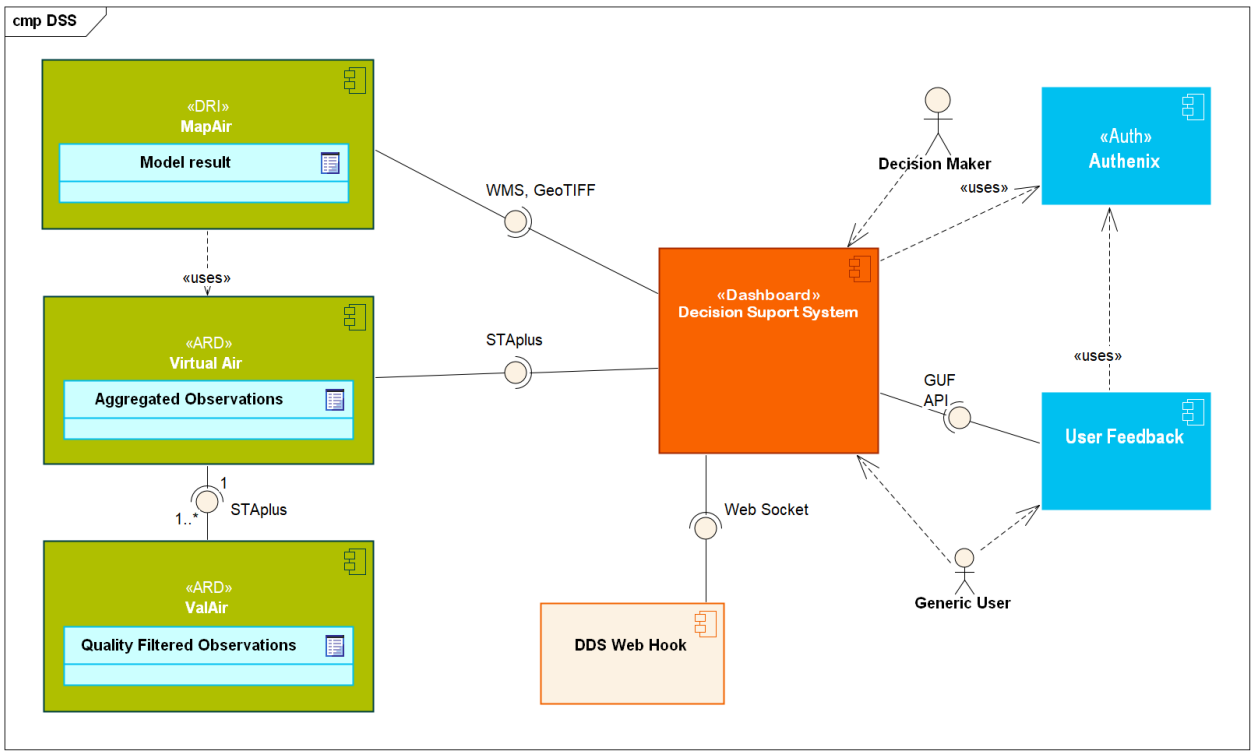

Figure 11: CitiObs DSS Architecture (integrated with the CitiObs general BackEnd)

7.1.3 Early Designs & Requirements

Through collaborative sessions and feedback loops, we have iterated design elements to align with CitiObs's intended user flows and functionalities. Special attention was given to key aspects such as user interface intuitiveness, data visualization, and system scalability to support future expansions. Table 2 presents a list of key requirements that emerged from these iterations, capturing essential technical specifications, and user goals.

Table 2: Core User Requirements for DSS

Req#	Description
R1	User will **land** to **single static dashboard** with a map overview.
R2	User can **select** only one/a **single parameter** from a **drop-down menu**
R3	User can **select** a **specific city** from a **drop-down menu**
R4	User can **select** the preferred **language** of the **application** to appear from a **drop-down menu**

R5	User can **zoom in/out** from the map
R6	**IoT sensors** will be **represented** by a **smart pin** and **colored** in **gradients** based on the last aggregated observed value
R7	User can see **diagrams** for a **specific smart pin** in the pop-up window by **clicking the view diagrams button**
R8	User can **select** a **specific period of times** with start and end date from a **calendar**
R9	User can **download** the personalized information in data from or charts form
R10	User can **select** a **quality parameter** by **clicking** a **button in the quality dialog box**
R11	User can **log into their account** by **clicking** the **log in/sign up button** in the tool's header
R12	User can **create an account**, if he/she doesn't have one, by **clicking** the **text button sign up here**
R13	User can **retrieve** its own **password**, if he had lost it, by **clicking** the **text button forgot password**
R14	User gets a **right sided menu that** can click to **expand**
R15	User can **get notifications** for a **satisfied condition** by clicking the notification from the menu
R16	User can **create rules** for a **specific parameter** and **define a condition**, by **clicking new rule button**.
R17	User will **get a pop-up window** with a set of **warnings** when a **condition** is **satisfied**

Figure 12, Figure 13 Figure 14 and Figure 15 further illustrate design mockups that highlight the primary interfaces and functionalities, providing a visual representation of how users will interact with CitiObs. Figure 12 represents the location of the sensors.

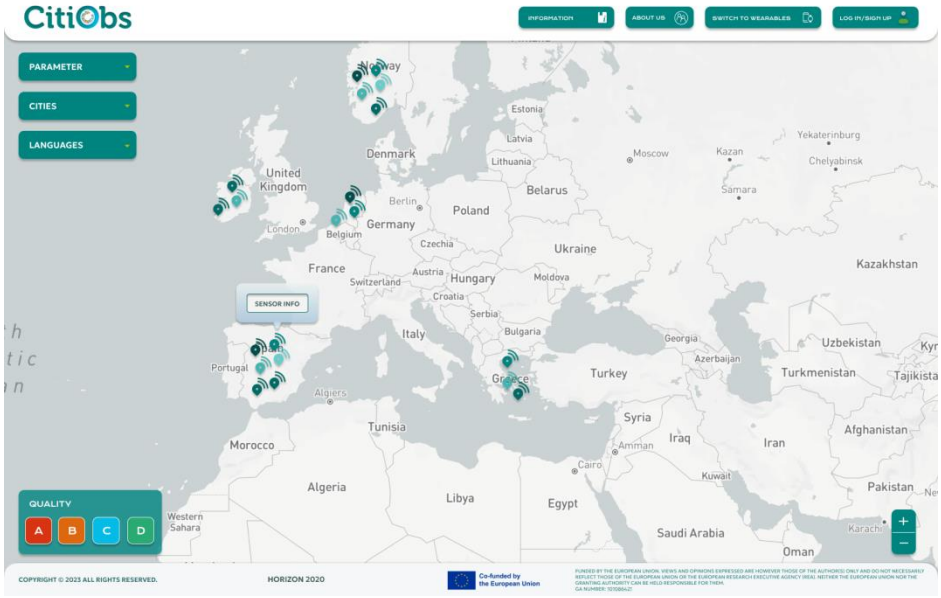

Figure 12: DSS Mockup – Sensors

Figure 13 represents the information associated to a particular sensor when it is queried by location (left click).

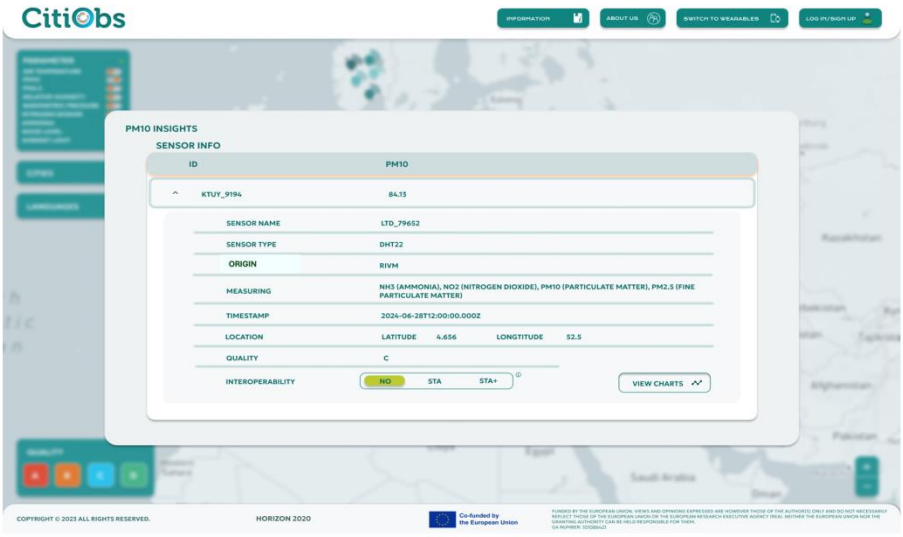

Figure 13: DSS Mockup – Sensor Info

Figure 14 represents the possibility to show the temporal evolution of a pollutant.

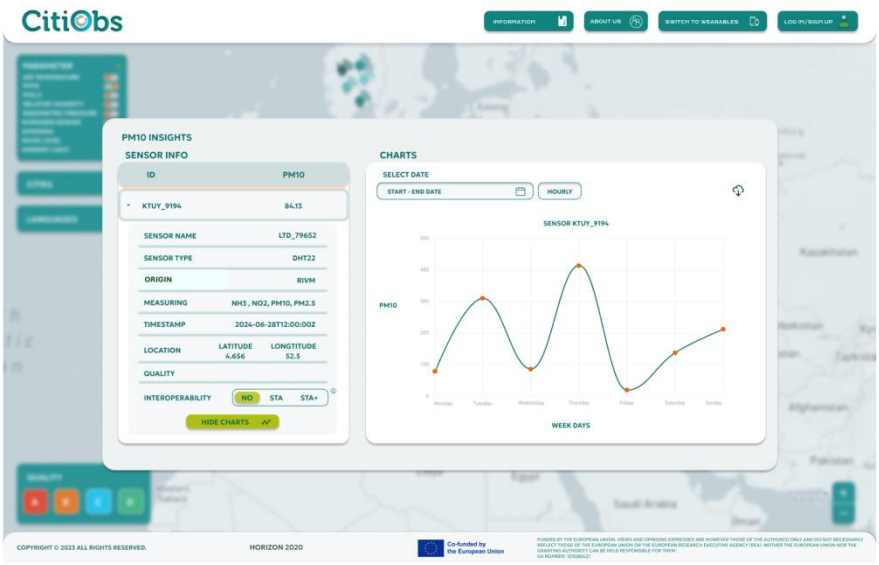

Figure 14: DSS Mockup -- Sensor Data Chart

Figure 15 shows the capacity to select a temporal interval.

Figure 15: DSS Mockup -- Sensor Data Search

These early designs serve as a step to validate assumptions, address potential usability issues, and ensure that development aligns closely with user expectations.

7.2 TAPIS – Tables from OGC APIs for Sensors

One of the outstanding features of the SensorThings API is its capacity to respond to complex data queries based on the ODATA protocol. Example queries can be found in the SensorThings API and STAplus standards as well as in tutorials such as the one developed by SensorUp (https://developers.sensorup.com/docs). Even with these resources and tutorials, creating complex queries can be challenging as ODATA is very flexible querying language. In CitiObs, we are developing a Browser-based Web Application named TAPIS to illustrate the generation of ODATA queries leveraging visual representation. TAPIS originally was inspired by the Orange Data Mining (https://orangedatamining.com/) software visual interface, but it works entirely as a Web-App. With TAPIS, users can create complex queries by combining visual elements. Once the user has achieved the desired result represented graphically, TAPIS can be used to illustrate the corresponding ODATA query that produces the result.

Furthermore, TAPIS supports authentication which enables users to import CSV based observation data into any STAplus compliant Data Service.

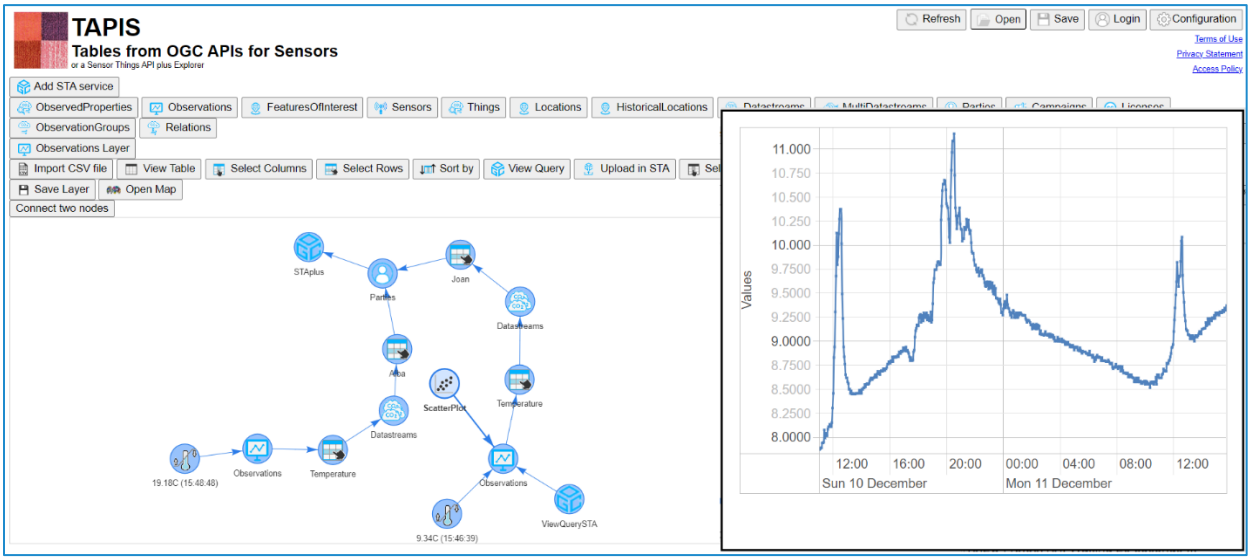

Figure 16: Tables from OGC APIs for Sensors

The Figure 16 illustrates and example where a query results in a table (e.g. a representation of the original JSON array of results in the response).

7.3 Python libraries for local communities

A series of analytical software tools for local communities interested in exploring, visualizing, processing and analyzing data from Citizen Science and Citizen Observatories sensors were designed. These tools will help citizens to access and combine data from various sensors and different observation parameters into a unified workspace, allowing them to analyze the data to identify relationships and trends. The tools will support communities in preparing various scenarios where sensor data is utilized to achieve specific objectives, such as communicating the impact of air pollution in a neighborhood, assessing and comparing conditions over time, and supporting education or informed decisions.

The analytical tools include a library of Python scripts (run locally on a user's computer) and Jupyter notebooks (web-based), serving as 'data recipes' for citizens to follow interactively. They will provide access to SensorThings API and process data for specific scenarios. Additionally, the library is available from the visual environment for accessing and analyzing sensor data through an Orange Data Mining desktop application (https://orangedatamining.com). Orange Data Mining is an open-source software desktop solution for data analysis, visualization, and machine learning. It offers a unique visual programming approach that makes data mining and analysis accessible to both novice users and experienced data scientists. Orange utilizes a canvas-based interface where the developed python library can become a widget to get data and perform data analysis workflows by connecting functional components (see Figure 17). This allows for intuitive construction of complex data pipelines without writing code. Orange provides a wide range of interactive visualization tools including scatter plots, bar charts, histograms, and heatmaps. These help users gain insights and identify patterns in their data.

Comprehensive documentation, including step-by-step guides and instructions, will ensure that even novice users, such as communities with little or no experience in data analytics can effectively work with the sensor data and derive valuable insights.

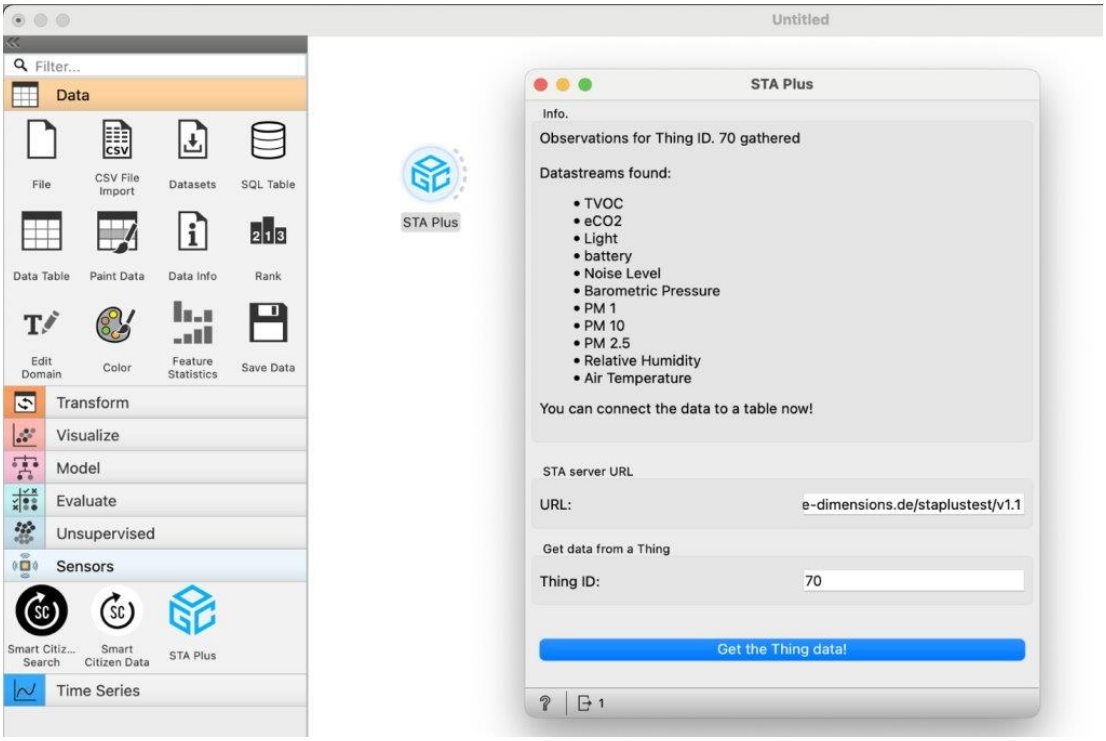

Figure 17: Python library integrated as Orange widget.

7.4 NiMMbus: Geospatial User Feedback

NiMMbus (https://www.nimmbus.cat/) is a solution for storing geospatial resources on the MiraMon cloud. The system implements the OGC Geospatial User Feedback (GUF) standard (http://www.opengeospatial.org/standards/guf). It enables the provision of comments, ratings, questions, etc. that can be associated with geospatial assets on dashboard using data identifiers. NiMMbus supports GUF resources, it allows for creating a citation of an external resource (pointing to an external catalogue or repository) and associate feedback items to it. NiMMbus follows the OGC GUF standard, it facilitated the integration in other systems such as the DSS, providing a feedback mechanism for this system.

An example of integration of the TAPIS interface with NiMMbus can be seen in Figure 18 and Figure 19.

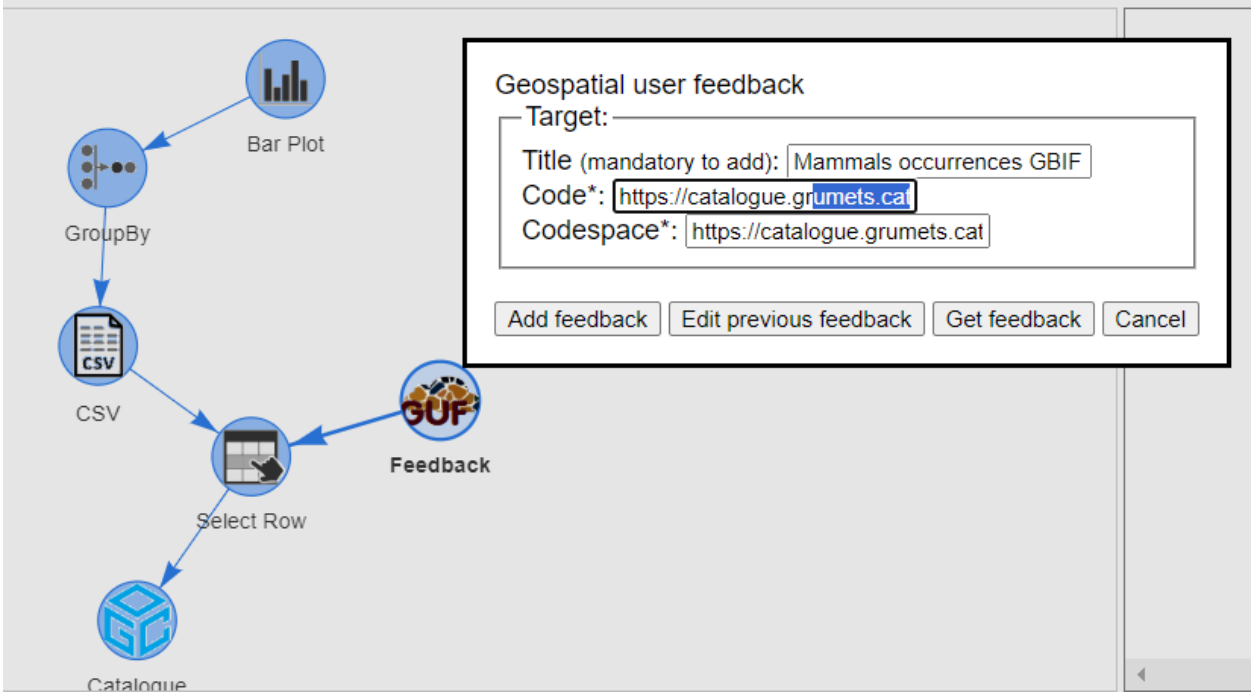

Figure 18: TAPIS integrates NiMMbus as a feedback widget.

The interface of the NiMMbus system is called. After logging in the system is possible to report our "findings" as a "comment".

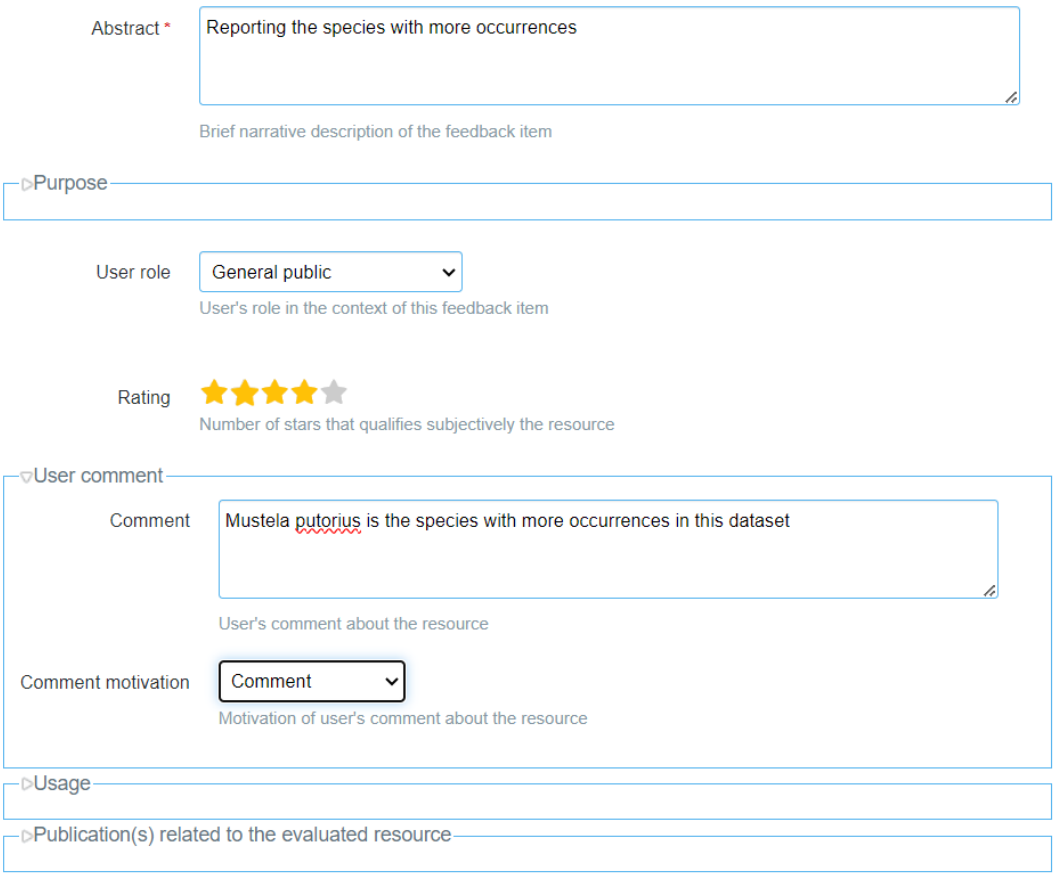

Figure 19: NiMMbus main interface to create a feedback item (a comment in this case)

SECTION 8: Connection to other initiatives

The following section describes how the project contributes to the Copernicus services, GEOSS and the Green Deal Data Space (see Figure 20).

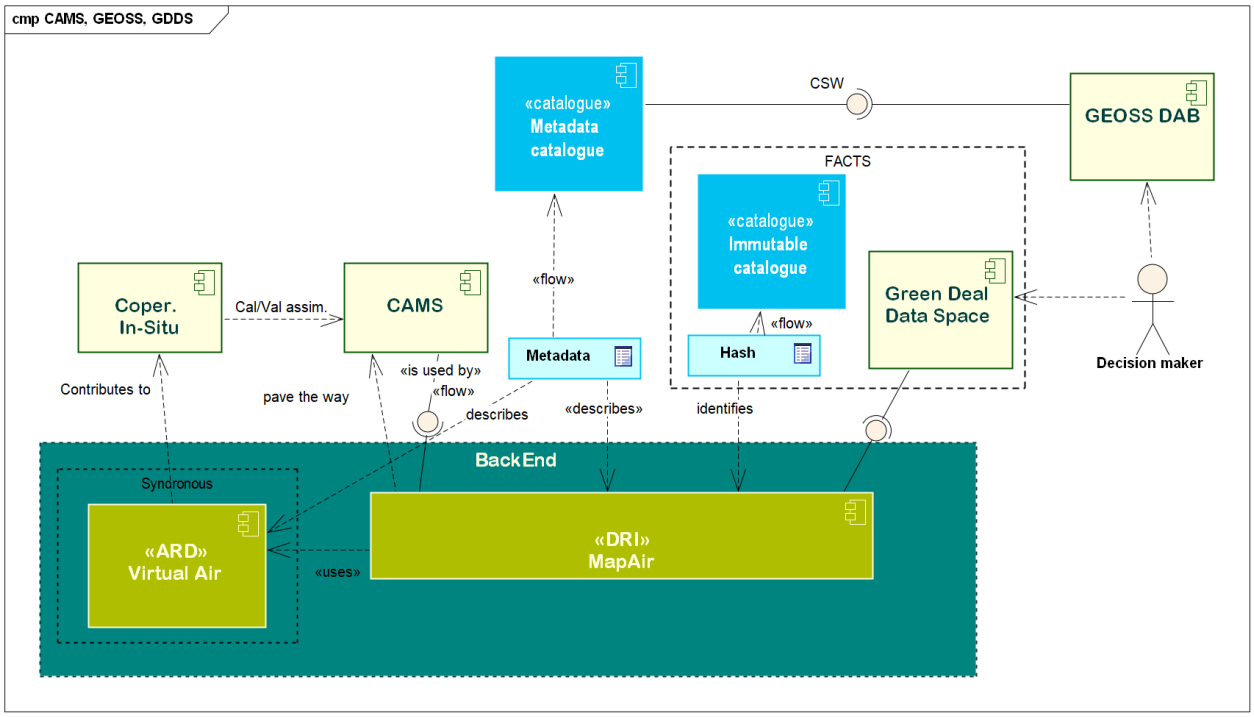

Figure 20: UML diagram on how the project top level of the back end connects to Copernicus, GEOSS and the GDDS

8.1 GEOSS

The interconnection to GEOSS implies to be represented in the GEOSS Yellow Pages as a data provider. In CitiObs, observations are offered in an integrated manner from Virtual Air via OGC STAplus standard. Currently, GEOSS is not supporting STA or STAplus APIs but it will do that in the near future. Taking that into consideration, the discovery of STA and STAplus instances can be done via metadata records registered in a catalogue. The solution is twofold:

- CitiObs will set up a GEOSS compliant metadata catalogue (a geonetwork instance) that exposes some records with distribution methods consisting on the relevant STA requests to the right data streams. To make the data access easier for GEOSS users, CitiObs observations may need to be made available as OGC API Features or WFS.

- Internal conversations with the Discovery and Access Broker revealed that they are preparing a connector to the STA or STAplus that may simplify the discovery to a comprehensive set of data streams of observations.

During the next upcoming months, we will collaborate with the GEO secretariat to request the inclusion of the CitObs GeoNetwork catalogue in the GEOSS yellow pages.

8.2 Green Deal Data Space

The integration in the Green Deal Data Space is not clearly defined yet. The Data Space Support Center considers that a Data Space Connector will be necessary to ensure trust. This technological solution introduces a control plane that connects providers while ensuring digital contracts. This forces participants into setting up a technology in their infrastructure to participate in the data space and exchange trusted data. However, this does not solve the main problem in data spaces: Trust in the data without data integration. Alternatively, an approach based on integrity and immutable provenance of datasets is proposed in CitiObs. Each trustable dataset is identifiable by a unique hash that is used as a universal identifier. This dataset is registered in an immutable catalogue together with the hash and the distributed identifier of the data provider. This process stores provenance information and ensures that producers remain in control of their data. Processing in the data space requires a certificate that is issued by the provider. This certificate may contain license conditions and a list of allowed operations that can be executed on the data.

8.3 Copernicus In-Situ Component

The Copernicus In-Situ Component is a crucial part of the European Union's Earth Observation and monitoring programme, Copernicus. This component complements space-based observations and services by providing essential ground-based, sea-borne, and air-borne data to enhance the accuracy and reliability of Copernicus products and services. The Copernicus In-Situ Component is increasingly recognizing the potential of citizen science and crowdsourcing as valuable sources of ground-based data to complement satellite observations. The Copernicus programme is exploring ways to incorporate citizen-generated data into its in-situ component. This approach aims to fill data gaps, enhance the granularity of observations, and engage the public in Earth observation efforts. CitiObs will dialog with Copernicus to find the best way to include VirtualAir virtual dataset in their catalogue of in-situ resources. The

catalogue is a comprehensive database called Copernicus In-Situ Component Information System (CIS[2]).

8.4 Copernicus Atmosphere Monitoring Service

The Copernicus Atmosphere Monitoring Service (CAMS) provides consistent and quality-controlled information related to air pollution and health, solar energy, greenhouse gases and climate forcing, everywhere in the world.

CAMS is one of six services that form Copernicus, the European Union's Earth observation programme which looks at our planet and its environment for the ultimate benefit of all European citizens. Copernicus offers information services based on satellite Earth observation, in-situ data and modelling.

The Copernicus Atmosphere Monitoring Service is an input to the level 4 (MapAir). The observations produced by the COs are used to create derivative product that improves resolution in urban areas. This methodology could pave the way to a future improved CAMS product. If the European Commission and ESA finds MapAir approach interesting and useful, they could include it as a requirement for a future tender that improves current CAMS products.

8.5 EOSC

In the last months EOSC had drastically changed the way data and services are integrated in the EOSC infrastructure. In the past, the integration was done by requesting inclusion in the EOSC marketplace. The EOSC marketplace no longer exists. Now, the integration of services requires setting up a node in the EOSC infrastructure. The first nodes (including the EU node) are starting to emerge, and it is still unclear how the results of an EU project could be included in one of the emerging nodes.

REFERENCES

[1] The GeoJSON Format Specification, January 15, 2015. Available at: https://datatracker.ietf.org/doc/draft-butler-geojson/

[2] OGC 08-094r1, OGC SWE Common Data Model Encoding Standard version 2.0.0. Available at: http://www.opengis.net/doc/IS/SWE/2.0

[3] OGC 09-001, OpenGIS SWE Service Model: Implementation Standard version 2.0. Available at: http://www.opengis.net/doc/IS/SWES/2.0

[4] OGC 10-004r3 and ISO 19156:2011(E), OGC Abstract Specification: Geographic information — Observations and Measurements. Available at: http://www.opengis.net/doc/as/om/2.0

[5] OGC 15-078r6, OGC SensorThings API Part 1: Sensing version 1.0. Available at: https://docs.ogc.org/is/15-078r6/15-078r6.html

[6] OGC 17-079r1, OGC SensorThings API Part 2 – Tasking Core version 1.0. Available at: https://docs.ogc.org/is/17-079r1/17-079r1.html

[7] OGC 18-088, OGC SensorThings API Part 1: Sensing version 1.1. Available at: https://docs.ogc.org/is/18-088/18-088.html

[8] OGC 22-022r1, OGC SensorThings API Extension: STAplus version 1.0. Available at: https://docs.ogc.org/is/22-022r1/22-022r1.html

[9] RFC 6902, JavaScript Object Notation (JSON) Patch. Available at: https://www.ietf.org/rfc/rfc6902.txt

[10] OGC 21-068, OGC Best Practice for using SensorThings API with Citizen Science. Available at https://docs.ogc.org/bp/21-068.pdf

ANNEXES

This annex provides a very high-level overview of the STA and STAplus standards.

A.1 OGC SensorThings API (STA) introduction

The OGC SensorThings API provides an open, geospatial-enabled and unified way to interconnect the Internet of Things devices, data, and applications over the web. It builds on a rich set of proven-working and widely-adopted open standards, such as the web protocols and the OGC Sensor Web Enablement Standards [OGC 08-094r1 and OGC 09-001], including the OGC/ISO Observation and Measurement data model [OGC 10-004r3 and ISO 19156:2011]. The OGC SensorThings API Part I - Sensing [OGC 15-078r6 and OGC 18-088] provides a standard way to manage and retrieve observations and metadata from heterogeneous IoT sensor systems. SensorThings API [OGC 17-079r1] is designed specifically for the resource-constrained IoT devices and the web developer community. As a result, the SensorThings API follows the REST principles and uses efficient JSON encoding. SensorThings API supports the use of the HTTP and MQTT protocols. The API supports powerful data queries based on the flexible OASIS OData protocol and URL conventions.

A.2 OGC SensorThings API plus (STAplus) introduction

The OGC STAplus [OGC 22-022r1] is a backwards-compatible extension to the SensorThings API that was created with requirements from Citizen Science. STAplus enriches the SensorThings data model in a way that allows to model that observations are owned by different users. In addition to the ownership, users may express a license for ensuring proper re-use of their observations. The STAplus extension also supports expressing explicit relations between observations as well as between observations and external resources. Relations can enrich observations to enable future extensions supporting Linked Data, RDF and SPARQL. Observation group(s) allow the grouping of observations that belong together.

In the CitiObs architecture the STAplus standard defines the main data exchange protocol, and it is used for a level of processing to support data collection and querying.

The use of STAplus in the context of Citizen Science from the Cos4Cloud project was published as an OGC Best Practices [OGC 21-068] document.