COMPAIR

DELIVERABLE

D3.3 AR design

Project Acronym:	**COMPAIR**	
Project title:	Community Observation Measurement & Participation in AIR Science	
Grant Agreement No.	101036563	
Website:	www.wecompair.eu	
Version:	1.0	
Date:	30 September 2022	
Responsible Partner:	HHI	
Contributing Partners:	-	
Reviewers:	All pilot partners All technical partners 21C External reviewers	
Dissemination Level:	Public	X
	Confidential, only for members of the consortium (including the Commission Services)	

This project has received financial support from the European Union's Horizon 2020 Programme under grant agreement no. 101036563

Revision History

Version	Date	Author	Organization	Description
0.1	22/06/22	Sylvain Renault, Oliver Schreer	HHI	Initial structure
0.2	02/09/22	Sylvain Renault, Oliver Schreer	HHI	Provision of content
0.3	15/09/22	Sylvain Renault, Oliver Schreer	HHI	1st draft for review
0.4	22/09/22	Vlatko Vilovic	inter 3	Review
0.5	22/09/22	Antonia Shalamanova	SDA	Review
0.6	23/09/22	Gitte Kragh Jiri Bouchal	external expert ISP	Review
0.7	27/09/22	Pavel Kogut Karel Jedlicka	21c external expert	Review
0.8	29/09/22	Oliver Schreer Sylvain Renault	HHI	Final revision of review comments
1.0	29/09/22	Oliver Schreer	HHI	Final version

Table of Contents

List of Tables

List of Figures

List of Abbreviations

Abbreviation	Meaning
API	Application Programming Interface
AR	Augmented Reality
BC	Black Carbon
BSON	binary JSON
CS	Citizen Science
DEVA	Dynamic Exposure Visualisation App
GA	Grant Agreement
GNNS	Global Navigation Satellite System
GPS	Global Positioning System, US owned satellite-based radio-navigation
GPU	Graphics Processing Unit
GUI	Graphical User Interface
JSON	JavaScript Object Notation
NO2	Nitrogen Dioxide
OGC	Open Geospatial Consortium
PM	Particulate Matter
QR-Code	Quick Response Code
SDK	Software Development Kit
UTC	Coordinated Universal Time
VR	Virtual Reality
WAQI	World Air Quality Index Project
XR	eXtended Reality

Executive Summary

The aim of COMPAIR is to allow citizens to monitor the environmental situation in their community and to understand their own implications and trigger behavioural changes. Hence, the project will exploit the latest advances in interactive technologies to support this aim with the use of Augmented Reality (AR). Although AR is heavily used in the industrial and medical domain, there are only a few widespread consumer applications, such as the well-known game PokemonGo. However, this technology is currently not very well exploited in citizen science and even less in combination with environmental challenges.

Hence, the development of an AR app faces several challenges. The app should attract citizens to become aware of the environmental conditions, the changes of the environmental situation over time or in different spatial surroundings. The visualisation of environmental data such as air pollution or traffic data in AR is a completely new domain. Just two example applications have been published in the last two years with very limited and simple visualisation capabilities. However, the current development frameworks for AR such as Unity3D offer a wide range of possibilities for interaction and visualisation. The challenge is to transfer and adapt various approaches and concepts for this COMPAIR use case. For some visualisation aspects completely new approaches must be developed.

Beside the user interface, several other technical challenges must be solved ranging from real-time server communication for fetching environmental data up to responsive design and hardware limitations and capabilities of mobile AR devices.

This deliverable presents the current state of the AR app design within COMPAIR, the first prototype implementations of functionalities as well as the testing and evaluation strategy. It has to be noted that we followed a user-centric design and development from the beginning. Initial mock-ups and proposals of functionalities and visualisations have been presented to the pilot partners, which ran a number of workshops with end-users during the summer 2022.

A specific section is dedicated to novel visualisation and interaction concepts with a dedicated focus on visualisation of near and far data. During the course of the project, these concepts will be implemented and thanks to the pilot partners present in the project, evaluation with a large number of end users will be performed. Due to this, we will be able to further improve the application, its usability and the user experience towards a helpful tool for awareness creation in the environmental sector.

1 Augmented Reality for Citizen Science

This document collects and categorises ideas, techniques, implementation and design studies related to the development and deployment of the COMPAIR Augmented Reality (AR) app, named AR app. During the requirements phase, the consortium identified that the functionalities of the AR app might go beyond what is possible with AR. Hence, we assigned a more general title to this app, which is now "**D**ynamic **E**xposure **V**isualisation **A**pp - DEVA".

Augmented Reality is one step in the well-known Reality-Virtuality Continuum defined by Paul Milgram in 1994 (Milgram 1994). As depicted in **Figure 1** below, Augmented Reality (AR) consists in augmenting the perception of the real environment with virtual elements by mixing in real-time spatially-registered digital content with the real world. This is achieved by using displays with see-through capability such as smartphones, tablets or glasses.

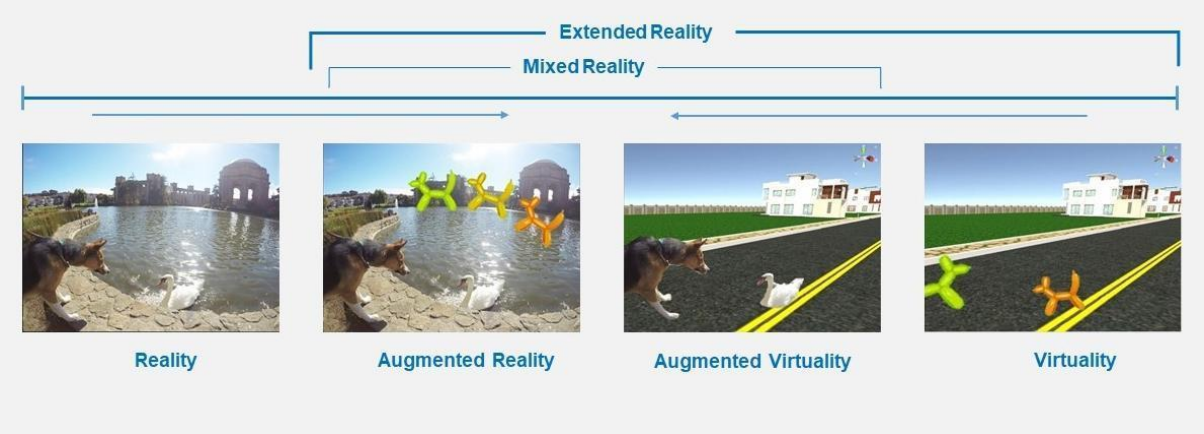

Figure 1: Extended Reality Scheme (courtesy of https://xr4all.eu/xr/)

AR is heavily used in many application domains such as in industry for training, assembly, maintenance, as well as in logistics and health (see ***Figure 2*** and ***Figure 3***). Thanks to the AR-capabilities of modern smartphones, AR technology enters the consumer market. Good examples are the IKEA app (Sokhanych 2022) (***Figure 3*** right) and Pokemon Go (***Figure 4***).

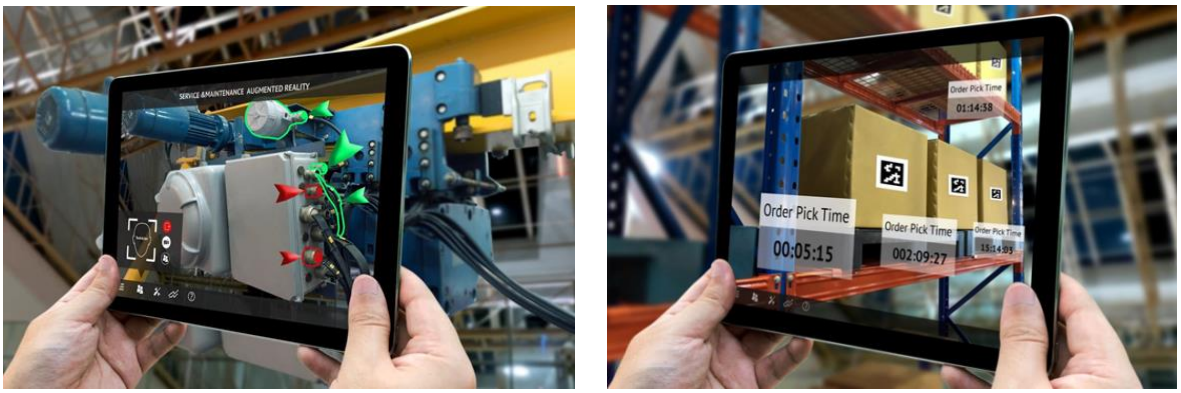

Figure 2: *AR examples for maintenance and logistics*

Figure 3: *AR examples for health and furniture, IKEA app (right)*

Figure 4: *Most popular AR app in the games sector - Pokemon Go*

As described in the project work plan, COMPAIR will benefit from recent advances in AR technology and exploit the capabilities for Citizen Science (CS). With the envisaged AR app, COMPAIR aims to attract and engage people to take part in CS experiments, which are performed in four different pilot countries. The main idea is to enable people to explore their surroundings via their smartphone or tablet camera, so they see a visual overlay of environmental information such as air quality or traffic information. Hence, the AR app will be the link between citizen science environmental sensors provided by the project, public environmental data, CS experiments and the users. In a long-term perspective, the AR app may be used by the wider public to investigate environmental conditions in an easy and appealing way with the aim to change behaviour.

Making complex and various amounts of environmental data available to the wider public is one major goal of the project. Hence, COMPAIR offers various dashboards that allow for inspection of environmental data. The current developments are reported in deliverable D3.4 Dashboard design due by end of September 2022. In such kinds of browser-based

dashboards, geo-spatial environmental data could be represented with heat maps overlaid on street maps either in 2D or 3D (Nurgazi 2019) (see ***Figure 5***).

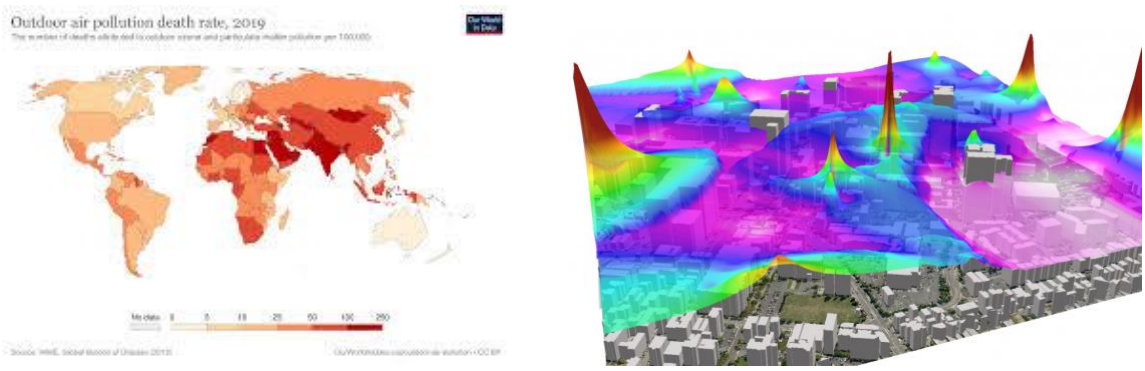

Figure 5: *Heat map visualisation in 2D (left[1]) and 3D (right[2])*

In general, there is no doubt that AR has a great potential to improve education and learning. Several surveys are available, which discuss this (Vargas 2020, Patel 2020, Khanchandani 2021). However, visualising environmental data in an AR application is not very well investigated. There are only a few AR applications published that focus on visualisation of air pollution data for mobile devices, including the two detailed below. To the best of our knowledge, there is no AR app available that provides augmented visualisation of traffic data.

Torres and Campbell presented in 2019 an AR app (Torres and Campbell 2019) visualising main pollutants from the World Air Quality Index Project (WAQI 2019) (see ***Figure 6***, left). Mathews et al. presented AiR, which is available for Android devices (Mathews et al. 2021) (see ***Figure 6***, middle and right). In both applications, pollutants are displayed with flying dots and depending on the severity of the pollution, the number of objects increases. Additional functionalities are provided such as access and visualisation of historical data.

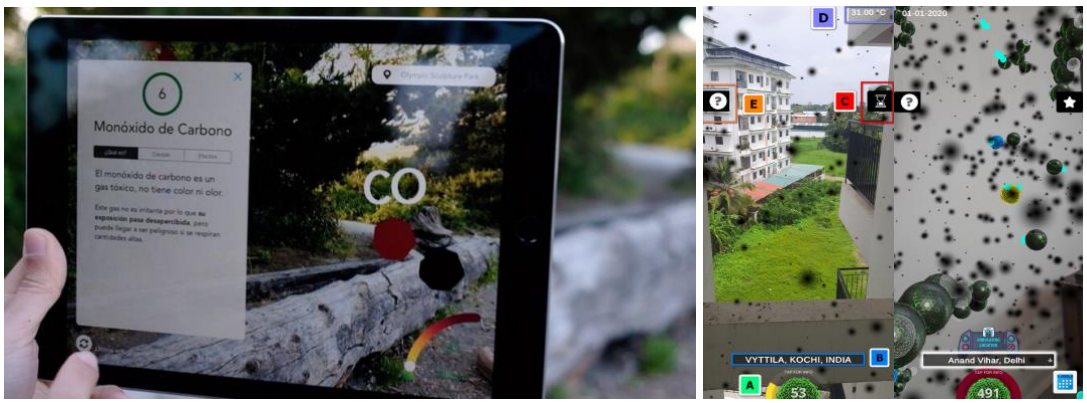

Figure 6: *AIRE - AR App by Torres et al. (left), AiR app by Matthews (middle, right)*

[1] https://ourworldindata.org/grapher/outdoor-pollution-death-rate
[2] https://www.valarm.net/blog/real-time-air-quality-monitoring-in-washington-d-c/

The current State-of-the-Art demonstrates that there is a need to develop an appealing AR application for environmental data and we notice a lot of potential to research, investigate and test novel visualisation concepts.

This deliverable presents the current state of development by M11, but it also aims to provide technical details about AR development as a reference for the community. The document is structured as follows. Sec.2 provides insights about the software architecture, the development environment, device specifications and the overall concept of the AR app. Sec.3 presents general aspects of communication and data formats, but also detailed interface definitions. Various aspects of the user interface are discussed in Sec.4., with a focus on the main user interface and specific aspects of the AR user environment. The current state of the implementation is described in Sec.5. Sec.6 provides an overview on the next steps and the detailed plans for user evaluation in collaboration with the pilot partners in the COMPAIR project.

2 Augmented Reality Framework

The common type of devices for Augmented Reality are specific see-through glasses such as from NReal or Hololens from Microsoft, and camera-based devices such as mobile phones or tablets. See-through glasses are expensive and therefore not well distributed, which is much better for tablets and even more regarding smartphones. Hence, the focus will be on app development for smartphones and tablets as this is what COMPAIR aims to develop..

All AR devices consist of a set of sensors to allow positional tracking of the device in order to place objects on the screen augmenting the real world. GPS[3] tracking is required to locate the device in the 3D world, while gyroscopic sensors detect the orientation in 3D space.

There are two major operating systems in the mobile world, iOS and Android. Therefore, related software development environments need to be considered that allow for accessing the sensors of the device and rendering of various kinds of graphical objects on the screen.

2.1 Development Environment

Google launched ARCore in August 2017 as an app development platform for AR on Android devices (SDK). Many different Application Programming Interfaces (API) allow the device to sense the environment such as environmental understanding, motion sensing, and estimation of light in the environment. Environmental or scene understanding tracks the size and shapes of surfaces and enables the application to detect common objects like tables and cars or flat surfaces like the ground. This is relevant for correct positioning of 3D objects.

For AR development on devices based on iOS such as iPhone and iPad, Apple provides the ARKit SDK. It allows third-party developers to access the sensors and use the information for compelling application development. Due to performance requirements, the ARKit

[3] GPS is the US-owned Global Navigation Satellite Systems (GNSS), other regions or countries rely on their own solutions on top such as Galileo from EU or BeiDou from China

functionality is only available for devices with Apple's A9 processor and higher. The provided functionalities concerning tracking, scene understanding as well as light estimation are available as well.

Another important element is the development framework, that allows for 1) easy and intuitive application development and 2) real-time rendering of content within the application. This kind of framework is called a real-time render engine or game engine. There are two major game engines that support cross-platform development capability. This is important due to the different operating systems on the current mobile device market.

The first engine is Unity (Unity3D 2022) launched in June 2005, which allows the creation of 3D as well as 2D interactive applications. Although it was originally developed for the games market, it has been adopted to several other domains such as automotive, construction and engineering. Another engine is called Unreal Engine, which is a 3D game engine presented to the public in 1998. The engine supports as well a wide range of desktop, mobile VR and AR platforms.

In COMPAIR, we are using the Unity engine for application development (Editor version: Unity 2021 LTS). Unity has a very large and active community therefore a lot of tutorials, demos and learning platforms can be found on the web. Unity uses a simple and easy to learn C# scripting and component based structure for editor and runtime functionalities with an excellent integration in Microsoft development tools such as Visual Studio. Extending Unity is easy and it offers an integration manager for packages (a kind of libraries for assets) and a powerful concept for reusing components (prefabs).

In recent years, Unity3D has developed an easy-to-use framework for Augmented Reality, named AR Foundation. It allows developers to work with AR platforms in a multi-platform way within Unity. This package integrates conjoint functionalities of ARCore and ARKit: Developers do not need to know about the base system, but can use common classes and components instead. The practical benefit of the AR Foundation is that it covers missing functions due to the rapid development from Google and Apple and users can develop independently on the operating system and platform..

Here is a list of some Unity features used for the development of 3D/AR applications:

- Asset management
- Scripting in C#
- Importer (Images, 3D meshes, video, audio)
- Extensible concept for prefabs (prefabricated objects), Unity packages and plug-ins integration via C# wrappers (many languages), local and distributed on the web,
- A lot of pre-included packages for the development of apps (+ version control).
- Good written documentation
- Lots of free assets
- Lots of tutorials, direct from Unity or from renowned persons (Youtube blogs, Learning web-pages, Github, etc.)
- Integration of Microsoft Visual Studio, with debug capabilities
- 3D render and GPU Performance analyser

2.2 Device Specifications

DEVA will be developed for Android and Apple iOS mobile devices (smartphones and tablets), leveraging the above mentioned Unity platform, in order to support the maximum possible coverage of devices.

If a tablet would be used on-the-way, a SIM card version is mandatory to have a permanent connection to the COMPAIR servers. In out-door sites with WLAN hotspots or in-door locations, the user can work with WIFI connections.

2.2.1 Requirements for Android Mobile Devices:

We propose to use modern smartphones/tablets from known manufacturers to guarantee a good quality and life-cycle of the devices. The device model should be from the middle and high-range of a product series. So, no special brand is needed, but the table from Google (see link below) lists the supported devices. It has to be noted that some required level of technology will potentially exclude potential participants.

List of devices
It is recommended to check the link below, before using DEVA on an Android device:
https://developers.google.com/ar/devices

Not all Android devices are compatible with AR technologies/apps. Low-cost models are normally not suitable for AR.

Table 1: *Android device specifications*

Operating System (OS)	Android from version 8 ‘oreo’ upwards. A phone/tablet with Android 10 or higher is recommended.
Memory	No special needs for memory consumption.
Display	No special size, resolution or colour schemes needed. We recommend using a phone with a large screen, up to 6”. Such screens usually have high or very high pixel resolution. Therefore, the image will be sharp and clear. For a tablet, a size of 10” and more is suggested.
Camera	No special type of front camera. A good camera with a high resolution is definitely better for the quality of AR. Modern smartphones/tablets usually have good cameras.

2.2.2 Requirements for iOS Mobile Devices:

We propose using newer iOS smartphones to guarantee the best quality for AR functionalities. Note that almost all Apple devices are compatible with AR technologies/apps. The selected model should be from the middle up to high range of current products. AR compatible models have good to very good CPU, GPU and front camera.

Apple specifications:
It is recommended to check the link below, before using DEVA on an Apple device:
https://developer.apple.com/documentation/arkit/verifying_device_support_and_user_permission

Table 2: *iOS device specifications*

Operating System (OS)	iOS up to version 11. We recommend a phone with iOS 11.0 or higher.
Memory	No special needs for memory consumption.
Display	No special size, resolution or colour schemes needed. We recommend using a phone with a large screen, up to 6". Apple devices usually have good screen qualities. For a tablet, a size of 10" and more is suggested.
Camera	OS smartphones/tablets usually have good cameras.

2.3 DEVA Concept

Like any Augmented Reality app, the Dynamic Exposure Visualisation App (DEVA) is based on many programming modules (e.g. Unity C# scripts, components) and libraries (Unity packages, Plug-Ins, SDKs). DEVA uses a temple model integrating various programming categories for the development of an extendible and reusable CS application (see ***Figure 7***).

It consists of five different pillars:

1. **Localization & Geopositioning:** This contains functionalities to locate the device in the 3D world based on sensor information such as GPS and gyroscope
2. **Sensors & communication:** This pillar is dedicated to all the sensors of the device including cameras, GPS and gyro-sensor. It also consists of hardware, protocols and API's that allow for communication of the device with the outside world.
3. **User interface:** This column is related to all capabilities and functionalities that belong to the user. Touch and interaction functionalities as well as design of the interface, organisation of buttons and menus, but also creative aspects such as colour scheme and responsive design play an important role.
4. **Data visualisation:** This pillar is related to all aspects, how external data received via different communication streams are displayed and presented to the user. Depending on the type of data, various forms of visualisation are required to allow quick and easy interpretation of the information by the user.
5. **Evaluation tools:** Finally, the capabilities of the user interface, the device and the user engagement need to be tested and evaluated. Hence, various evaluation tools must be implemented to allow for an effective application development that satisfies the user requirements and provides information about user engagement and behaviour. However, the project strictly follows the GDPR rules and keeps all user data anonymous.

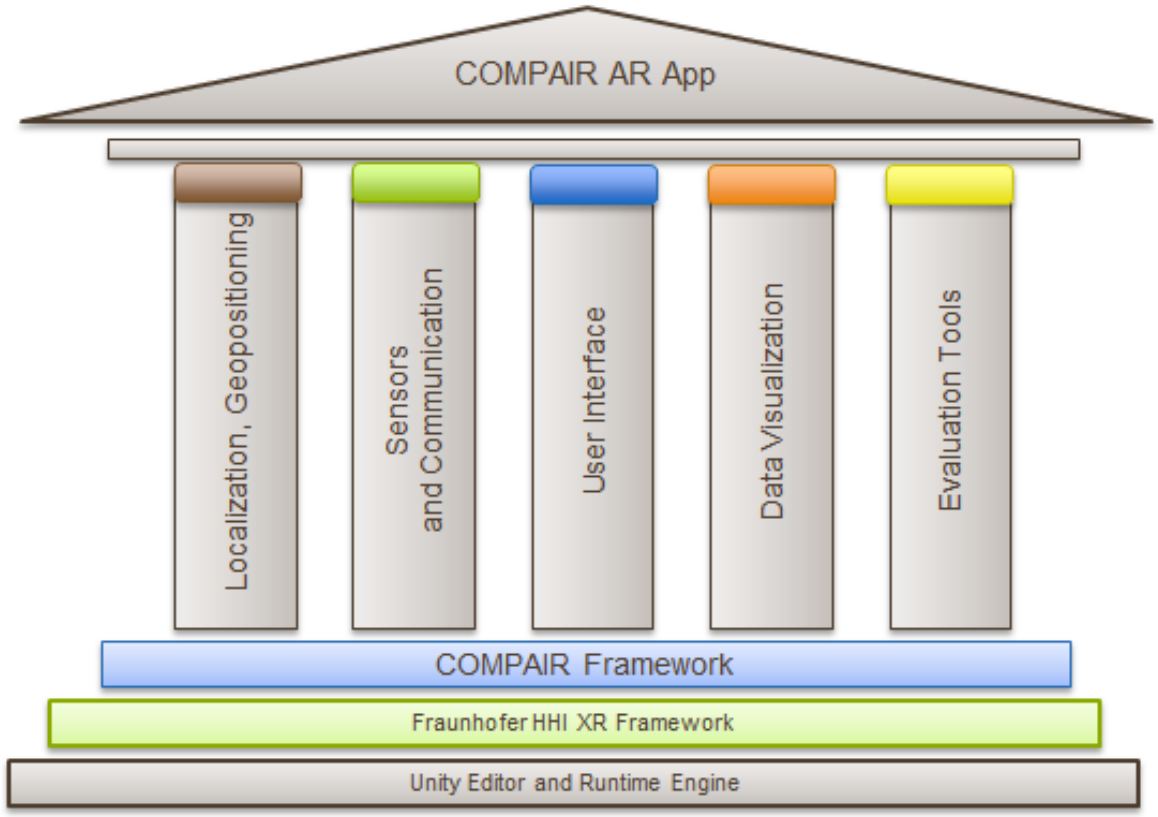

Figure 7: *Temple diagram of DEVA*

Beside those five pillars, the COMPAIR AR App DEVA is based on the Unity Editor and the related runtime engine. Furthermore, the development can benefit from an eXtended Reality (XR) framework that has been developed over the past years at Fraunhofer HHI. Finally, application specific modules and components are developed within the project and allocated in the COMPAIR framework.

DEVA Pipeline

Based on the building blocks described above, we developed a more detailed processing pipeline for DEVA as shown in ***Figure 8***. The colour code of the five different pillars is also applied in this diagram. Beside, there are modules in red colour, which are considered as external modules that are not part of DEVA. The light purple coloured modules assign scripts.

Starting from the top left in the diagram, the COMPAIR sensors provide the necessary input for the visualisation. Besides, it is also planned to access external environmental data to be visualised. The request for all the environmental data is managed by the ***Data Manager***, which performs additional processing and preparation of the raw sensor data for the AR app. All the remaining building blocks enumerated from 1-14 are part of DEVA. On the app itself, the communication with the ***Data Manager*** is performed over the ***Internet*** via ***REST API*** and ***WebSockets*** *(block no.1)*. A description of both protocols is provided in the next section. The received data is transformed into a dynamic data structure in the ***Data Representation*** *(block no.2)* specifically dedicated to the requirements of the Unity framework. If needed, some ***Data Processing*** *(block no.3)* is performed.

The core of the application is the ***Render Process*** *(block no.9)* in the centre of the diagram, which is based on Unity. With a number of ***Interaction Scripts***, the user can interact with the application, request data, change the visualisation mode and much more. A graphical user interface (GUI), the ***Render GUI*** is responsible for the proper rendering of menus, and a separate module takes care of the rendering of 3D objects.This ***Render Process*** requires input from a number of modules that are organised around this main processing module.

Some functionalities will require personalised data and some user specific settings need to be stored. This is managed by the ***Main Process*** *(block no.4)*, which takes care of all user specific data that are stored at the ***Data Manager*** and provided via the communication pipeline. Another important input for the ***Render Process*** is the positional information received by the ***Global Positioning System (GPS)***. For testing purposes, ***QR codes*** can also be used to place specific objects in 3D space, all dedicated to *block no.5*.

All the sensor data coming from the modules in the top right of the diagram are organised and prepared in three different modules, the ***3D Data Representation*** module *(block no.6)*, the ***Visual Parameters and Objects*** module *(block no.7)* as well as the ***Post Process*** module *(block no.8)* and feed finally the ***Render 3D Objects*** module in the ***Render Process***. The 3D data are either ***Meta Data*** such as detailed sensor information, explicit ***Values*** representing different pollutants such as NO2 and finally ***Geometries*** such as 3D meshes representing sensor icons, geometrical figures as spheres, cubes, etc., or abstract shapes such as clouds, particles, etc. The ***Visual Parameters and Objects*** module already contains a number of potential visualisations for our data. We will go into detail in Sec. 4, when we describe the User Interface Design. Any of those visual objects are fed to the ***Post Process*** module, which will have functionalities to modify static and dynamic objects, time- based animations, but also provides Shader Effects for more sophisticated visualisations.

On the left-hand side, the ***User Interface*** module *(block no.10)* provides the necessary graphical elements such as menus, toolbars or windows. This module receives various local ***Assets*** *(*Unity assets, *block no.11)* such as the *GUI*, *3D Meshes*, but also input from gamification. The ***Gamification*** module is based on the interaction with the overall ***Gamification Engine*** on the ***COMPAIR platform*** *(block no.12)*. This ***Gamification Engine*** aims to engage the users in their activities and serves not only the DEVA, but also the dashboards developed in the project. The GUI development is strictly following a user-centred design. Hence, depending on the user experience various levels of details and complexities will be implemented to provide the desired look and best experience. There will also be some ***Evaluation Tools*** *(block no.13)* developed in order to track user behaviour and usage of the app and the different functionalities in an anonymized way. This will help the developers to identify issues in the UI design and the user interest in different functionalities.

Finally, the Render Process updates all the information on the mobile device in the ***Presentation on Device*** module *(block no.14)*. Here, responsive design is mandatory in order to cope with different display sizes, resolutions, but also landscape and portrait format.
In order to support agile development, but also to measure the user experience and usability, ***Evaluation Tools*** will be included in the app.

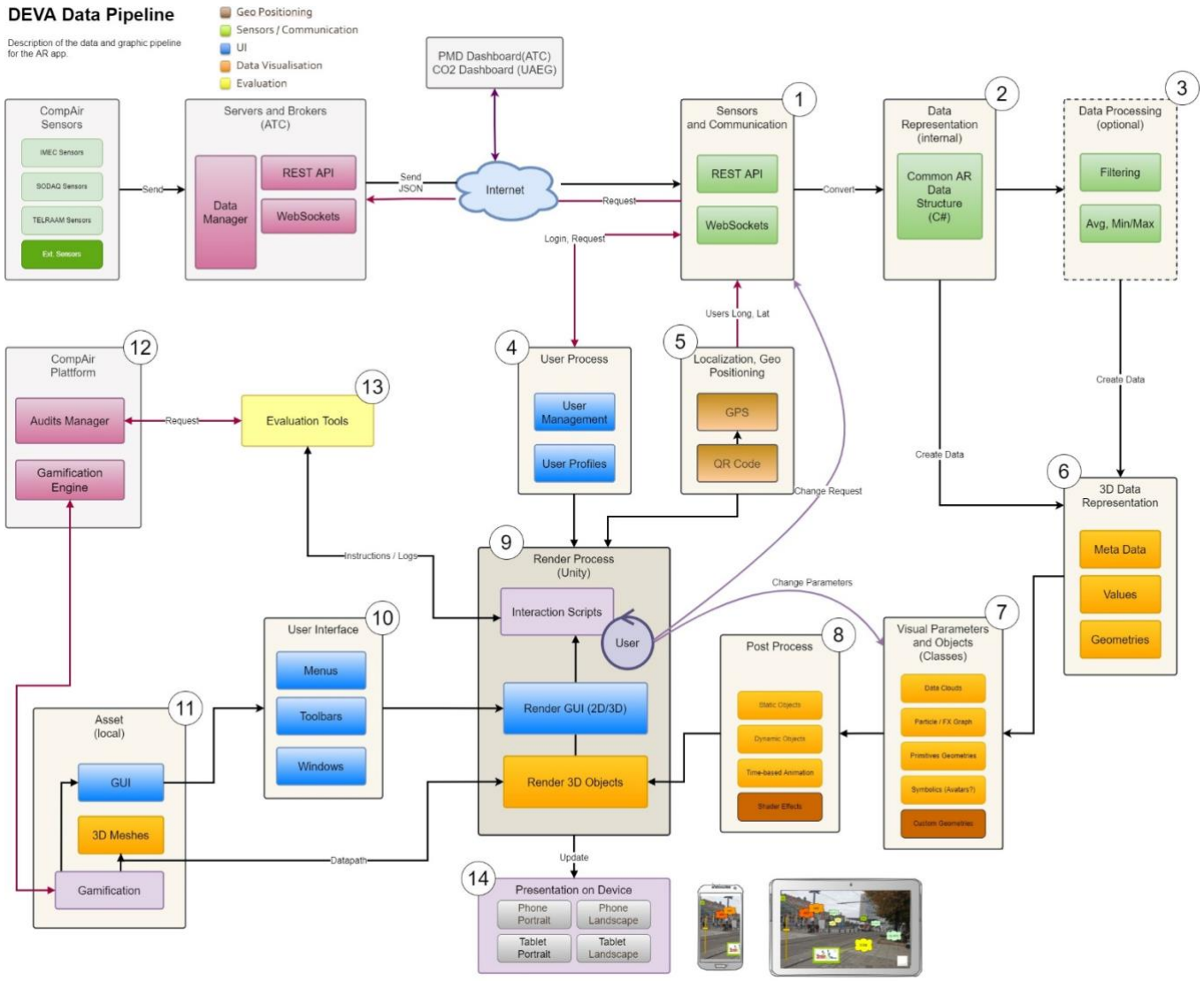

Figure 8: *DEVA pipeline (see text for details)*

3 Communication and Data Formats

In this section, the communication protocols between DEVA and the Data Manager are described as well as the current state of the defined data formats.

JSON

Depending on the amount of data, either JSON (JavaScript Object Notation) or BSON (binary JSON) will be used for storing and transporting data. JSON is mandatory for all small size data structures, while BSON is used for large data sets such as a series of historical data. The data in a JSON string is described in an easy to read and understandable way (see ***Figure 9***). It can be automatically converted or transferred in C# classes and back to a string, by using special modules for serialisation and deserialization. Unity integrates such modules since JSON became very popular in the last few years.

Figure 9: *Example JSON string for a user (left) and a thing (sensor device (right)*

3.1 REST API

One important protocol is the REST API (Representational State Transfer Application Programming Interface), which is a communication protocol used for distributed systems and web services. In COMPAIR, it serves the synchronous data transfer between Data Manager and DEVA. We will handle a large variety of requests for data and information from DEVA to the Data Manager. Error handling is also managed via REST.

OpenAPI

In order to simplify the interface development, we will use the OpenAPI Specification, previously known as the Swagger Specification (Swagger 2022). This framework allows defining interfaces for client/server modules in a unified way (HTTP request end points (paths) and data structures known as components) (see ***Figure 10*** for an example). There exists also an OpenAPI Initiative (OAI 2022) created by a consortium of forward-looking industry experts who recognize the immense value of standardising on how APIs are described.

A unique interface is defined in YAML, which is another mark-up language, similar to XML, dedicated to data serialisation. In this interface definition, actions and schemata are described. The final YAML file can be translated in approx. 30 languages and environments for clients and servers depending on the tools used to auto-generate codes. Two well known tools can

be found: Swagger from Smartbear is the reference one and OpenAPI Generator is a free implementation with an excellent compatibility to the OpenAPI standard. Currently, many other tools were developed and integrated in SDKs. For DEVA, we will use the OpenAPI version 3.0 and the free generator (OpenAPI Generator 2022).

```
paths:
  /srv/ping:
    get:
      summary: Test the CompAIR server availability.
      tags:
        - Server
      operationId: getPing
      responses:
        '200':
          description: Ok, returns a string message.
          content:
            text/plain:
              schema:
                type: string
                example: "pong"
```

Figure 10: *Exemplary end-point for a "ping" request (part of an OpenAPI file)*

3.2 WebSockets

The second protocol used within DEVA is WebSockets. WebSocket is a world-wide used protocol to communicate over the Web, inside web browsers as well as in Web independant applications. It uses the same HTTP/S mechanism like any web browser. The WebSocket protocol enables interaction between a web browser or another client application and a web server. It offers lower overhead compared to other alternatives such as HTTP polling, facilitating real-time data transfer from and to the server. Therefore, it is dedicated for fast synchronous and asynchronous bi-directional communication. Furthermore, one useful benefit of this protocol is that applications do not have to set up special firewall settings to send/receive data.

The WebSockets framework of COMPAIR provides the messaging between Data Manager and DEVA e.g. for warnings or real-time data visualisation. Using different ports, we can partition the access to the server for multiple DEVA instances or separate specific requests inside the same instance to get a better structure and control.

3.3 Data Structures

In this section, we describe the current state of data managed within DEVA, as well as the data types (see ***Figure 11***). It is important to mention that not all the information is mandatory, but can be provided on a voluntary basis. This is especially important for some user specific data. In terms of data structure and naming convention, we follow the Open Geospatial Consortium (OGC) standard as defined in the OGC SensorThings API Part 1 (OGC 2022). In blue, the OGC conform class names and attributes are assigned in ***Figure 11***.
We distinguish the following different types of data structures:

- **User:** a unique user ID, name, age, height, language, role of the user, gender and a list of things, i.e. devices assigned to this user. The personal data are not mandatory and

require a strict consent by the user. Information such as gender or age may be useful for evaluation purposes.

- **Location:** the location of the thing, the orientation and the active perimeter, which describes the active area around the user
- **Thing:** a universally unique identifier (UUID) for the thing, product name and company, use mode (stationary or mobile device), mobility (static, dynamic)
- **Sensor:** Detailed information of the sensor is defined such as the sensor name (e.g. model name), the device ID and a list of measurements, which the sensor is able to do. Therefore, a sensor can measure different kinds of measurement, named properties in OGC. When a system (or a user) will access such a property, it should start an "observation".
- **ObservedProperty:** the specific measurement type of sensor is defined in this structure such as environmental sensor types (humidity, temperature, PM1.0, PM2.5, PM10, NO2, O3, wind speed) and abject types recognized by traffic sensors (pedestrians, bike, car, and truck). The sensor also has a unit symbol and description, such as percentage or concentration. In this structure, some specific value ranges for the sensor and some reference values to notify the user will be considered.
- **Observation (single measurement):** the specific sensor information, its value, GPS position of the measurement, time of measurement (UTC) and the accuracy.
- **ObservationSeries (multiple measurements):** in addition a list of values can be defined in this structure for a given sensor type, the centre position of all sensor values in the case they are stemming from different sensors, the radius of the region covered by the sensors, the time period, the minimum and maximum value as well as the average.

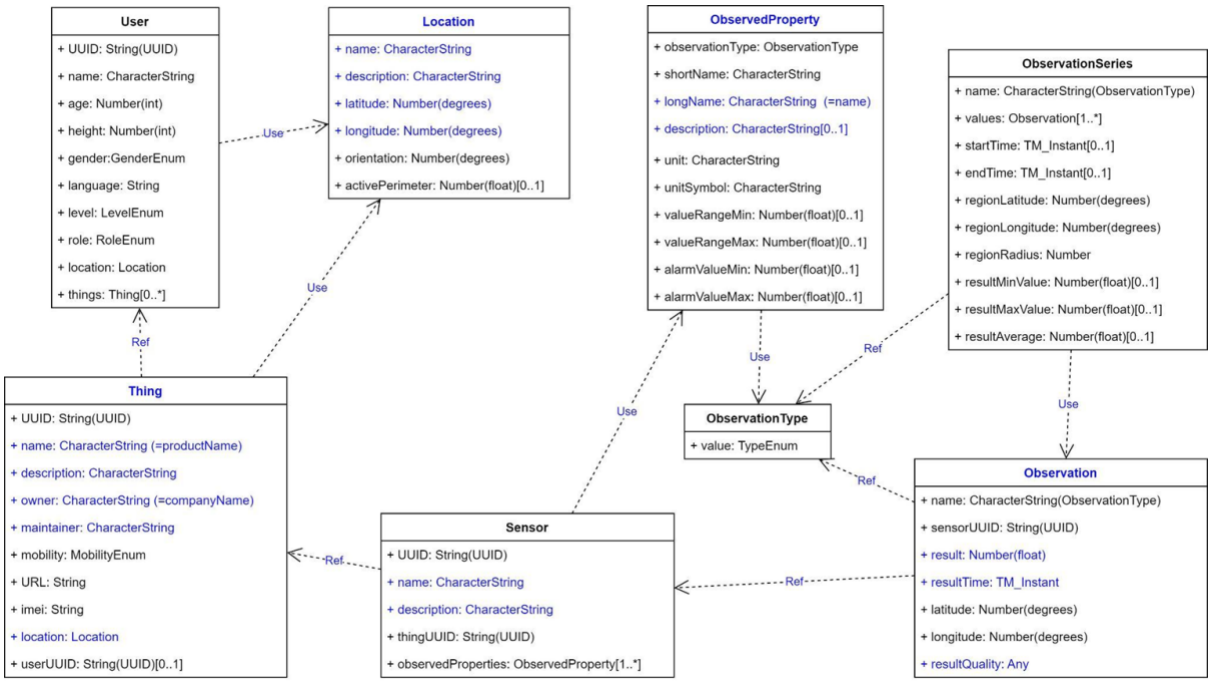

Figure 11: *Class diagram for the data used by DEVA*

4 User Interface Design

In this section, we present the current state of the design of the user interface of DEVA. Several options and possibilities for the main user interface are presented based on the first implementation of DEVA. Concerning the AR user environment, the developed concept for 3D visualisation as well as the interaction concept are presented.
The findings in this section are based on end user involvement right from the start of the project. At a very early stage (6 months after project start), we already provided mock-ups of the potential application. We contributed to the pilot workshops during May/July 2022 and presented the fundamentals of Augmented Reality and the specific ideas, possibilities and limitations around this app and the underlying technology to the participants. During those workshops, two at each of the four pilot cities, we received quite interesting feedback, which was considered in the development.

4.1 Main User Interface

The main user interface (2D interface) is organised and designed as many mobile applications using a similar representation for GUI elements. In the DEVA, Unity is used as the 3D render engine. Therefore, the management of UI assets and placement of those elements on the screen and in formulas and windows differs a little bit from standard GUI development tools. In the COMPAIR Framework a series of components are aiming with the same look-and-feel for the standard UI but with an extension towards 3D animations and effects e.g for gamification. In the DEVA concept, UI elements may respect some design standards such as responsive design, white spaces, organic elements up to gamification.

The main screen of DEVA consists of the following elements (see ***Figure 12***, left):

- **Status bar (first line):** This bar provides status information about COMPAIR (logo), the app and its version, the status of the mobile device such as:
    - Network connectivity (incoming and outgoing activities),
    - GPS connectivity and availability,
    - WLAN connectivity and availability,
    - Sensors connectivity with regards to incoming data from the server,
    - Audio setup, such as volume, sounds for the alarm, etc. and
    - Battery status of the smartphone or tablet.
- **User bar (second line):** this is a user specific line consisting of the following elements (from left to right).
    - **Hamburger menu** (system menu), which offers various system related functionalities, settings, help and login (see ***Figure 12***, right) The items will be visible by clicking the button.
    - **User** name and logo (logged user)

- **Alarm** for sensors by exceeding set ranges for measurement. A number indicating how many alarms were registered sustains the alarm symbol.
- Current **date and time** and the selected time ranges (via an icons)

- **Developer bar (third line, optional):** this bar only appears if the user is logged in as a developer. It offers functionalities for debugging and logging system activities (see ***Figure 13***).
- **See-through window:** this is the space showing the current camera view of the front camera (see below for more explanations).

Note that DEVA has to handle the cut-out of the device's camera correctly. In ***Figure 13***, the camera hole on the left hand side is shown.

Figure 12: *Main DEVA user interface with various interface elements (left) and example of Hamburger menu with the items open (right)*

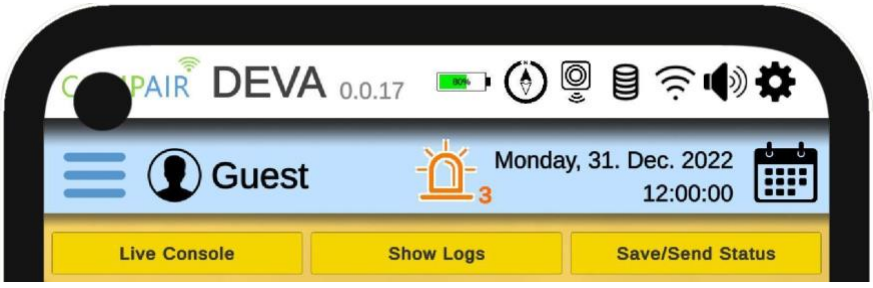

Figure 13: *The top three bars with hamburger menu, logos, battery, various icons, user info, alarms and developer functions.*

- **Augmented graphical elements:** various 2D or 3D elements augment the real camera view. Different ideas and concepts are presented in *Sec.4.3*

- **Toolbar and toolbox:** Additional non-system related functionalities offer settings for various visualisations, data selection and so on. The round tools button (big green button) in the centre is a common symbol for tools. Opening the tools will let the toolbox windows appear so the user can select a tool, function, etc. The toolbar itself presents four buttons for four selectable functions among all existing tools (like a shortcut).
- Other major GUI elements in DEVA are **windows** (or forms) showing help, sensor information, settings or measurement details (e.g. in histograms). ***Figure 14*** presents two examples for a settings window.

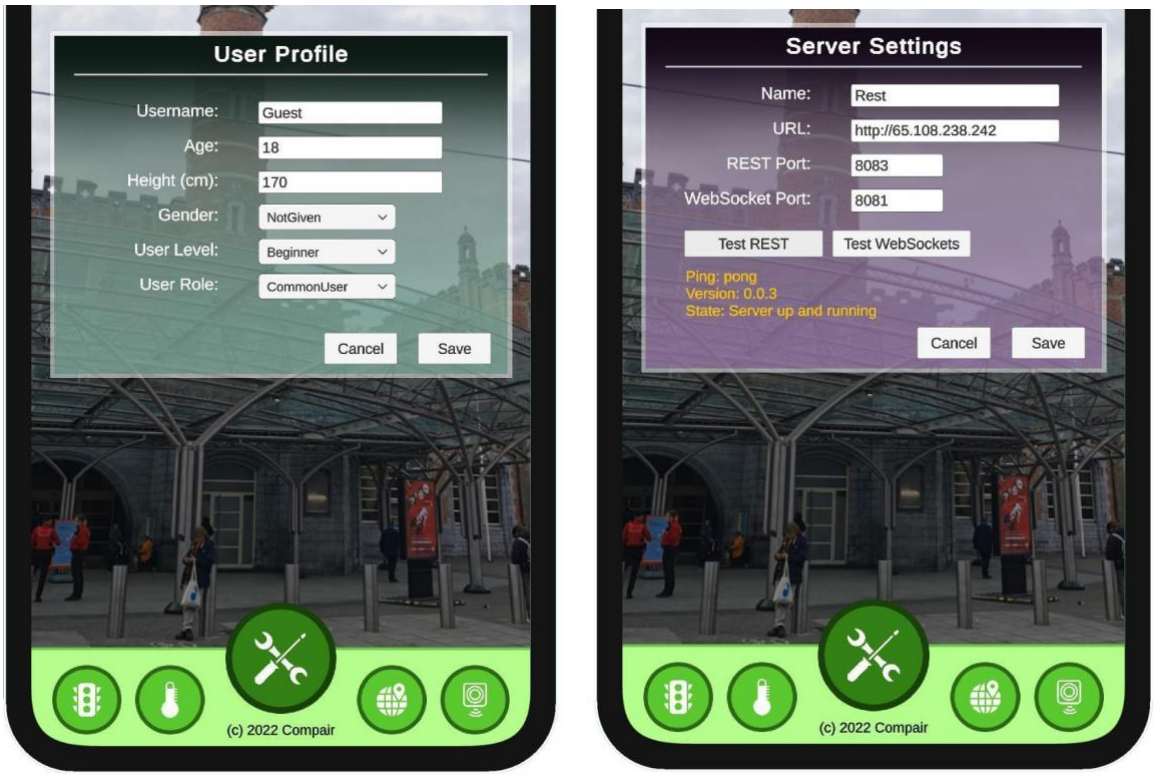

Figure 14: *Two examples for setting windows (left: user 'Guest' settings; right: server settings)*

Windows

Windows are modal elements whereby only one instance can be drawn at a time and the user has to close the window to continue the experience with AR. Windows are placed as needed on top of all other elements of the screen. The COMPAIR Framework proposes a reusable windows system, where the title and the background gradient changes individually.

Icons in the top lines

The icons in the status and user bars as well as the sensor alarm symbol are active icons and allow accessing some base information and settings without the need to open the system menu. The icon colour informs or alerts the user over the state of the function or services. For that, DEVA uses typical traffic light colours (green, yellow or orange and red, see below ***Figure 15***):

- **Green** (or standard colour white) means "everything is fine",
- **Yellow** (or orange) means that there are some (new) warnings that the user has to pay attention to. A window with further details should appear by clicking on the symbol.
- **Red** means that something is not working properly, e.g. a disconnected service.

In some cases, if a service is disconnected, the change to the red colour will happen only after 10 sec. (this time can be adjusted by the app) passing first through the yellow colour. E.g. losing the database signal (the connection to the Data Manager) for a short moment will not produce a red alert immediately. If something very important happens in the app, icons would flash, e.g. a red alert would pass to a red blinking icon after a minute. This time can be adjusted in the app by the user as well.

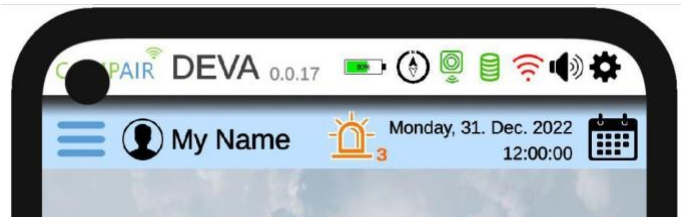

Figure 15: *Colour scheme for services not running (here: WLAN icon in red). The alarm icon in the second line (in orange) allows the user to open a window with more details.*

Toolbox

The tools in the toolbox can be customised in their sorting and visibility. In the first DEVA implementation, only a few tools will be implemented and presented to the user (see ***Figure 16***). Therefore, some positions in the box should stay empty. The more functionalities are integrated in the app during project runtime, the more tools (functions, aggregation of functions, settings, etc.) will be offered to the user.

Figure 16: *The opened toolbox. A second click on the big green button closes the box.*

See-through area

This area represents the main area of the screen, where augmented air pollution and traffic information are visualised. UI elements surrounding this area are positioned in a clear way, respecting the “white space” rules. This is the space between the layouts, lines of paragraphs, between paragraphs, between different UI elements, etc. Further, the DEVA UI interface may not cover more than 30% of the whole screen to leave approximately 70% of the space for AR. This rule does not hold, if a window (usually modal) is appearing on the screen mainly in the centre.

Screenshots

In the following ***Figure 17***, different screens are presented, which demonstrate the possible look and feel of the GUI. On the left, the splash screen after the start of the application is shown. The second image shows the start screen, where some background tests regarding AR device compatibility are performed. The third image shows the login screen with some standard functionalities such as register, login or access via an administrator account. The latter is important for the development phase and will be removed in the final version. The fourth image shows the main screen of DEVA with the current toolbox. This toolbox offers functionalities such as settings or preferences. The fifth and last image presents the UI with a sensor settings menu.

Themes

The COMPAIR framework will be equipped with different mechanisms to customise the look-and-feel of the 2D and 3D user interface. Therefore, it is possible to adjust colours, icons, themes and obviously the language of the interface. For this purpose, the user is able to define primary and secondary colours, values for transparencies and foregrounds as well as text colours. This technology is named Tri-Colors (adjacent colour and triad) or Four-Colors (tetrad) design palette (Meaning of triadic colours 2022). The custom parameters are saved in reusable themes. The system offers three standard themes: dark, light and colourful. For an user-friendly usage of the app in the night, we could imagine implementing an eye-care modus, where the blue component of colours is reduced.

Another important aspect is user friendliness, especially concerning colourblind people. This needs to be carefully investigated while choosing the right colour palette.

To select the right colours according to some base colours, a special web page can be called e.g. Paletton.com. The designer can choose a base colour and select the type of palette: monochromatic, adjacent colours, triad or tetrad. Using such a palette avoids some mistakes in the design of an UI. Possible COMPAIR colour schemes are shown in ***Figure 18***.

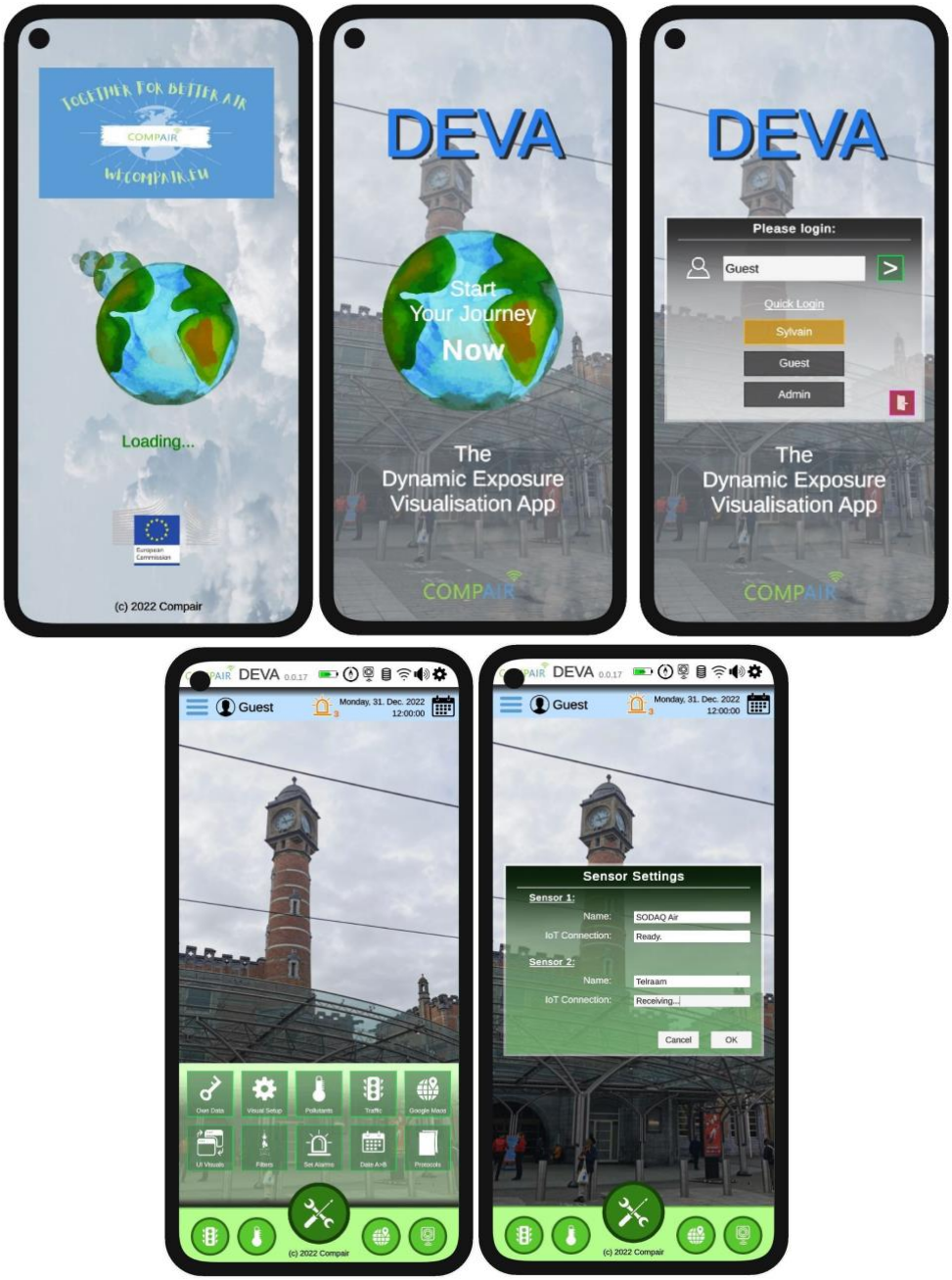

Figure 17: *Examples of different screens and submenus (from left to right and top to bottom): Splash screen, start screen, login screen, main screen with toolbox and window example for the sensor settings menu*

COMPAIR

Figure 18: Example of a 3-Colours selection using the two base colours of the COMPAIR logo: green (hex code: 8bd24c; left) and blue (hex code: 5b9bd5; right), colours proposed by Paletton.com.

In COMPAIR, a series of 3-Colors themes for dark mode, light mode and colourful mode will be defined and saved in the assets of Unity according to the base green/blue colours of the logo. Some extra dimming values for a day/night mode would also be created: the day mode may use the same colours as setted and the night mode may use the dark mode with a blue light reduced colour palette.

The UI system will be implemented in a user-centred fashion. The process of creating the required UI elements requires the involvement of users throughout the design and development process in an iterative way. By creating a dynamic and flexible structure in the definitions of parameters and functions (inheritance, scriptable Unity objects), the COMPAIR Framework and also the DEVA app can easily be extended and optimised for future needs and user desires.

Common design in COMPAIR

In COMPAIR, two other applications are also created: the policy monitoring dashboard (PMD) and the CO2 dashboards (both as web applications). To give all three applications a similar project look-n-feel, some elements (icons, buttons, checkboxes, sliders, forms, etc.) and colours will be shared. So, a user switching to or from DEVA or the other apps may receive a good orientation. For this, the project partners will select a set of most important recurring design components and parameters (see ***Figure 19***, middle circle in green). Furthermore, it is important to plan some user processes (selection, forms…) in the same manner. All this can be adjusted and redesigned during the project runtime as the services and possibilities grow. Thus, the COMPAIR Framework uses special Unity assets (scriptable objects) to maintain dynamic graphical parameters and design themes.

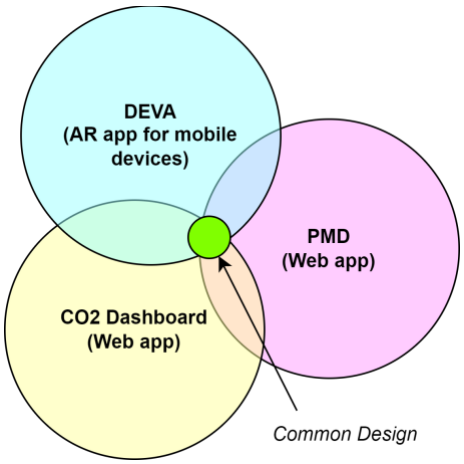

Figure 19: *Common UI design between the different COMPAIR apps*

4.1.1 Responsive Design

An important aspect for a convincing user experience is responsive design. Right from the beginning, some aspects need to be considered that require dynamic programming and adaptive visualisation of graphical elements.

Different mobile devices such as smartphones or tablets have different display resolutions. Therefore, any graphical element must be defined in relative dimensions instead of absolute

ones. By doing so, the relative sizes of graphical elements adapt automatically to different screen resolutions.

The second important aspect is the aspect ratio and format of the screen. Depending on the user preferences, mobile devices are used in portrait or landscape format. This has a direct impact on the arrangement of graphical elements. For example, the elements on the top line must be rearranged, if the user switches from portrait to landscape format (see ***Figure 20***).

Tilting the device (extra function)
If the user tilts its device up to the sky, then DEVA will react to this and show customizable content on the screen e.g. a window with some weather data. The other way round, if the user's device points to the ground, the app will show the GPS and compass windows. Hence, the orientation of the device is used as a shortcut.

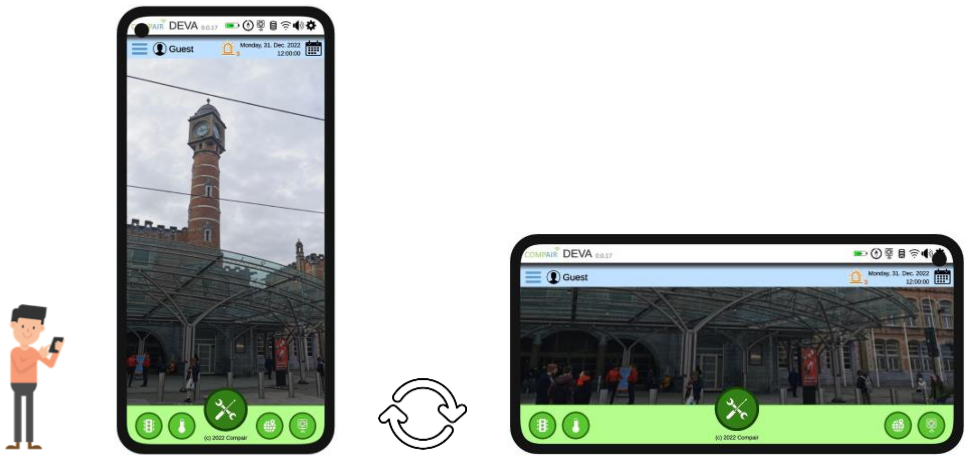

Figure 20: *New arrangement of the GUI, while changing the orientation of the device from portrait (left) to landscape (right)*

4.1.2 Gamification concept

Today, gamification has become more and more significant in the design and conceptualisation of user interfaces. "Gamification is the strategic attempt to enhance systems, services, organisations, and activities by creating similar experiences to those experienced when playing games in order to motivate and engage users" (Hamari 2019). Gamification elements in the UI increase the learning ability and trigger attraction, independently of users' ages and not only for children.

DEVA uses gamification for different purposes:

- To improve user engagement.
- To increase the attention.
- To enhance knowledge retention.
- To amuse children and young people.
- To bring "life" to the app.

The gamification elements should not give the app new functionalities but should extend existing UI processes and make their usages more enjoyable. Therefore, a new series of

icons, colours, animations, transitions and effects would be integrated in very particular, well-considered places, UI processes. As a simple example for an UI gamified element, a text value should be replaced by a banner or by a big speech bubble and text size (see ***Figure 21***).

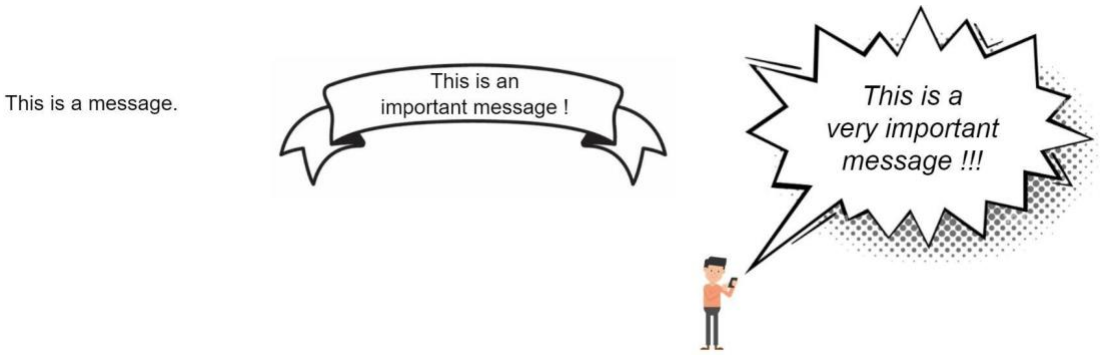

Figure 21: *A simple text should be replaced by a banner (1st step) or a bubble (2nd step), when gamification is used*

Organic design

An early stage of gamification could be a UI design using organic elements. Such elements seem alive because they have their own dynamic and small animation, mostly inconspicuous. In most of the cases, UI elements begin to move slowly when the user moves the mouse cursor over there. When using animated sequences, an organic design attempts to use smooth transitions, with some acceleration and deceleration.

Story telling

In some cases, a gamification module can bring a short “story” to the user, so they have to look (and wait) for a moment, enjoying introductory animations. Activities can be rewarded with points. A jumping colourful button or flying window is entertaining or more amusing than a static grey button or a suddenly appearing window. Such stories are more suitable for youth.

Which user is gamification aimed at?

The UI design of DEVA developed by Fraunhofer HHI includes the definition of user groups (or categories). ***Figure 22*** shows one possibility to group users depending on their interest or ages. This is a draft classification scheme in order to demonstrate the possibilities in terms of user dependent visualisation and the level of gamification. In the first implementation of DEVA, the user management allows the user to enter (1) their level of expertise, age and role (example in ***Figure 22***, left side) and (2) select an UI theme (colour and icon schema) and activate the gamification mode (example in ***Figure 22***, right side).

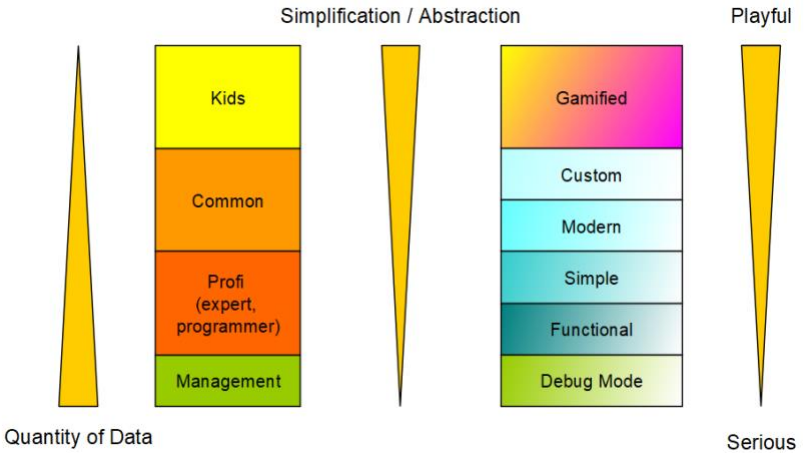

Figure 22: *User specific gamification and visualisation concept*

The difference in the user interface and gamification per user level/category may be defined as follows. It is so far an example and the design will evolve during the project:

- **Professional:**
  - Contrast full UI theme
  - No visual effect
  - A lot of information about the sensors, meta-data, etc.
  - Optional: hidden functions
- **Common:**
  - Default UI theme: Colours schemes (3-Colours) for simple/functional/modern
  - Simple effect (organic, enableable)
  - Common details (no meta-data, technical information)
- **Playful:**
  - Colourful UI theme
  - All visual (and sound) effects on (organic design, gamification)
  - Reduced/aggregated information (no meta-data, not too much technical text)
  - Optional: avatar in the AR scene; optional: in-app game.
- **Administration:**
  - Special admin functions
  - Show log files and debug information (debug mode)
  - Direct access to common functions
  - Simple colour theme: Functional UI
  - No visual effects

Implementation

In the COMPAIR Framework, all UI customisable and gamification elements are “marked” through special scripts. These scripts manage the look and the behaviour of the elements in the GUI and in the 3D augmented reality scene. The internal programming structures (see DEVA pipeline Nr. 6 and 7 (3D) and for 2D elements Nr. 11) are directly controlled and updated from the various user’s settings and the result shall be seen in real-time (2D and 3D render process, Nr. 9).

Note:

- While implementing gamification, it is important to be aware of some limitations and drawbacks. Too much gamification may overload DEVA and it will lower the performance of the application due to complex graphics operations. It will perhaps change the character of the application, which certainly addresses a serious topic, our environmental conditions. Hence, the original objective of the application may move to the background.
- The implementation of gamification elements needs to be continuously evaluated and discussed by all project partners (process Nr. 12 in the DEVA pipeline). Even more, the development of sophisticated gaming functionalities may take a lot of development effort.

4.2 AR User Environment

The elements described in the previous section are standard elements of any mobile application. The challenging part of research and development is the creation of the AR user environment. As presented in Sec.1, there are only two applications published, which are concerned with the visualisation of air pollution in AR. The basic visual concept of those examples is to augment the scene with graphical elements, such as rocks or balls of different amounts and sizes, depending on the level of pollution. It is obvious that there is a need to go far beyond and develop new means of visualisations. A fundamental element for AR is the correct positioning of 3D data in the AR space, which we call geo positioning and discuss in the next section. Different possibilities and ideas for the 3D visualisation of sensor data are presented as well. Finally, we present visualisation concepts of data far away from the user which are not in the direct visual surrounding.

4.2.1 Geo positioning

For any of the visualisation methods, the mobile device must be aware of the position and orientation of the user. Moreover, the viewing frustum of the current camera view must be known in order to locate measurements from sensors at the right place. This can be achieved by exploiting information from different internal sensors.

GPS

The GPS sensor provides the position of the device (GeoPose, see OGC below) with an accuracy of several metres. With this information, the sensor data in the surrounding area of the user can be retrieved from the data server. On top of it, the orientation of the user in space must be known, in order to be able to augment the relevant sensor data in the viewing frustum. The internal compass and/or gyroscope sensor offers the orientation of the device. For testing purposes, an AR/QR code can be used as well to assign specific data to a given location in space.

The measured air pollution data, the traffic data and the devices themselves (which are called “things” in OGC) also have GPS positions. Receiving this information from the COMPAIR Data Manager allows it to calculate distance, perimeters or regions in the same system. However, for the rendering inside the render engine (Unity), the positions are to be transformed to a cartesian coordinate system. In this end system, the 3D meshes representing a measurement

and other UI elements (label, buttons, text blocks with details, etc.) are placed correctly according to the GPS position of the measured data.

In special locations, e.g. bus stations, school buildings, etc. DEVA also proposes a marker-based localisation system. For this, the user has to scan an AR marker (a special picture) to access the data in this place. It does not need a GPS signal because its GPS position is represented by the marker (database for AR markers). To have such a COMPAIR marker onto a wall should have a double impact: increase the attention and incite a user to interact with the app. The marker may also be placed in a frame with further information about DEVA and the COMPAIR project.

OGC

The Open Geospatial Consortium (OGC) addresses the topics of geospatial (location) information and proposes standards to make these services fair for all. Such GeoPose information is anchored to Earth's surface, outside or in buildings. In the data exchange documentation" [Smyth 2022] the consortium explains how GeoPose objects are transferred between services. For the positioning of the objects itself onto the earth surface, two main possibilities exist: a local and a global [Perey et al. 2022]. For the local system, many AR solutions use marker based localisations (called trackables). For the global one, GPS poses are the standard today.

After identifying the localisation of a user or an object, the virtual scene could be extended with some AR content. For this, anchors are placed in the scene, related to trackables or GeoPoses. Anchors should then be extended by the (running) application with content at their final positions. A database concept for managing and transferring trackables and anchors between server and client applications can be found in the group specification ARF-005 of the ETSI ISG standardisation group (ETSI-ISG ARF, 2022).

4.2.2 3D visualisation of sensor data

Current state of the art on AR visualisation of pollution information relies on quite simple approaches, e.g. visualising pollutants through flying balls or rocks. For DEVA, we aim to exploit a wider range of visualisation capabilities provided by current render engines such as Unity. Complex graphical elements such as clouds or specific shaders can be used for this purpose.

In ***Figure 23***, some examples are presented that will be implemented. Depending on the user preferences, different types of visualisation methods can be chosen. Sensor values are of different types (NO2, PM2.5, PM10, etc.) and of different amounts. The user will be able to select directly from a list box the sensor types, the amount (quantity) of data and the distance to the user. DEVA proposes two kinds of data representations: a simple one with the usage of common graphics (primitives) and attributes and a more complex one using effects and particles, investigating more the Graphics Processing Unit (GPU) of the mobile device.

The **simple representation** uses simple 3D models (meshes) to mark the location of the pollutant/measurement in the 3D see-through screen (see ***Figure 23***). Those models can be easily extended as they use static mesh data such as:

- Sphere

- Cube
- Tetrahedral
- Cloud geometry
- Icons/pictograms (representing a cloud, a pin or a device)
- etc.

Figure 23: *Examples of different visualisations of sensor data: (f.l.t.r.) clouds or point clouds, text or values, 3D pins*

These properties can be easily distinguished as follows:

- For each kind of pollutant a specific colour is defined
- For the measured amount of pollution, the size of the graphical element changes respectively.
- To increase the 3D location of elements, the renderer can activate transparencies so that far-away objects disappear in the distance.

The more complex **extended representation** uses the full range of possibilities of the GPU of the device using a shader-based rendering. Therefore, Unity uses three techniques: graphic shaders, visual effect (graph) shaders and compute shaders (Unity Shader Overview 2022). Such effects are used more in games than traditional software and allow creating amazing real time 3D constructs. In combination with the particle system of the engine, also integrated in the shader system, the user gets a good control of the visualisation. All this increases the attractive and fun aspect of the app, especially for kids. DEVA converts the location and the value of measurements in special array lists to implement the extended representation to be rendered as:

- Particle clouds (with or w/o animation)
- Complex shapes (single pollutant values, aggregations, groups…)
- Artistic formation.

The properties depend on the shape and effect complexities:

- Colour and gradient depending on the value range of pollutants. Here some animation for min/max detection can enhance the understanding of data.
- Textures of the shapes to give a more realistic representation of the pollutant (static or animated)
- Parameter influencing the base of the shape (mathematical parameters and formulas)
- Selection of static or animated effects (as clouds, fogs, fire…)

Not all conceivable visual representations are meaningful and can be implemented in the first version of the COMPAIR framework and therefore DEVA (prototype for the pilot). The app should offer only a couple of visual solutions and parameters for both representations in the beginning. Using the flexible pipeline (see DEVA Pipeline diagram, *block no.7 and 8)* the system can be easily extended in the future, on-demand by the project partners and as a result of the various usability and user experience tests together with the pilot partners.

Note:
Figure 23 shows a visualisation using the two representations (**simple** and **extended**) in an abstract form.

Equaliser
To simplify the setup of those parameters, a concept of parameter-equaliser is proposed. Similar to an HiFi equaliser, the user can adjust the "importance" for the different sensor types: A higher value means more data represented by larger geometrical objects; a lower value means less data (or aggregated data) and smaller objects.

In ***Figure 24***, the temperature "importance" value is higher than for RH, PM2, etc. so the system can focus the 3D representation more on temperature observations. A value change can be directly visible in the See-through window bringing the important objects (observations of the corresponding sensor type) in the focus of the user (bigger size, text value…) and making less important objects smaller or transparent.

It is important to note that the visualised data does not necessarily belong to a single observation of an individual sensor. Moreover, averaged sensor data are of interest for a specific user defined spatial region. Other statistical data will be displayed as well such as maximum or minimum values of pollutants in the surroundings.

Beside pollution measurements, additional information is of interest for the user, such as specific information about the sensors themself such as manufacturer (see ***Figure 25***, left). Other means of visualisation can also be included such as small histograms as depicted in ***Figure 25***, right. This information can be placed on the top of the see-through area or three-dimensional where the data are located. Bringing windows closer to the user or on the top permits the user to interact with objects and UI elements directly via the touchscreen.

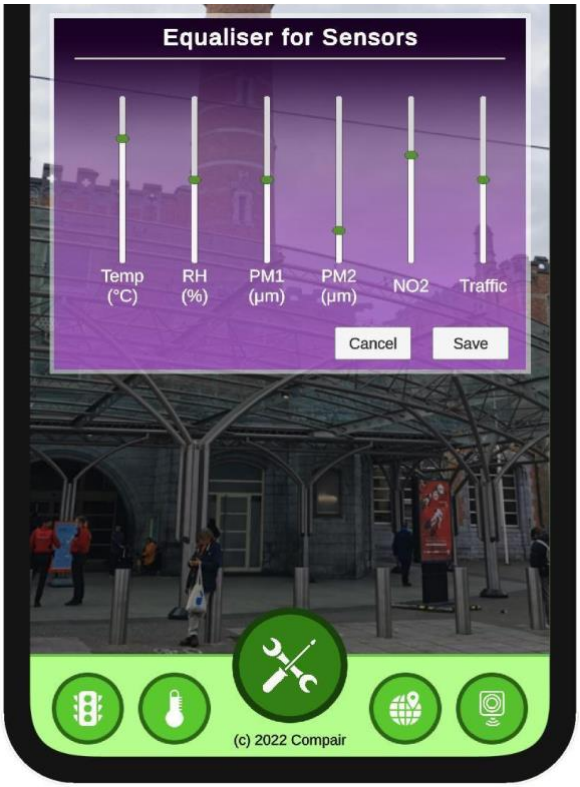

Figure 24: *Example of a setup for the equaliser.*

Date range and live data

The user can select a date range to restrict the data to a period of interest. Therefore DEVA will have a simple setup window for time (last hour, 3 hours, 6 hours), for day (current day, last week, last month) and possibly a more detailed period selection with a start and end date.

The user has also the possibilities to activate a real-time visualisation of pollutant data which means that the app would continuously receive data from the server. To distinguish such live data from other historical data, DEVA would add an extra visual (icons or surrounding frame) to the 3D objects. This visual can disappear after a while. The user can switch between the two modes or can have both working in parallel.

Figure 25: *Examples of data visualisations for specific sensor information (left) or histogram visualisation (a schematic representation so far) (right)*

4.2.3 Visualisation concepts of near and far data

The sensor data provided by the citizen science community are of a specific nature, which requires some analysis. The aim of COMPAIR is to visualise air pollution and traffic data provided by citizen science sensors distributed in some local communities. Although the project will offer a significant amount of sensors to our pilot partners, the distribution of sensors per several dozens of square metres will be limited. On other hand, the AR device offers a view in the near surroundings depending on the location, where the user is. This may be a wide field of view or a view obstructed by buildings, vegetation and objects (environment).

Near-n-Far

Due to this, we face the fact that only a few measurements from sensors will be available in the near surroundings. Therefore, any AR visualisation must offer capabilities to display information, which is further away and possibly not in the visible space. Hence, we developed a Near-n-Far concept that is based on an AR space supporting a visualisation of data in the near, middle and far distance. In this section, three different ideas are presented that allow for augmenting the scene with data at different distances to the user. The middle area is optional, and should be implemented in future versions, after having received some feedback from users during the pilot phase. For this purpose, the COMPAIR Framework uses a flexible internal array containing the parameters for the different zones e.g. distance from the user. The data are sorted in zones and then projected onto some mathematical surfaces or shapes

according to the methods. An interactive change of the method or of the mathematical formulas directly impacts the 3D representation, so the user can experience the possibilities without too much delay.

The DEVA Near-n-far solution is similar to some interactive solutions, named “proximity”, where the user can interact adequately with a device at different distances (De la Barré et al. 2009). Objects and pollutants data are to be placed in the attentiveness of the user, in the area between the user’s eyes and chest. Identifying the optimal location for products or other objects of interest is a research topic for many studies, most in the field of commerce and trade (Sigurdsson et al. 2009). Objects of interest, with the possibility of interacting with them, have to be closer and at arm/chest height to him. Objects far away could be addressed with a kind of remote controller (inside a render engine as raycast named).

Note:
In augmented reality, the identification of the environment to detect overlap and occlusion between real and synthetic objects is an important topic, but puts a lot of technology and new issues into play (by loading 3D city maps or activating some plane/object recognitions in the AR process). Here, the three proposed solutions (outlined below) attempt to solve these problems in a convenient manner. DEVA will allow the user to quickly activate the different solutions and change their parameters. During the upcoming user evaluations, we will identify the most appropriate concept.

Dome: The first concept is a-dome-like presentation as depicted in ***Figure 26***. It is based on an invisible half-sphere around the user. Different data are visualised depending on their proximity to the user, e.g. closer data are arranged in the lower part of the half-sphere, while distant data are located higher-up on the half-sphere. Depending on the environment and needs of the user, the perimeter of the sphere should be adapted. In a dense city location, a smaller sphere should be more suitable.

The height of the visual used by the projection onto the surface is calculated in relation to the distance of the values to the user (using the GPS coordinate of the user and the measurement location). So the system can adjust the height according to the distance and avoid overlapping of objects. The bottom of the sphere stays empty. Other attributes like size, colour, transparency or visual effect can be added as desired.

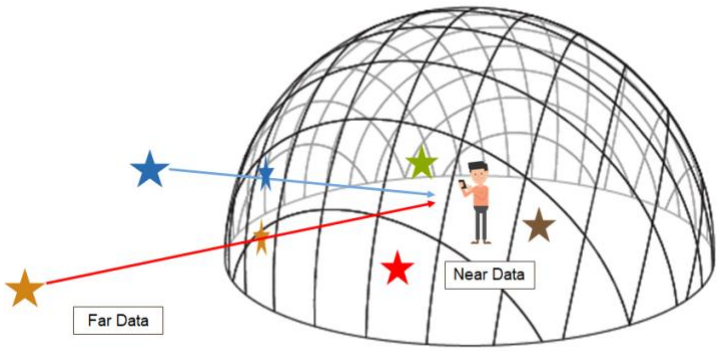

Figure 26: *Dome representation*

Theatre

Another possibility is an amphi-theatre like presentation, where data are arranged in different heights and with different sizes depending on their distance to the user.

Similar to the ***Dome*** projection, the ***Theatre*** mode offers the user a possibility to automatically arrange far data avoiding the overlapping of them. Near data remain in their original position and distant data are shifted along a curve (see ***Figure 27***). The further away a visual from the user is the higher it will be moved. But, in comparison with the ***Dome*** representation, the data are not projected onto a surface: only their height position changes. Other attributes like size, colour, transparency or nice visual effects can be added as desired.

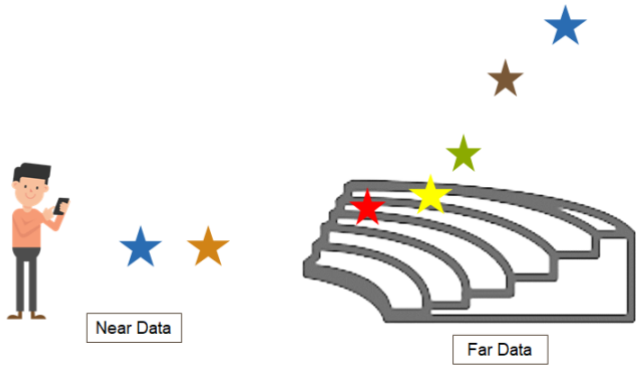

Figure 27: *Amphi-theatre representation*

Ring

A third option is the so-called ***Ring*** representation (see ***Figure 28***). The user is surrounded by a virtual ring and an optional compass, where north-south directions are assigned. This so-called data ring contains information about sensor data further away. The user can rotate the data ring (compass) and get information about far data in any direction.

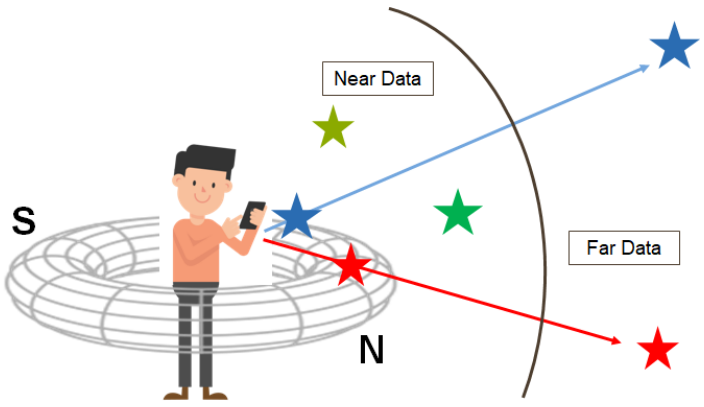

Figure 28: *Ring representation with the (optional) compass*

Here, pollutant and traffic data are projected onto a geometry model of a ring, surrounding the user as a swimming ring. Other shapes are possible. In comparison with the two previous methods, the original 3D visuals remain in the AR scene and would be highlighted by a user

selection on the ring, in parallel. Consequently, the user can always view the real data and the projected ones.If too much data needs to be presented on the ring, then a filter shall be used (e.g. distance or view frustum filter). Other attributes like size, colour, transparency or nice visual effects can be added as desired.

Note:
DEVA may not give to the user any restrictions for the setup of parameters for any of the three solutions, thus the user can experience them interactively. The parameters will be saved in the user's profile.

4.2.4 Interaction Concept

DEVA is programmed to run on mobile devices, whose basic interaction is *touch interaction* in various ways. To design interactions, Unity incorporates a flexible system, called Input System, to define and map user actions (touch, gamepad, keyboard, mouse, etc.) in method calls. The touch interface implements all possible combinations of touch interactions for 1 up to 10 fingers.

DEVA offers the following interaction forms:

- **Touch** on a specific point and select (1-finger). This is usually used to operate the GUI or to select individual 2D/3D objects (via ray-casting).
- Perform the **pinch** gesture (bring thumb and pointing finger close together and open again). This is usually used for zoom-in/zoom-out functionality or to define regions for multi-selection or grouping.
- **Slide** (or strip) over the screen (1-finger). This is used to move a window or object(s) over the screen. In DEVA, this function has to be evaluated.

3D-Beam: A more sophisticated possibility is the 3D beam interaction (see ***Figure 29***). Depending on the orientation of the mobile device, a so-called beam or laser pointer allows the user to select different data in the augmented 3D space. A final click provides the detailed information about the selected object or triggers further interaction. The beam function would be initiated as a normal 1-finger pointing in combination with a time delay (½ to 1 seconds). The beam stays active until the user removes the finger from the screen. The object is "virtual touched" by the beam and shall be highlighted to inform the user of the focus.

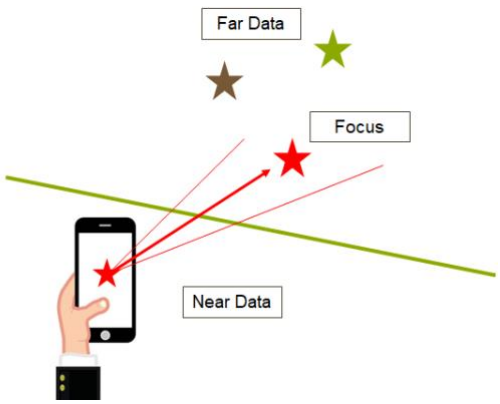

Figure 29: *3D beam interaction*

In order to avoid overlap between the finger on the screen and the rendering in the see-through area, it would be desirable to define a dedicated location (a round button, a beam “hot-spot”) where the user has to put the finger to interact. From the design side, this would be similar to the control elements from mobile games, where the gamepad is simulated onto the screen. In ***Figure 29***, this situation is visualised: the user's finger is below the used ray.

5 Implementation

5.1 Functionalities of the first prototype

According to the GA, a first closed alpha version is available for testing by M12 (end of October 2022). The first prototype provides the following functionalities, whereby the focus is also on the user interface design, the development of the general framework and the definition and implementation of data structures:

- The overall design of the GUI is completed.
- The user can login in the system and edit his/her parameters.
- Setting forms for parameters of most important modules are completed.
- Connectivity to the COMPAIR Data Manager is running.
- Visualising 3D observation data in the see-through window is running.
- Filtering and selecting some sensors by type is completed.

It is important to note that all the screenshots presented in this document are stemming from the implementation of DEVA. These are not mock-ups.

5.2 Development in Unity

The development of DEVA is performed with the Unity Editor (www.unity3d.com) in its current LTS version 2021.3.x (patch number has no importance). The project uses the base techniques of Unity: assets, components, prefabs, scripts, scenes, plugins, package systems, etc. The Unity project holds a specific AR framework for the COMPAIR project: the “COMPAIR Framework”. This framework also uses two Fraunhofer HHI packages for XR and Mobile XR. In these frameworks, developers should find useful scripts, prefabs, images (for icons and materials/textures), setups, objects and scenes.

DEVA Scene

DEVA was developed in only ONE scene. The level system of Unity is not used for this because mainly all elements must be accessed on runtime. The scene is well managed: the number of scripts was reduced to a minimum. A main manager script (CompairManager) manages the whole and uses a state-machine and other control methods to supervise the activities (screens, user interactivities, etc.). ***Figure 30*** shows a part of the project.

GUI

The scene does not use the new UI Toolkit from Unity because it is in developpement and has too many changes in the last year decreasing the productivity. DEVA uses the conventional GameObject based system. For user input, DEVA uses the new Input System.

Augmented Reality

To access the AR function of the mobile devices (Android, Apple), Unity offers the AR Foundation framework. This framework joins all the capacities proposed by the two actors. DEVA uses only the AR Foundation, but no individual libraries.

Render Pipeline

To achieve optimal rendering with some new graphical techniques, the project uses the Universal Render Pipeline (URP), a configurable and customizable rendering pipeline, optimised for mobile devices.

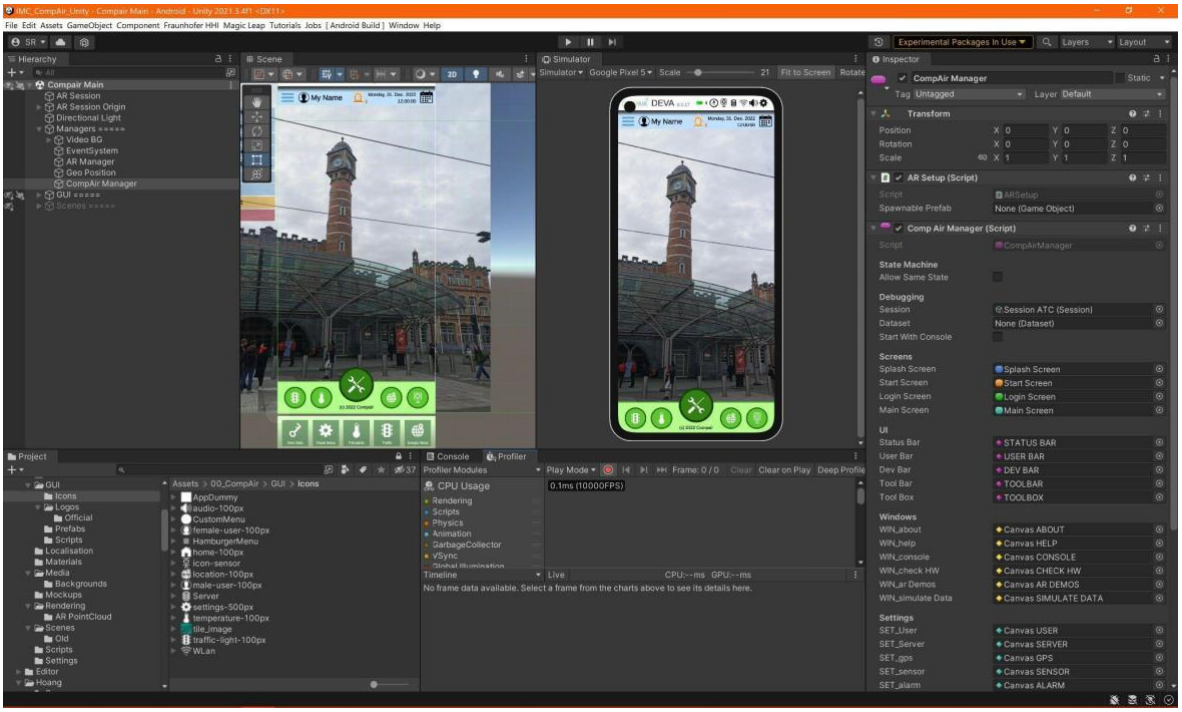

Figure 30: *Unity Editor window showing the DEVA project*

5.3 Publishing of DEVA for Android and iOS

The DEVA app should be published on Android devices as well as Apple iOS devices. Thanks to the cross-platform capabilities of Unity, we are able to provide applications for both operating systems. For both brands, we would support smartphones and tablets. It is possible to use the device in portrait or landscape mode, but devices have an optimised run mode, especially for the compass function, depending on the hardware design.

Android

Publishing apps for Android devices is performed in two ways: via the Play Store from Google, or via a direct-copy of the app with an USB cable in developer mode to the device. The latter approach is used in the very first phase as long as the provision on the Play Store is not yet

available. A direct-copy allows users to manage the app location and version on the device. The copy process takes only a few seconds.

Apple iOS

Publishing apps for iOS requires more effort because the app has to be compiled on an Apple Mac computer and published on the Apple Store. For this, developers need an active account by Apple (Apple-ID). After this, apple users can install the app from the store as they do other apps.

6 Test and Evaluation Strategy

6.1 Test strategy

The project follows strictly an agile development concept. For each of the different tools developed in the project, e.g. dashboards or DEVA, a so-called product owner (PO) is assigned. This person acts as the link between the developers i.e. the technical partners and the end-users, which are the pilot partners in COMPAIR. Based on the user requirements, a number of small tasks have been defined by the technical and pilot partners. The tasks are grouped in so-called sprints, which last for four weeks. Towards the end of each sprint cycle, the individual tasks must be completed and reviewed by the product owner. In a dedicated review meeting, the results are presented and then evaluated. Based on the outcome, the task is set as completed or rejected for further revision and update. To summarise, a very short development, evaluation and testing cycle guarantees immediate early stage identification of issues and related risk management.

6.2 Evaluation Plan

Beside the agile development approach, the project has set several milestones at which specific versions of our tools are handed over to the four pilot partners for testing with project external users. This is represented by the four different phases as shown in ***Figure 31***.

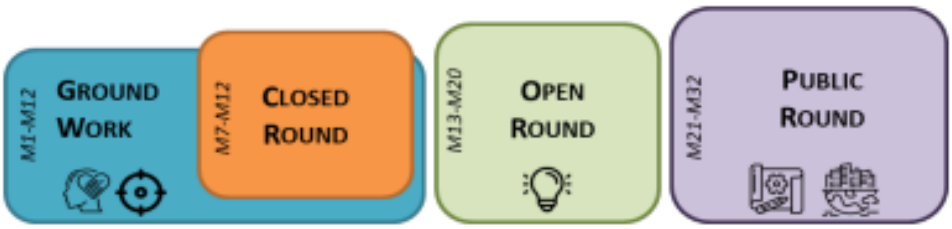

Figure 31: *Different phases of the COMPAIR strategy taken from the GA*

By M12 a closed alpha version will be ready and provided to the pilot partners for the Open Round. An open user group will be using DEVA and pilot partners as well as the technical partners will receive feedback. This can be of quite different nature such as:

- Obvious bugs in the implementation (buttons do not work, application crashes, etc.)
- Missing functionalities
- Feedback on the design of the graphical user interface
- Misleading naming of GUI items and text elements

During this phase, further functionalities will be implemented and feedback will be taken into account.

By M20 a revised open beta version is completed and the open public version will be released for user testing to the pilot partners, but also to the public.

Beside the external users involved in the different pilot cities, COMPAIR requested volunteers interested in testing and using the different applications. So far we received around 20 expressions of interest to volunteer for the project submitted via website.

The complete development process is visualised in ***Figure 32***.

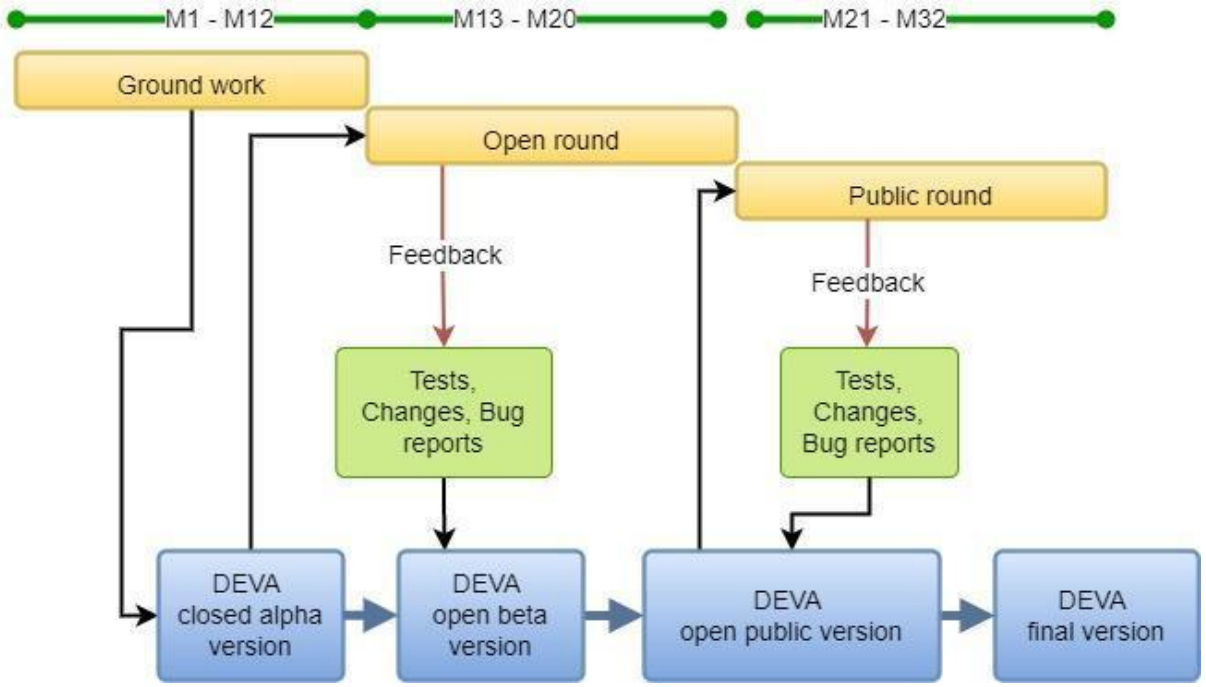

Figure 32: *Development plan for DEVA during the course of the project*

7 Conclusion

This public deliverable presents the current state of conceptualisation and development of DEVA, the COMPAIR Augmented Reality app. The pilot partners and their end users have been involved right from the beginning of the design and development. Feedback received from various workshops during summer 2022 has already been included in this process.

The main effort in the first 11 months of work on DEVA was put on the following:

- the overall system structure of DEVA including the pipeline development
- the definition and implementation of data structures following the OGC standard
- the setup of communication infrastructure in collaboration with ATC by using Rest API and WebSockets. The OpenAPI framework is used to support the interface definition and collaboration between the partners
- a Graphical User Interface implementation following responsive design principles
- the development of novel visualisation concepts in the context of augmenting environmental data from air pollution and traffic sensors

The current state provides the foundation for the first closed alpha version of DEVA by month 12 of the COMPAIR project. Based on detailed evaluation and testing strategy, we will further extend the functionalities of the app and integrate the user feedback provided by the partners in the different pilot cities.

Thanks to the dynamic and flexible software concept of DEVA, extensions and modifications can be implemented with minimum effort.

8 References

Books/Articles

De la Barré, R., Chojecki, P., Leiner, U., Mühlbach, L., Ruschin, D., (2009), Touchless Interaction-Novel Chances and Challenges, Proc. of the 13th International Conference on Human-Computer Interaction. Part II: Novel Interaction Methods and Techniques, Berlin, Germany, Springer Verlag, Heidelberg, Germany, pp. 161–169, July 2009, *doi: 10.1007/978-3-642-02577-8_18*

Hamari, J. (2019). Gamification. Blackwell Pub, In The Blackwell Encyclopedia of Sociology, Malden. pp. 1-3. https://doi.org/10.1002/9781405165518.wbeos1321 Archived 2022-02-26 at the Wayback Machine

Khanchandani, K., Shah, M., Shah, K., & Panchal, V. (2021). A Review on Augmented Reality and Virtual Reality in Education, Int. Research Journal of Engineering and Technology (IRJET), Vol.8, Issue 2, February 2021, https://doi.org/10.1186/s13173019-0084-8.

Mathews, N.S., Chimalakonda, S., & Jain, S. (2021). AiR: An Augmented Reality Application for Visualizing Air Pollution. *2021 IEEE Visualization Conference (VIS)*, 146-150.

Milgram, P., Takemura, H., Utsumi, A., Kishino, F. (1994). Augmented Reality: A class of displays on the reality-virtuality continuum, *Proc. SPIE vol. 2351, Telemanipulator and Telepresence Technologies*, pp. 2351–34, 1994.

Nurgazy, M., Zaslavsky, A., Jayaraman, P.P., Kubler, S., Mitra, K. and Saguna, S., (2019). CAVisAP: Context-Aware Visualization of Outdoor Air Pollution with IoT Platforms, 2019 International Conference on High Performance Computing & Simulation (HPCS), 2019, pp. 84-91, doi: 10.1109/HPCS48598.2019.9188167.

Smyth, C.S., (2022). Draft OGC GeoPose 1.0 Data Exchange Standard, Open Geospatial Consortium, 2022

Perey, C., Morley, J.G., Lieberman, J., Smith, R., Salazar, M., Smyth, C., (2022). OGC GeoPose Reviewers Guide, Open Geospatial Consortium, 2022

Patel, S., Panchotiya, B., Patel, A., & Aishwariya Budharani, S.R. (2020). A Survey: Virtual, Augmented and Mixed Reality in Education. *International Research Journal of Engineering and Technology,* Vol. 9, Issue 5, May 2020.

Torres, N.G., Campbell, P.E. (2019). Aire: visualize air quality. In ACM SIGGRAPH 2019 Appy Hour (SIGGRAPH '19). Association for Computing Machinery, New York, NY, USA, Article 1, 1–2. https://doi.org/10.1145/3305365.3329869

Vargas, J.C.G., Fabregat, R., Carrillo-Ramos, A., Jové, T. (2020). Survey: Using Augmented Reality to Improve Learning Motivation in Cultural Heritage Studies. Applied Sciences. 10. 897. 10.3390/app10030897.

Sigurdsson, V., Saevarsson, H., Foxall, G., (2009), Brand placement and consumer choice: An in-store experiment. Journal of Applied Behavior Analysis, September 2009 42(3):741-5; DOI: 10.1901/jaba.2009.42-741

Websites

ETSI-ISG ARF (2022), Portal: https://www.etsi.org/committee-activity/activity-report-arf?highlight=WyJjbG91ZCIsImNvbXB1dGluZyIsImNvbXB1dGluZyciLCJjb21wdXRpbmcnLCIsImNvbXB1dGluZycuIiwiY2xvdWQgY29tcHV0aW5nIl0=; Standards: https://www.etsi.org/standards-search#page=1&search=&title=1&etsiNumber=1&content=1&version=1&onApproval=1&published=1&historical=1&startDate=1988-01-15&endDate=2022-09-14&harmonized=0&keyword=&TB=858&stdType=&frequency=&mandate=&collection=&sort=3

OGC SensorThings API Part 1: Sensing Version 1.1, (2022) https://docs.ogc.org/is/18-088/18-088.html

Open API Initiative (2022), https://www.openapis.org/

OpenAPI Generator (2022), https://openapi-generator.tech/

Swagger.io (2022), https://swagger.io/

Sokhanych, O., (2022). 14 best Augmented Reality furniture apps. https://thinkmobiles.com/blog/best-ar-furniture-apps/

The World Air Quality Index - WAQI. 2019. Air Pollution in the World: Frequently Asked Questions. http://aqicn.org/faq/

Meaning of triadic colours (2022): https://www.color-meanings.com/triadic-colors/

Unity3D - real-time game engine (2022), https://unity.com/

Unity - Shader overview (2022), https://docs.unity3d.com/Manual/Shaders.html