D2.2 – Initial GOEASY Platform and Pilots Reference Architecture

Deliverable ID	**D2.2**
Deliverable Title	**Initial GOEASY Platform and Pilots Reference Architecture**
Work Package	**WP2**
Dissemination Level	**PUBLIC**
Version	**1.0**
Date	**2018-05-31**
Status	**Text**
Lead Editor	**ISMB**
Main Contributors	**ISMB, FIT**

Published by the GOEASY Consortium

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 776261.

Document History

Version	Date	Author(s)	Description
0.0	2018-03-19	ISMB	First Draft with TOC
0.1	2018-03-23	FIT	Initial set of application/ pilot requirements added
0.2	2018-04-31	ISMB	Overall content added
0.3	2018-05-18	ISMB	Functional View and Security Perspective sections updated
0.4	2018-05-23	ISMB	Contributions from BQ, FIT and CNET integrated into the document. Overall update of the document.
0.41	2018-05-22	FIT	Changed sections related to the DBMS and updated Deployment Diagram
0.5	2018-05-25	ISMB	Contributions form FIT integrated and overall improvement of the contents
0.6	2018-05-28	ISMB, FIT, BQ	Inputs from FIT and BQ integrated into the document. Section 3 updated. Ready for internal review
0.6_FIT	2018-05-28	FIT	Comments and adjustments regarding internal review
0.7	2018-05-30	ISMB	Comments and feedback from FIT's and BQ's internal review integrated int the document
1.0	2018-05-31	ISMB	Final version, ready for submission.
2.0	2018-06-21	ISMB	Minor modifications based on reviewer's comments.

Table of Contents

1 Introduction

This deliverable is the first of a series of public reports providing a comprehensive overview of the GOEASY platform (GEP) and pilots reference architecture. There will be an updated version of the architecture description in deliverable D2.5 due in month 18 and a final version in deliverable D2.7 due in month 30.

First, the methodology used to achieve and document the architecture is presented. The architecture is the result of a bottom-up phase, where individual partner technologies had been in focus, and a top-down phase, where applications and platform services had been defined. The definition has been guided by the "Initial Visions, Scenarios and Use cases" collected in D2.1 of the WP2, and the projects' objectives deriving from the GOEASY DoA (Description of Action).

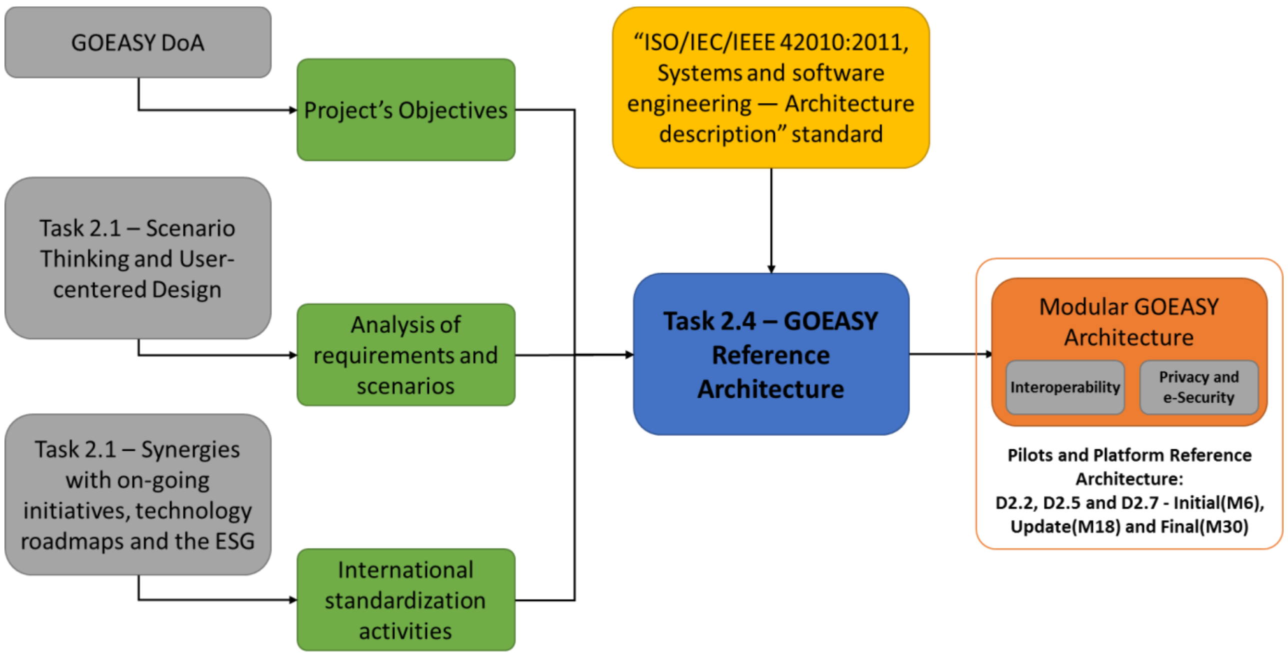

GOEASY DoA
Project's Objectives
Task 2.1 – Scenario Thinking and User-centered Design
Analysis of requirements and scenarios
Task 2.1 – Synergies with on-going initiatives, technology roadmaps and the ESG
International standardization activities
"ISO/IEC/IEEE 42010:2011, Systems and software engineering — Architecture description" standard
Task 2.4 – GOEASY Reference Architecture
Modular GOEASY Architecture
Interoperability
Privacy and e-Security
Pilots and Platform Reference Architecture:
D2.2, D2.5 and D2.7 - Initial(M6), Update(M18) and Final(M30)

Figure 1. GOEASY Reference Architecture flow

The process used for software architecture design is based on ISO/IEC/IEEE 42010:2011 "Systems and software engineering - Architecture description". It implies a process that builds on a set of relevant architecture viewpoints. In this first release, the focus is on defining the following views:

1. The **Functional view**, describes the components, their functionality, and their interactions.
2. The **Deployment view,** describes how and where the system will be deployed and what dependencies exist, considering hardware requirements and physical constraints.
3. The **Information view** describes the data models and the data flow as well as the distribution. The viewpoint also defines how the distinction between fresh and latest values is achieved.

Finally, the pilot's use cases are described which will be relevant for the GOEASY project. The purpose of these use cases is to clarify how the GOEASY platform will work and which components are relevant for the different tasks.

1.1 Scope

This deliverable defines the initial architecture for the GOEASY platform. The requirements for the architecture can change during the course of a project and some aspects of the architecture need verification during development, and therefore the architecture described here cannot be considered to be final or complete.

Within the GOEASY work package structure, Work Package 2 (Agile Requirements and Architecture Engineering) is responsible for specifying the system architecture design. Having completed the previous steps in WP2, i.e. an initial set of requirements (MS1), this deliverable defines the system architecture, preparing for prototypal implementation to be carried out by the technical work packages.

The architectural description includes aspects related to the identification of the major system components, how they should interact and how their external interfaces should be defined.

Document sections are presented as follows. In section 2 the methodology for documenting the architecture will be introduced and the first version of the initial conceptual architecture of the GOEASY platform will be recalled. Sections 3 to 5 contain the different views of the architecture (i.e. functional, deployment and information views), which are then instantiated for specific technical use cases in section 7. Finally, two important aspects in the definition of the GOEASY platform are addressed in sections 8 and 9, in the presentation of the Security and Scalability perspectives respectively.

1.2 Related documents

ID	Title	Reference	Version	Date
[RD.1]	D1.2 - Initial Visions, Scenarios and Use Cases			

2 Architecture design

In this chapter we present the methodology and individual steps that were taken to achieve the architecture that is presented in later chapters.

The architecture definition process was defined based on the following principles:

- Each partner brought in technologies and software that were to be considered for the architecture.
- The project will deliver a cloud-based platform.
- The GOEASY platform (GEP) should be able to provide trusted and dependable location-based services (LBS) in mobility to various devices in a cross-platform fashion.
- The cloud-based GEP should implement communication approaches already proven to guarantee high-level of scalability and e-security.
- The GEP will offer APIs and SDKs to enable rapid development of mass-market applications, adopting open standards.
- The GEP is supposed to adopt existing multi-service federation approaches to facilitate and foster interoperability of LBS with other Internet of Things (IoT) platforms, with special focus on e-security aspects.

2.1 Methodology

This section presents the key concepts of the methodology for software architecture design adopted in GOEASY. In particular, the basic design references and principles adopted in the architecture definition process are presented, thus enabling the occasional reader to better understand the design choices taken by the project consortium. Experienced readers, aware of the ISO/IEC/IEEE 42010:2011 standard, might want to skip this section.

2.1.1 Software Architecture ISO/IEC/IEEE 42010:2011 Standard

The process used for software architecture design in GOEASY is based on ISO/IEC/IEEE 42010:2011 “Systems and software engineering - Architecture description” [1]. Such a standard establishes a methodology for the Architectural Description (AD) of software-intensive systems. It implies a workflow, which includes the following steps:

- Identify and record the **stakeholders** for the architecture and the system of interest;
- Identify the architecture-related **concerns** of those stakeholders;
- Select and document a set of **architecture viewpoints** which can address the stakeholder concerns;
- Create **architecture views** (one view for each viewpoint) which contain the architectural models;
- Analyse **consistency** of the views;
- Record rationales for architectural choices taken.

The IEEE 42010:2011 standard extensively uses **viewpoints** and **views** to document different aspects of a software system allowing to focus on specific concerns and issues, while at the same time ensuring an overall consistency of the architecture design. **Viewpoints** are collections of patterns, templates and conventions for constructing one type of view. One example is the functional viewpoint (and therefore the functional view) which contains all functions that the system should perform, the responsibilities and interfaces of the functional elements and the relationship between them.

2.1.2 Definitions

Relevant definitions for the definition of the GOEASY reference architecture are presented in Table 1. They are derived from [1] and from the extensions provided by Rozanski and Woods in [2].

Table 1. Relevant ISO/IEC/IEEE 42010:2011 Standard Definitions [citation]

Architecture	Fundamental concepts or properties of a system in its environment, embodied in its elements, relationships, and in the principles of its design and evolution. In other words, it's the concept of a system's structure, properties, interaction with its environment, etc.
Architectural Description	Work product used to express an architecture, such as component diagram, data flow diagram, etc.
Stakeholder	An individual, team, organization, or classes thereof, having an interest in the realization of the system
Concern	An interest in a system, which is relevant to one or more stakeholders. It might be a requirement (functional or non-functional) or an objective that a stakeholder has regarding the system
View	A set of models and descriptions representing a system or part of a system from the perspective of a related set of concerns
Viewpoint	Collection of patterns, templates and conventions for constructing one type of view
Model	A simplified representation of an aspect of the architecture, could be in form of a UML diagram
System-of-Interest	The system whose architecture is under consideration

The relationships between these concepts and the corresponding system-of-interest are shown in Figure 2.

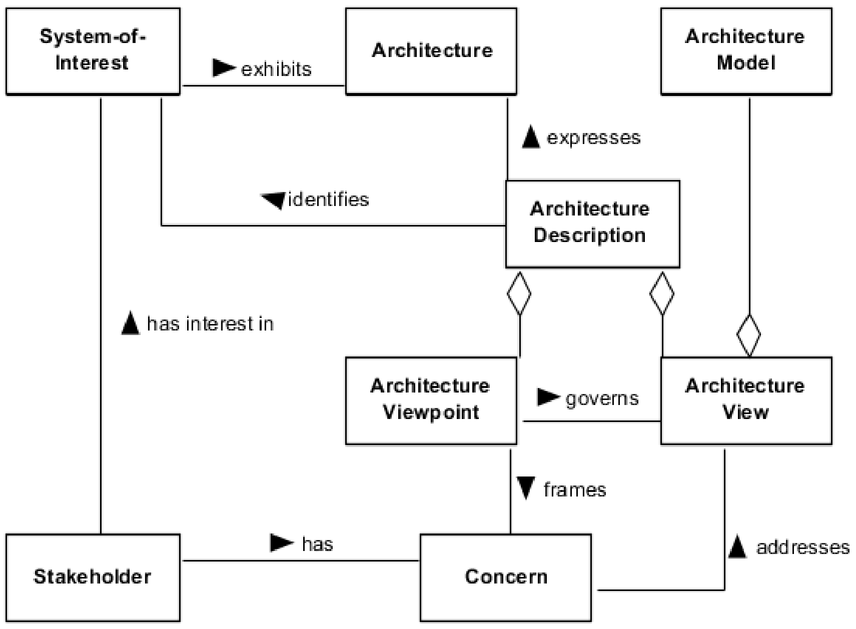


Figure 2 - Architecture description concepts (Adapted from [1])

The core of the modelling approach formalized within the IEEE 42010:2011 standard is composed of the architecture view and the architecture viewpoint concepts. According to [1], they are defined as follows:

- **Architecture viewpoint:** “Work product establishing the conventions for the construction, interpretation and use of architecture views to frame specific system concerns.”
- **Architecture view:** “A representation of a whole system from the perspective of a related set of concerns.”

In other words, a viewpoint defines the aims, the intended audience and the content of a class of views, and defines the concerns that such views will address e.g. Functional Viewpoint or Deployment Viewpoint.

According to [2] using vision (view) and point of view (viewpoint) to describe the system architecture can bring many benefits such as:

- **Separation of concerns**: Separating different models of a system into distinct (but related) descriptions helps the design, analysis and communication processes by allowing designers to focus on each aspect separately.
- **Communication with stakeholder groups**: Different stakeholder groups can be guided quickly to different parts of the AD based on their particular concerns, and each view can be represented using language and notation appropriated to the knowledge, expertise, and concerns of the intended readership.
- **Managements of complexity**: By treating each significant aspect of the system separately, the architecture can focus on each in turn and so help conquer the complexity resulting from their combination.
- **Improved developer focus**: Separating into different views those aspects of the system that are particularly important to the development team, helps ensuring that the right system is built.

2.1.3 Design process

In the design process of a software architecture, there are several principles that should be followed to ensure high quality. The relevant stakeholders should be engaged in the system design and their concerns taken into account, also dealing with the possible conflicting or incompatible concerns from different stakeholders. Besides, an effective way to communicate decisions and solutions should be implemented and the whole architecture design process should be flexible and pragmatic to be able to deal with changing requirements. The entire process should be technology-neutral.

2.1.3.1 Architecture Design Process

Rozanski and Woods have based the architectural design process on the following definition [2]:

"Architecture Definition is a process by which stakeholder needs and concerns are captured, an architecture to meet these needs is designed, and the architecture is clearly and unambiguously described via an architectural description."

The foundation for this process is the ISO/IEC/IEEE 42010:2011 standard and the GOEASY project used the process proposed by [2], which is aligned to such standard (see Figure 3).

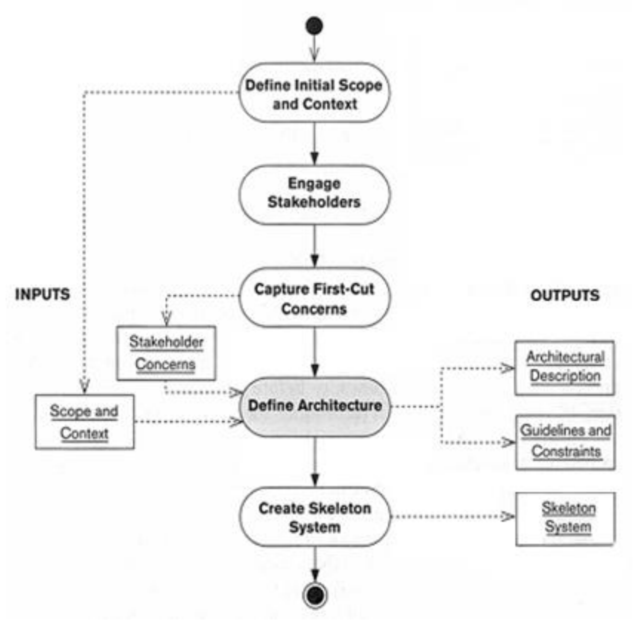


Figure 3 - Activities supporting architecture definition [2]

2.1.3.2 Architecture Viewpoints

Rozanski and Woods [2] defined seven core viewpoints to document a software architecture. They are:

- **Context viewpoint:** The context viewpoint describes interactions, relationships as well as dependencies between the system-of-interest and its environment. The environment includes those external entities with which the system interacts, such as other systems, users, or developers.
- **Functional viewpoint:** Describes the system's functional elements, their responsibilities, interfaces, and primary interactions. A Functional view is the cornerstone of most AD. It drives the shape of other system structures such as the information structure, concurrency structure, deployment structure, and so on.
- **Information viewpoint:** The information viewpoint describes the data models and the data flow as well as the distribution of data along the system components. This viewpoint develops a complete but high-level view of static data structures and information flows within the system being designed.
- **Concurrency viewpoint:** Describes the concurrency structure of the system and maps functional elements to concurrency units to clearly identify the parts of the system that can execute concurrently and how they are coordinated and controlled. This entails the creation of models that show the process and thread structures that the system will use and the inter-process communication mechanisms used to coordinate their operation.
- **Development viewpoint:** This is the viewpoint that addresses concerns from the developers' point of view. It describes how the software development process is supported, e.g. what conventions should be followed and how the artefact management will look like.
- **Deployment viewpoint:** Describes the environment into which the system will be deployed, including capturing the dependencies the system has on its runtime environment. This view captures the hardware environment that a system needs (primarily the processing nodes, network interconnections, and disk storage facilities required), the technical environment requirements for each element, and the mapping of the software elements to the runtime environment that will execute them.
- **Operational viewpoint:** Describes how the system will be operated, administered, and supported when it is running in its production environment. The aim of the Operational viewpoint is to identify

system-wide strategies for addressing the operational concerns of the system's stakeholders and to identify solutions that address them.

As the GOEASY project is in the initial phase of the relevant system design, this first release of the architecture will focus on the **functional**, **information** and **deployment viewpoints** to document the initial GOEASY architecture design.

2.2 Initial conceptual architecture, stakeholders and requirements

This section provides a brief summary of the high-level conceptual architecture and D2.1 outcomes updated at M6, which are fundamental to the description of the architectural choices depicted in subsequent sections.

The overall methodology for GOEASY is already described in D2.1 [RD.1]. It combines the user-centered design process according to ISO 9241-210 [citation] with an agile system development approach.

2.2.1 High-level conceptual architecture

GOEASY goal is to provide the technical and business foundations to enable a new generation of trusted and dependable mass-market location-based services (LBS) and applications for engaging, stimulating and rewarding citizens for more sustainable behaviors.

This will be done by establishing an open ecosystem built upon unique Galileo features (such as increased trust and improved availability). Also GOEASY platform will leverage open standards and platforms enablers to federate with existing authentication and e-security services, IoT, Smart City and Collective Awareness Platforms (CAPs).

The overall GOEASY concept is outlined in Figure 4.

- GOEASY Devices are trusted physical internet-connected devices.
- Application and services running on-board devices can access the trusted GOEASY platform via the open GOEASY APIs, which allow access to core GOEASY services such as the end-to-end authentication of position information, the trusted measurement and exchange of position information, dependable LBS or privacy-aware Database Management System (DBMS).
- To provide its services, the core GOEASY enabled devices and platform depends naturally on GNSS services. Moreover, GOEASY is externally supported by third-party services federated with the platform, cloud-based applications, beyond interacting directly with devices with GOEASY components on-board, can also access GOEASY services via the open GOEASY application API.

One of the main barrier to the diffusion of such services is associated with privacy and e-security concerns of users, who are (or at least, should be) rightfully worried in sharing their precise location with unknown service providers, which may make unknown uses of such information. For this reason, the main component of the platform is the GOEASY e-security infrastructure. It guarantees end-to-end position authentication and is tightly integrated with identity and privacy services such as the privacy aware DBMS, which stores the data provided by the users in a secure way, also providing selective, controlled access and anonymization services.

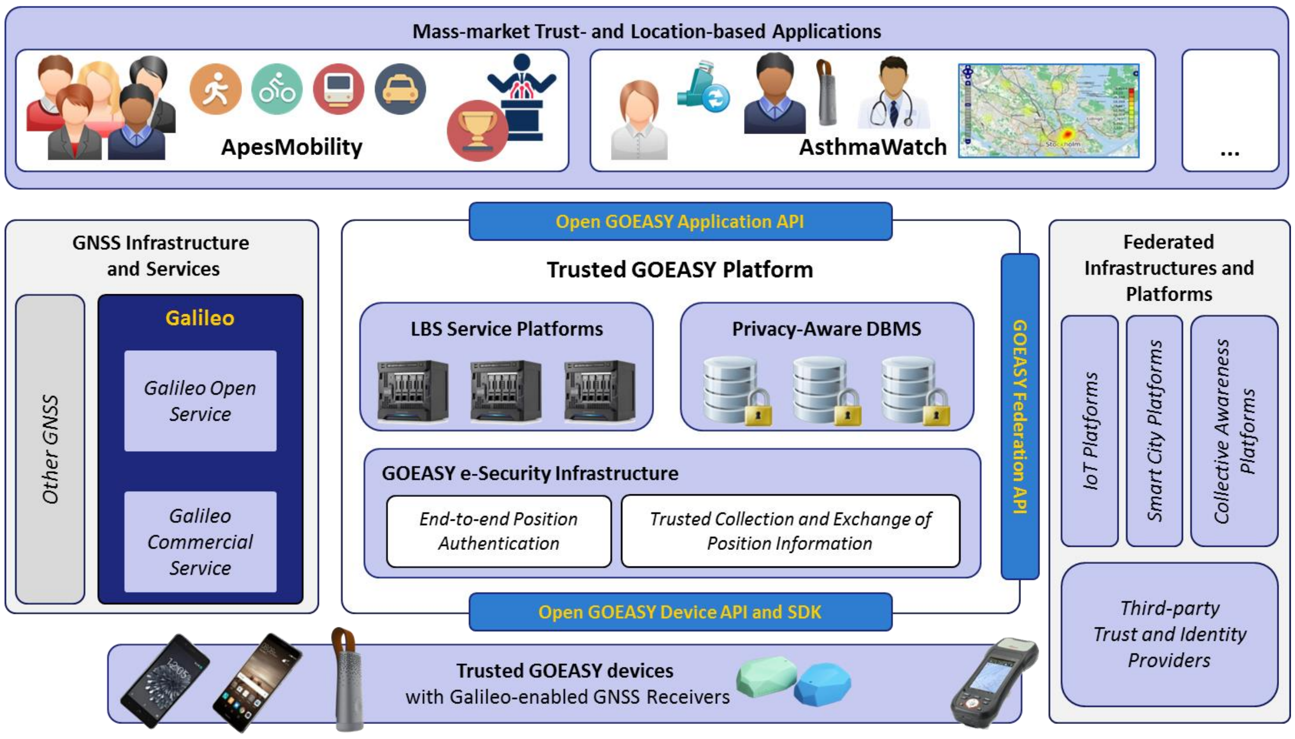

Mass-market Trust- and Location-based Applications
ApesMobility
AsthmaWatch
...
GNSS Infrastructure and Services
Other GNSS
Galileo
Galileo Open Service
Galileo Commercial Service
Open GOEASY Application API
Trusted GOEASY Platform
LBS Service Platforms
Privacy-Aware DBMS
GOEASY e-Security Infrastructure
End-to-end Position Authentication
Trusted Collection and Exchange of Position Information
GOEASY Federation API
Open GOEASY Device API and SDK
Federated Infrastructures and Platforms
IoT Platforms
Smart City Platforms
Collective Awareness Platforms
Third-party Trust and Identity Providers
Trusted GOEASY devices with Galileo-enabled GNSS Receivers

Figure 4. The GOEASY high-level conceptual architecture

To guide the design and the implementation of the GOEASY platform, the relevant stakeholders, initial scenarios and requirements have been identified in D2.1 based on the conceptual diagram from Figure 4.

2.2.2 Stakeholders

The GOEASY platform is designed following a user-centered design (UCD) process as explained in D2.1, which is a framework that offers multiple methods that are built on close interaction and discussion with users. The UCD process is applied iteratively. This allows to adapt to changing user needs and requirements as well as to limitations and problems which may occur during the project development at any stage.

In the first step the GOEASY vision and application scenarios were analyzed in order to understand and further specify the context of use. This included identifying and understanding the stakeholders of the platform and of the two pilot applications.

Table 2. GOEASY platform Stakeholders

Stakeholder	Description
GNSS Expert	A person, group or an organization responsible for providing expertise on GNSS domain. Responsible for providing expert advice about the domain e.g. rules, regulations, limitations, etc.
Security Expert	A person, group or an organization responsible for providing expertise on security and privacy aspects of the GOEASY platforms.
SW Application Developers	A single person or group of people that is able to exploit the project's SW results (partial and final) in order to build third-party applications on top, while providing feedback on their performance.
IoT Solutions Providers	Group of providers of IoT solutions intended as services, applications or other products interested in secure exchange of GNSS position data.

Table 3. ApesMobility Stakeholders [from RD.1]

Stakeholder	Description
User (Citizen)	A person (citizen) or group interested in using the application in order to get rewarded and/or to get inspiration received by fellow users to adopt new sustainable behaviors.
Municipality	An urban or administrative department, which is the contracted partner for greenApes and in which region the application is launched. Its interest is establishing more sustainable behavior in its community. Furthermore, it is responsible for giving guidelines to which behaviors are more crucial for the area and starting challenges to engage citizens in more sustainable lifestyles.
Organization	A partner organization of greenApes, which has proven to be sustainable. An organization has a commercial interest and uses greenApes to get more attention and gainsgains more customers. Partners can be e.g. restaurant, café, shop, e-commerce, cinemas or NGOs interested in the matter. Partners can reward users with prizes or discounts.
greenApes Srl SB	A company who developed the application, provides the technological infrastructure to run the application in municipalities and manages the community of users.

Table 4. AsthmaWatch Stakeholders [from RD.1]

Stakeholder	Description
Asthma patient	A person or group interested in using the application in order to prevent asthma attacks and get help in case of emergency.
Non-asthma patient	A person or group interested in doing outdoor activities (e.g. picnic, running), who mind doing it in an area with suitable conditions.
Municipality	An urban or administrative department, which is interested in supporting a healthy lifestyle in its community.

2.2.3 Requirements

According to the requirements emerging from discussions and interviews with the different stakeholders, multiple technical constraints were gathered and analyzed in D2.1. As first step in the architectural design, such technical constraints guided the definition and identification of the main GOEASY components.

2.2.3.1 Platform Requirements

As introduced in D2.1, requirements are documented as user stories in Jira[1], i.e., issue tracking system used during the execution of the project. Based on the revised and refined user stories, an initial set of user stories for the platform itself is documented and listed in Table 5.

[1] https://www.atlassian.com/software/jira

Table 5. GEP requirements exported from Jira

Key	User Story
GOEAS-45	As a SW application developer, I want to get an authentication service to make sure the location data is not spoofed.
GOEAS-46	As a GOEASY developer, I want to be able to control who has access to stored data in order to protect data from unauthorized access.
GOEAS-47	As a SW application developer, I want to transmit data to the GOEASY platform to make it available for location-based services.
GOEAS-48	As a SW application developer, I want to be able to aggregate data to hide details that endanger personal identifiable information.
GOEAS-49	As a SW application developer, I want to access data from the GOEASY platform to make use of it in my application.
GOEAS-50	As a SW application developer, I want to access aggregated data in order to ensure to not deal with personal data.
GOEAS-51	As a SW application developer, I want to make use of GOEASY's location-based services to enhance my application for its users.
GOEAS-52	As a SW application developer, I want to know about conditions regarding air pollution based on location data in order to inform users about possible dangers.
GOEAS-53	As a SW application developer, I want to certify specific mobility behavior of users in order to reward them for sustainable behavior.
GOEAS-54	As a SW application developer, I want to be able to detect specific mobility behavior of users without a data connection in order to reduce data and battery consumption of the mobile device.

2.2.3.2 Application/Pilot Requirements (from D2.1)

The initial set of requirements, which were documented in D2.1, are revised and refined. Although it is not part of this deliverable it is an opportunity to show the current status of the application requirements. The full list of user stories exported from Jira is attached in the Appendix A for ApesMobility and in Appendix B for AsthmaWatch.

2.2.4 Data Classification

During the architecture workshops and discussions that took place during the first 6 months of the GOEASY project, a lot of attention was given to the type and nature of data that needs to be stored-by and exchanged using the GOEASY platform.

In view of the recent General Data Protection Regulation (GDPR) [3] changes, the consortium has classified the position data that needs to be stored and/or exchanged through the platform in the following categories:

Table 6. Data classification in GOEASY

Data type	Description
Identifiable data	GNSS data that can be easily linked to the user and needed in emergency situations or when the user asks to share the specific location for any reason. Identifiable data is encrypted by the platform.
Pseudonym data	GNSS data that is communicated by trusted GOEASY devices using a pseudonym, and which is stored in a secure way without direct links with the ID of the final user.
Anonymous data	GNSS data that is uploaded into the GOEASY platform in a completely anonymous way before storage and/or exchange operations.

Thanks to this classification, the GOEASY platform is able to provide dependable Location-Based Services by processing anonymous or pseudo-anonymous datasets in an aggregated way, respecting current EU regulations concerning the exchange and sharing of personal data. GOEASY enables apps and services to inform clearly platform users about the personal data usage that will be made by the platform, in correspondence with the new GDRP regulations.

3 General Data Protection Regulation Considerations

To ensure full compliance of the GOEASY platform architecture with the new EU General Data Protection Regulation, which takes effect in May 2018, a deep analysis of the key principles of the regulation will be made continuously by the GOEASY consortium. In the following paragraphs a brief summary of the main conclusions from such exercise are presented, for each one of the seven principles of the GDPR. The definition of the first version of the GOEASY functional architecture has tried to take into consideration all of the following principles of data protection, as it will be explained in the subsequent sections.

3.1 Lawfulness, Fairness, and Transparency

User consent: explicit consent must be given by the users before their personal data is captured, processed, and stored. To support this requirement, personal data must be collected for a very specific, pre-defined purpose that must be clearly communicated to each user.

Data control: all EU individuals have the 'right to be forgotten' or request that their personal data be deleted from all data stores within the organization (and from its third-party suppliers). Users have the 'right to access' all personal data currently being held by the organization, and thus can request a copy of all data stored (a request that must be fulfilled within a month of receipt).

The **GOEASY** project will address this principle by providing appropriate information and tools for end-users of the platform to manage their data (with special attention to positon data) and be in complete control of the information they share with the applications leveraging the GOEASY platform.

3.2 Accuracy and Purpose Limitation

Data authenticity: all personal data must be accurate and kept up to date, giving the right to request that inaccurate information be corrected. Personal data can only be processed for its initial intended purpose; further processing of the collected data is prohibited without renewed consent from the individual.

GOEASY platform will guarantee the authenticity of all -position- data exchanged, and will allow users to rectify and/or correct any inaccurate information.

3.3 Data Minimisation and Storage Limitation

Limiting scope of data collection and storage: 'Privacy by Design and by Default' must be implemented, integrating data protection and privacy controls into the development of new business, applications, and services that somehow deal with personal data. Only the personal data that is absolutely necessary should be collected, and collected data must be stored for no longer than is required and that individuals must be informed about it.

3.4 Integrity and Confidentiality

Encryption and protection: Building on 'privacy by design', personal data should be rendered anonymous whenever possible, ensuring that EU residents will no longer be identifiable by their data. When this is not possible, GDPR mandates that controllers implement appropriate technical and organizational controls to safeguard the processing of any personal data. The effectiveness of implemented controls must be measured and documented on a regular basis.

Regarding the last two principles, the design and definition of the **GOEASY platform** and its components is following a privacy-by-design approach, implementing a security framework and privacy policies to safeguard personal data exchange while minimising the amount of personal data that needs to be transmitted over the platform. The GOEASY project is aiming at dealing with as much anonymized personal data as possible, as it will be explained in the following sections.

3.5 Accountability

Organizations processing personal data of EU residents must be able to give evidence to demonstrate compliance with all GDPR principles. As such, organizations that systematically collect and process personal data must appoint a Data Protection Officer (DPO).

Data breach notification becomes mandatory: All controllers and processors of personal data must designate a supervisory authority — a country Data Protection Authority (DPA) who, in addition to the DPO, maintains primary oversight of all data-processing activities. A data-breach notification scheme must be implemented to ensure all known breaches are reported to the appropriate DPA within 72 hours.

Data processors are also responsible for the maintaining the privacy and confidentiality of personal data. They must ensure adequate technical and administrative controls to protect personal data

4 Functional View

The GOEASY architecture specification adopts a component-based architectural style where the system functions are provided by a set of well-defined, self-contained, modules named “components”. Components talk to each other through well-defined interfaces, which make them decoupled from each other. With such architecture style, maximum flexibility and extensibility could be achieved since the system is not bound by certain component implementations. Instead, components could be easily replaced with new ones as long as they share the same interface. In summary, a plug-in oriented architecture is being promoted within GOEASY project and will be further developed within the updated versions of the architecture specification.

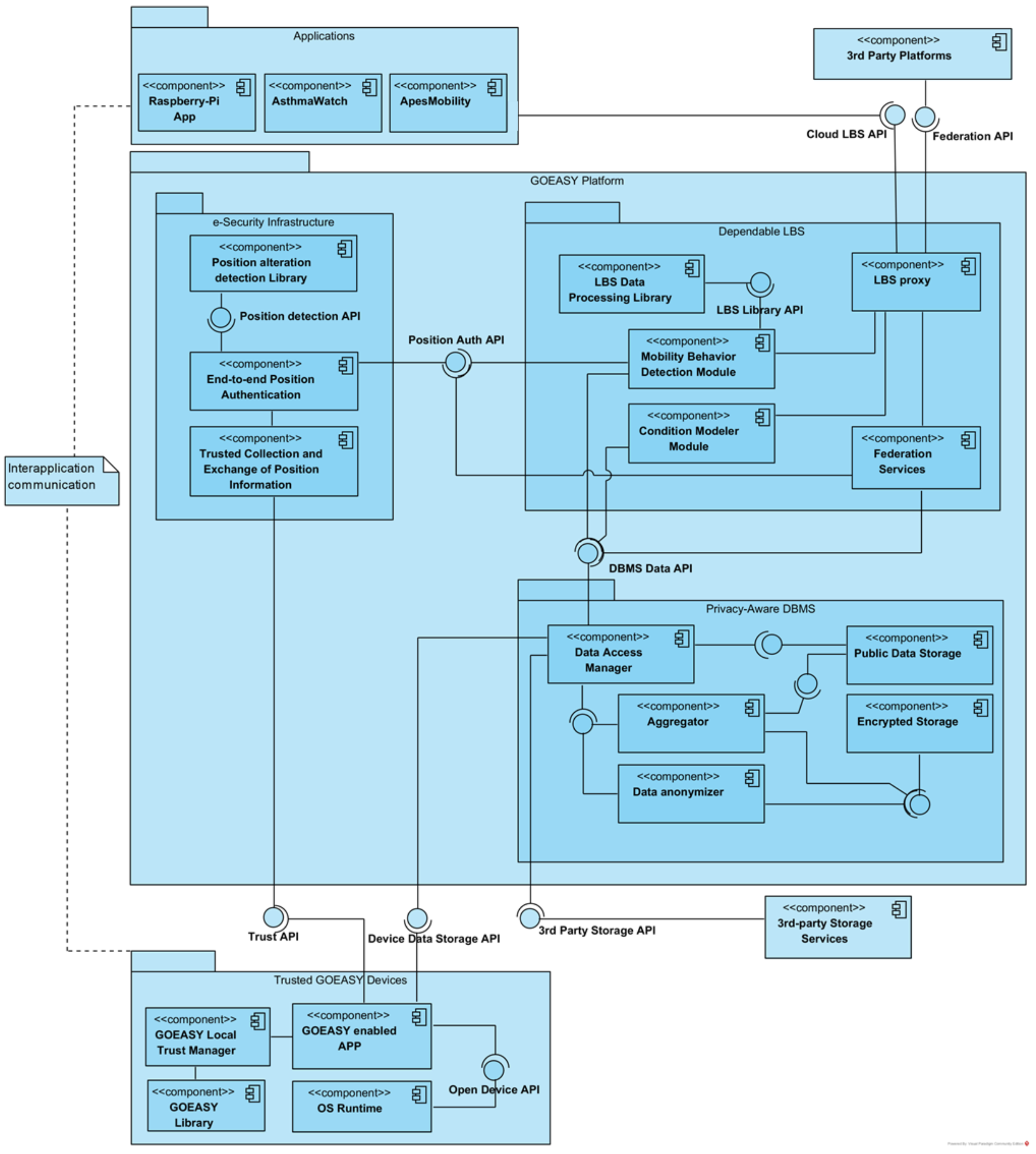

Applications
<<component>> Raspberry-Pi App
<<component>> AsthmaWatch
<<component>> ApesMobility
<<component>> 3rd Party Platforms
Cloud LBS API
Federation API
GOEASY Platform
e-Security Infrastructure
<<component>> Position alteration detection Library
Position detection API
<<component>> End-to-end Position Authentication
<<component>> Trusted Collection and Exchange of Position Information
Position Auth API
Dependable LBS
<<component>> LBS Data Processing Library
LBS Library API
<<component>> LBS proxy
<<component>> Mobility Behavior Detection Module
<<component>> Condition Modeler Module
<<component>> Federation Services
Interapplication communication
DBMS Data API
Privacy-Aware DBMS
<<component>> Data Access Manager
<<component>> Public Data Storage
<<component>> Aggregator
<<component>> Encrypted Storage
<<component>> Data anonymizer
Trust API
Device Data Storage API
3rd Party Storage API
<<component>> 3rd-party Storage Services
Trusted GOEASY Devices
<<component>> GOEASY Local Trust Manager
<<component>> GOEASY enabled APP
<<component>> GOEASY Library
<<component>> OS Runtime
Open Device API

Figure 5. GOEASY Functional Architecture - 1st Release

From a functional standpoint, the main components of the overall GOEASY platform are logically grouped into three main blocks: the GOEASY Privacy-aware DBMS, the GOEASY e-security Infrastructure and the GOEASY Dependable LBS (see Figure 5).

- The GOEASY Privacy-aware DBMS stores the data provided by the users in a secure way, also providing selective, controlled access and anonymization services.
- The GOEASY e-security infrastructure provides end-to-end authentication of location information and trusted measurement and exchange of position information
- The GOEASY Dependable LBS offer innovative services based on the use or exchange of location information to promote more sustainable mobility behaviors.

Furthermore, at the bottom of the diagram it is possible to see the GOEASY trusted devices, which can interact with the platform or with the applications directly.

The following sections better detail the three GOEASY main blocks, providing insights on their inner organization, on the foreseen sub-components and on the interactions occurring both at component and at package level.

4.1 GOEASY Platform

4.1.1 Privacy-Aware DBMS

The DBMS stores the data provided by the users in a secure way, also providing selective, controlled access and anonymization services by design, the public DBMS stores only pseudonymized/ anonymized data. The pseudonymization/ anonymization happens on the client side. Additional privacy and scalability can be added by utilizing the aggregation techniques. Data protection is the key here, this privacy friendly approach is possible due to the specific GOEASY use-cases.

Further privacy can be gained from an optional IP-protection layer. An optional encrypted storage for the user can be added if it is required by certain, future use-cases. An API on top of the DBMS allows data requests by a GOEASY LBS and possible 3rd parties.

Privacy-aware DBMS has three main components that will be described briefly in the next paragraph, while the general privacy approach implemented by the DBMS will be explained in detail in section4.1.1.5.

4.1.1.1 Data Access Manager

Data Access Manager component provides secure access services to the data storage, i.e., it determines access permissions to the public and/or private data storage.

4.1.1.2 Aggregator and Data Anonymizer

Aggregator and Data Anonymizer components perform processes that ensure privacy and security of user data. As shown in Figure 9, data in a certain geographic location are shown aggregated so that individual data are not used to indicate user density in a given location.

4.1.1.3 Public Data Storage

Public data storage stores data that does not include Personal Identifiable Information (PII) on the cloud for public access, i.e., for applications that provide LBS.

4.1.1.4 Encrypted storage

Encrypted storage stores data that contains PII. It also stores the mapping information that is used to associated anonymized data with the private data, e.g., when needed to be used to compute rewards to users.

4.1.1.5 GOEASY Privacy approach

The idea of privacy aware information bases is the provision of an aggregation and storage environment for LBS of GOEASY. On the one hand, the focus of the DBMS lies on the privacy awareness of its users. This can be achieved by consequently applying data minimization, data separation, data anonymization and data avoidance technique and it goes side by side with the use-cases specified for GOEASY (D2.1). On the other side, a DBMS also wants to provide a data environment for consumable LBS either from GOEASY or possible 3rd party services. Even though these two aims are contradictory, in the specific scope of GOEASY a satisfactory solution is possible, which grants data privacy for individuals and data richness for data hungry services.

Since the GOEASY use-cases do not require the storage of personal position data inside the DBMS, the contradictory aims can be brought together. An option for a private encrypted space for user's data – basically a cloud – can be optionally implemented if it is required by future use-cases.

The proposed core of user's privacy is the data **anonymization** and/or **pseudonymization**. Basically, every position data sample which is leaving user's cell phone (ApesMobility use case) or a mobile sensor platform (AsthmaWatch use case) towards the DBMS, does not need to be personalized. Figure 6 shows an exemplary, generic payload without a reference to user's real name.

In case a separated reward system is in place the identity (ID) of the user is needed to grant the exchange of rewards between the user and the GOEASY platform. Figure 6.also shows examples of such messages.

The proposed **generic payloads** (i.e. Geo_Message and Asthma) are sent to the DBMS. The DBMS can forward this data to the general public or to a specific LBS. The generic payload represents a pseudonymized data sample. If the reward system is not used, the ID can be skipped entirely, thus resulting in a fully anonymized data sample.

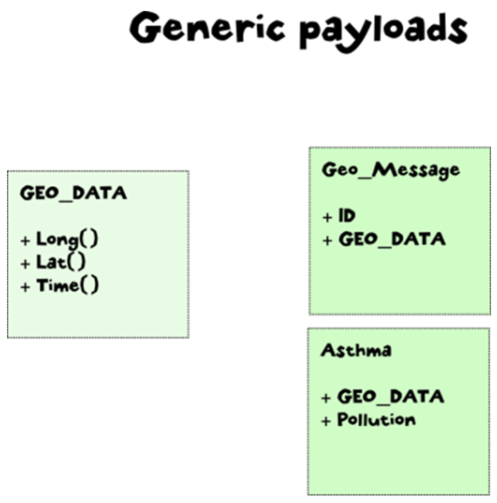


Figure 6. Generic anonymized payloads

An additional layer of protection can be applied to further blur the connection between the user and his IP. Figure 7 depicts the layer composed of multiple proxies (V1, V2, …, Vn) between the user and the Public DBMS (Pub DBMS). The layer only passes user's data to the DBMS. The actual data payloads are not addressed here.

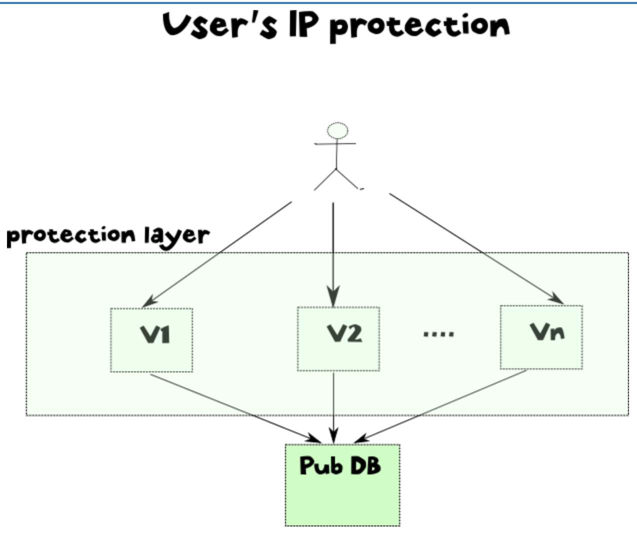


Figure 7. Additional layer of protection hiding user's IP

Another technique to protect user's privacy is data avoidance and aggregation. When done in the realm of the user (e.g. cell phone) no critical position data is forwarded to the DBMS. Considering the use-case where only distances are needed for a given LBS, the transmission of personal geolocation data can be totally avoided, thus preserving user's privacy. Figure 8 depicts a path distance calculation based only on sub-paths, not on actual geo data. The outcome is the same but forwarding of private data has been avoided.

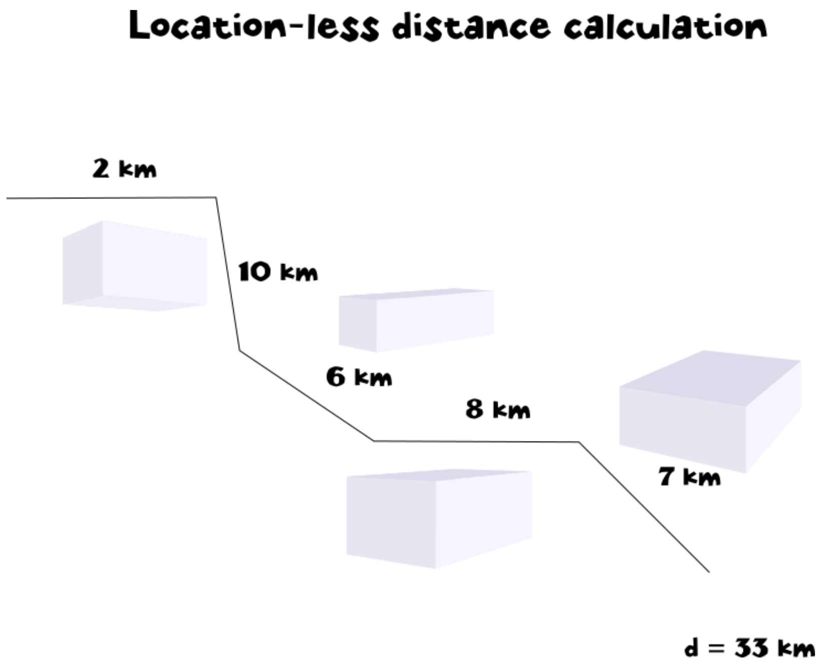


Figure 8. Location-less distance calculation

The upper example shows how privacy related data can be avoided by being **aggregated** by software running on the client side. Where possible, aggregation of data should be also a part of the DBMS itself. As an example, the anonymized but geolocated data stack can be aggregated to further expand the privacy of certain users.

Under specific circumstances, a limited sample of geolocated data can reveal a great deal of information, and with some signal processing this data can be linked to real users. For example, a user living in an isolated house in the outskirts of a city will probably produce only a few easily detectable moving patterns. If these patterns do not mix with other users, linking this data to a real person is possible. The Strava fitness tracking app can be named [4] as an example of disclosure of hidden data due to a small amount of data-samples,.

A possible solution to this can be further aggregation of data on the DBMS. To circumvent the described problem, mixing or aggregating of such low-sample areas is an option. The whole map can be divided into quadrants. Data inside one quadrant will be aggregated, which means the geolocation data of multiple data points is merged. This results in "blurring" or lowering the resolution while increasing the privacy of users from the low-sample areas. Figure 9 depicts how multiple data points can be merged together. Depending on the requirements the DBMS data aggregation can be executed before the actual storage of data. Here aggregated data is stored. The aggregation can also be performed afterwards when certain LBS request data from the DBMS. The second option preserves the original data.

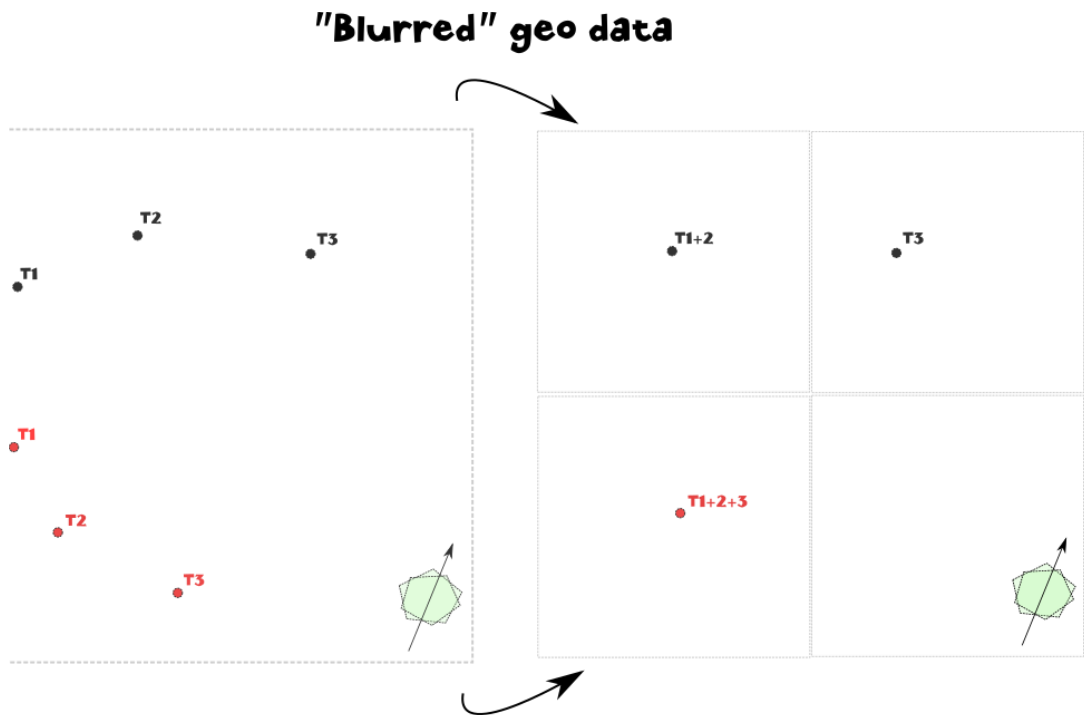


Figure 9. Lower resolution maps due to data aggregation using quadrants

An architectural overview of the proposed system can be seen in Figure 10. The User has a variety of possible connections to the DBMS system. Probably the most used one will be the path through the pseudo-anonymizer. As described before, the pseudonymization/anonymization happens on the client side. This happens through the consistent usage of generic payloads.

The aggregation of data can happen on client side or the DBMS side. It's heavily dependent on the use-case. When user's data is pushed to the DBMS it can pass an optional IP-Protection Layer (V1, V2, …, Vn) to further enhance the privacy level.

Optionally, the user can access an **encrypted storage** inside the DBMS. The user has full control over this space. The DBMS cannot process the data inside the encrypted partition.

The DBMS API allows LBS and 3rd party actors to retrieve data stored in the public storage. There may be different access rights for LBS and 3rd party stakeholders. The DBMS API may trigger aggregators to further increase the privacy level or reduce the amount of transferred data. Probably the most used use-case will be the generation of heat maps by a given LBS. The API needs to be able to cover such requests by providing areal information for requested data (e.g. location, pollution). Because the requests may cover vast areas resulting in big datasets, the DBMS needs to address the scalability issue at some point.

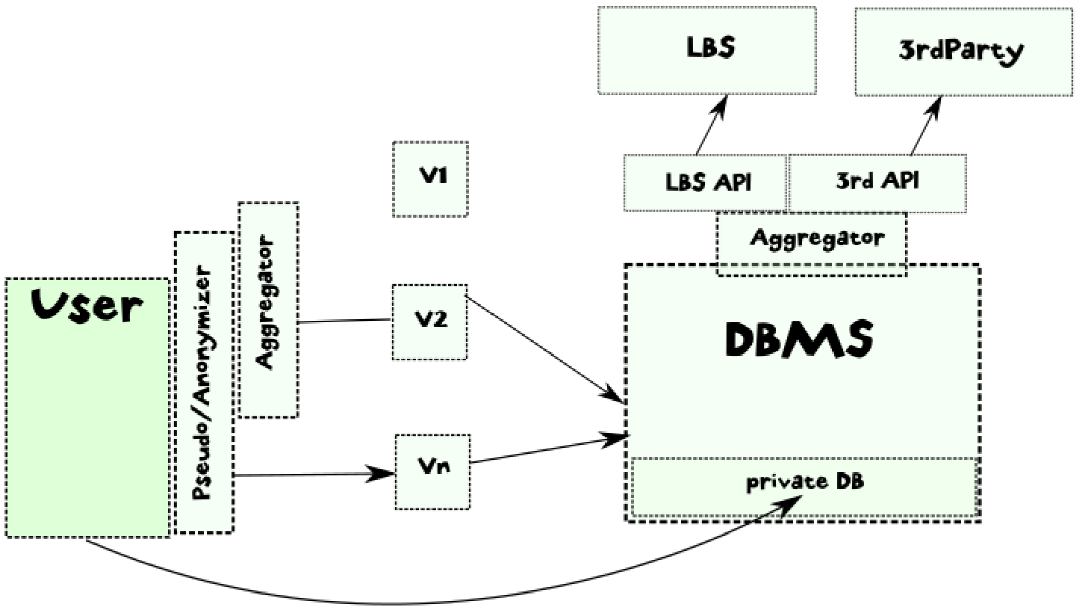


Figure 10. Privacy aware DBMS

4.1.2 Dependable LBS

This module is focused on the enabling technology for the backend cloud needed to support the envisioned location-based services. It consists of a set of enablers. This includes real-time event processing of incoming messages and observations, big data storage technologies for the management, structuring and aggregation of geo-based data. In general, this will result in components, built upon existing scalable cloud open enablers, suitable to store, selectively retrieve and analyse the data under real-time constraints provided by reference applications. In addition to the overall cloud enabling architecture we will develop:

- Real-time event processing enablers.
- Dedicated enablers suitable to support privacy-oriented features.
- Specific enablers to support the AsthmaWatch pilot.
- Specific enablers to support the ApesMobility pilot.

4.1.2.1 LBS proxy

Manages requests from the GOEASY applications and from 3rd party platforms federated with the GOEASY platform and forwards them to the specific LBS.

4.1.2.2 Mobility Behavior Detection Module

In charge of processing GNSS position data to detect the mobility pattern of a user or group of users. More specifically, this module must be able to recognize the travels made by a specific user using public transportation (possibly integrating data from the municipalities or the public transportation databases).

4.1.2.3 LBS Data Processing Library

Library containing relevant information and patterns needed to support the mobility behavior detection module, for example patterns and information needed to identify a certain travel type.

4.1.2.4 Condition Modeller Module

In charge of processing specific critera (e.g. air pollution) to calculate level of air pollution for specific areas. Furthermore, considering the calculated conditions, this module must be able to calculate the “healthiest” route given 2 geographical points (departure and destination), possibly integrating data from 3rd party applications (e.g. AstmaWatch data).

4.1.2.5 Federation Services

This module is in charge of guaranteeing interoperability with other platforms already developed and synergistic with GOEASY. Federation APIs will allow GOEASY to be federated with IoT platforms, like FIWARE enablers or LinkSmart; with Smart City platforms, such as the one developed by the ALMANAC project or the CityPulse one; and with Collective Awareness Platforms, Federation APIs will also be used to leverage third party trust and identity providers.

Thanks to this module, services can be combined transparently across different autonomous administrative domains based on common trust and technical agreements, also managing security and privacy issues related to the data flows generated by private data sources and transparently traversing different federated private/public networks. This is done exploiting techniques to model data flows and authorization mechanisms through descriptive semantic languages, as well as access control methodologies based on open standards and leveraging on dynamic, scalable authentication and authorization mechanisms supporting multi-domain environments.

4.1.2.6 Cloud LBS APIs

In order to enable the creation and growth of third party ecosystem we will define standardized, easy-to-use and open APIs for developers. We intend to base our APIs on the OGC (Open Geospatial Consortium) SensorThings format, which defines a data model for exchange of IoT data, refer to section 296.1.1. for more information on this subject.

The API will provide a high level semantic layer for app developers to use. Different existing ontologies will be evaluated to foster use of common concepts. In addition to the open cloud-based LBS APIs, the following will also be developed:

- Automatic tools for cross-platform API definition and documentation.
- Privacy-related extensions to the API i.e. sub-set of meta-APIs to gather, annotate and control how different privacy related aspects are handled in the remaining API resources.
- Alignment of APIs with location-based collective awareness platform.

4.1.3 E-Security Infrastructure

This package is in charge of guaranteeing end-to-end position authentication integrated with identity and privacy services offered by the privacy aware DBMS, which provides selective, controlled access and anonymization services.

Thanks to the distributed e-security infrastructure of the GOEASY platform it is possible to build a secure and reliable communication channel to exchange data between the GOEASY enabled apps and their twin in the cloud. To enhance the security of the communication between the mobile application and the cloud methods reported in Table 7 could be implemented.

Table 7. GOEASY e-security infrastructure features and possible solutions

Feature	Possible solutions
TRUST	Mobile phone two-factor authentication, which allows users to authenticate themselves, using their personal access login info, plus a one-time-valid second method connected with the device. Could be: a dynamic passcode consisting of digits sent to their mobile device by SMS, an app notification, or some combination of figures touched on the device screen. Existing resources: Event-based One-Time Password (OTP); Time-based One-Time Password (TOTP); Security Question Authentication; Challenge Response Authentication; Transaction Data Signing (TDS) and Mutual Authentication
AUTHENTICATION	X.509 certificates to manage public-key encryption. Roles and/or policies will be mapped to each certificate, so as to be able to change the authorization level, without ever touching the device.
SECURE COMMUNICATION	Using IoT messaging protocols for the exchange of data between the application and cloud such as XMPP guarantees a reliable and secure communication channel, using embedded security mechanisms for authentication and encryption, such as SASL and TLS.

4.1.3.1 Trusted Collection and Exchange of Position Information

This module offers the Trust API to trusted GOEASY devices in order to enable the secure exchange of position information through the platform.

4.1.3.2 End-to-end Position Authentication

After position information is made available on the device, mechanisms are in place guaranteeing the protection of data and security throughout the entire software stack, from the kernel drivers used to interact with the receiver, to the OS runtime and to the app developed for GOEASY. Figure 11 shows the approach proposed in the DoA for the end-to-end trust followed by the GOEASY platform.

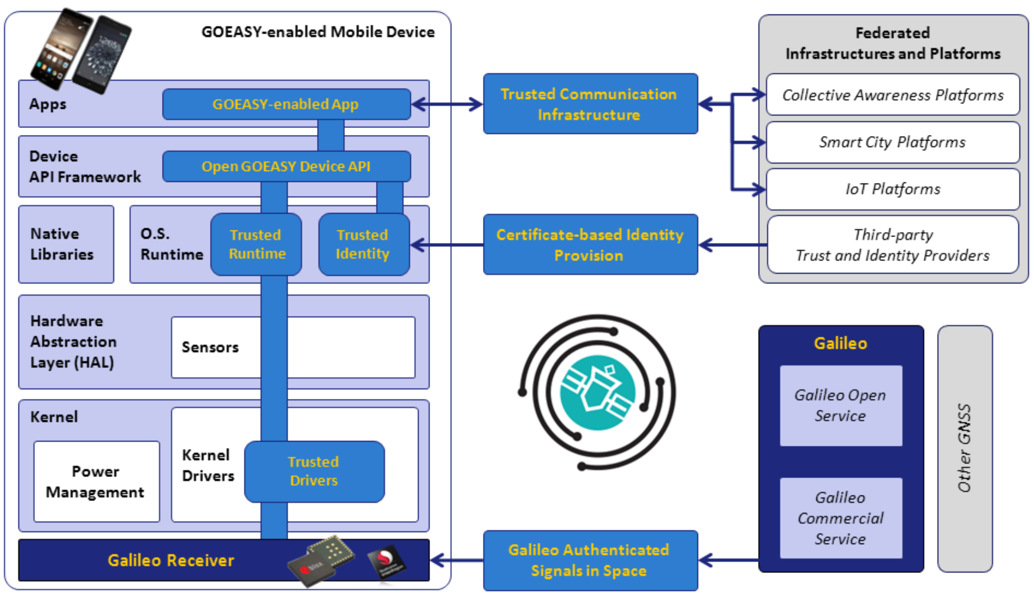

GOEASY-enabled Mobile Device
Apps
GOEASY-enabled App
Device API Framework
Open GOEASY Device API
Native Libraries
O.S. Runtime
Trusted Runtime
Trusted Identity
Hardware Abstraction Layer (HAL)
Sensors
Kernel
Power Management
Kernel Drivers
Trusted Drivers
Galileo Receiver
Trusted Communication Infrastructure
Certificate-based Identity Provision
Galileo Authenticated Signals in Space
Federated Infrastructures and Platforms
Collective Awareness Platforms
Smart City Platforms
IoT Platforms
Third-party Trust and Identity Providers
Galileo
Galileo Open Service
Galileo Commercial Service
Other GNSS

Figure 11 - The GOEASY approach to end-to-end trust

4.1.3.3 Position Alteration Detection Library

Software libraries and configurations to detect software-based attempts to alter collected position information.

4.2 GOEASY Trusted Devices

Since Google announced Android Nougat (v7.0 and API level 24) mid 2016, there is a new GNSS subsystem available, which is widely increasing the number of potential features to be developed on top of Android OS, especially for those related to positioning augmentation and authentication. Android API does not provide a straightforward pseudorange associated to a GNSS timestamp, but other parameters as the "GNSS receiver's internal hardware clock value" or the "received GNSS satellite time at the measurement time". The computation process showing how the parameter made available by Android API need to be combined is detailed in [5] §2.4.

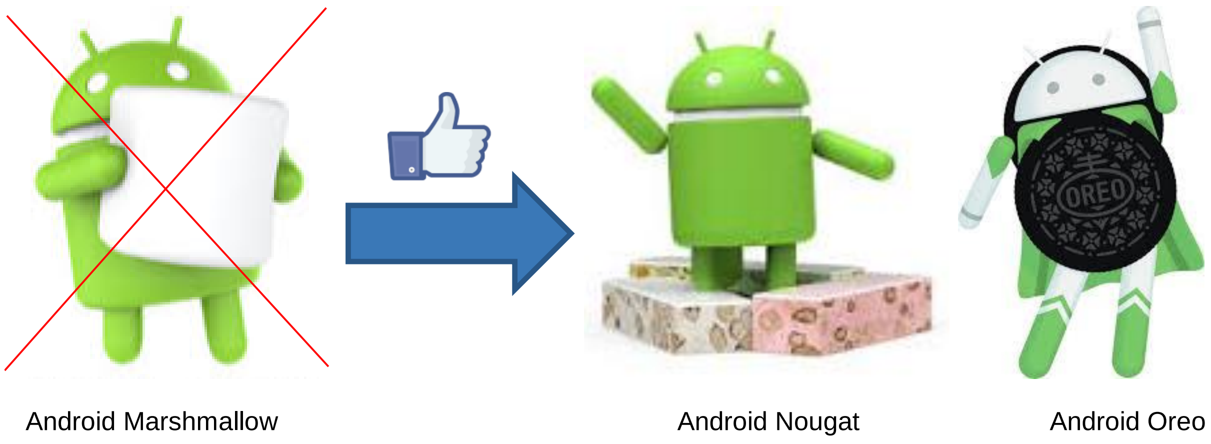


Figure 12. Android Nougat update enabled the new GNSS API including raw data.

Part of that new subsystem, there are three classes that are relevant in the scope of GOEASY project:

- GnssMeasurement
- GnssClock
- GnssNavigationMessage

Among all the parameters available from Android 7 onwards, the GOEASY consortium has prioritized the following ones that are critical to build the authentication system:

- GnssNavigationMessage class
    - Data of the reported GNSS message
    - Satellite ID
    - I/NAV message ID

Even the fact that Android API is available, not all the devices running Android 7.0 or above are able to expose the GNSS raw data. That is why the consortium is looking for potential countermeasures to include also authentication measures even in those devices not exposing GNSS raw data.

The initial scope of the project involved Qualcomm (BQ's chipset manufacturer) as a key partner to enable the raw data exposure on BQ devices (also other Qualcomm chipset based devices), but the chipset manufacturer has confirmed that there are no plans to support raw data exposure in none of their product lines, even in the high end chipsets.

Alternative solutions are being defined for those devices not supporting RAW GNSS data. Some of them are supposed to be part of the applications and some others will be exclusive of BQ phones.

To ensure that at least all the devices using the GOEASY solution could have a first "barrier", the proposal is to follow the "Pokémon Go" anti-spoofing protection model, including the following checks:

- Is the device rooted or enabling superuser permissions?
- Is the bootloader unlocked?
- Is the "allow fake position" option enabled in the developer options menu?

If the answer of any of these questions is "yes", then the position given by the device is not authenticated and therefore rejected. If the answer of all of them is "no", then the first authentication barrier is passed. As a second countermeasure, the proposal is to match the position given by the GNSS system (high accuracy) against the approximated position given by the network.

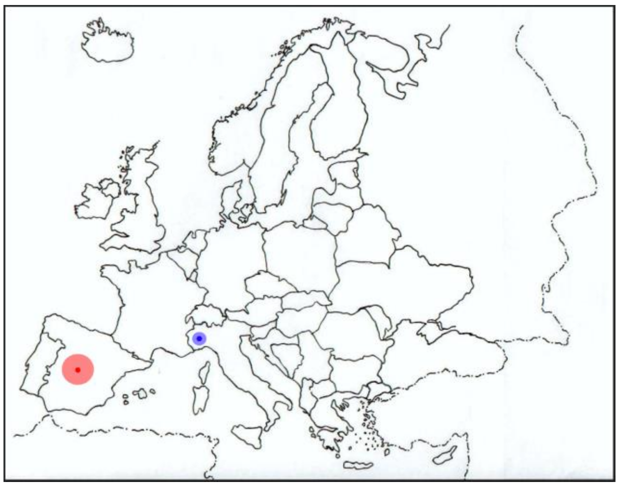


Figure 13. Example of position not authenticated (blue position is not a subset of red position)

Network position accuracy is much lower than the position given by the GNSS system, but it is a good approach to authenticate that the user is at least in the same area as indicated by the app.

We have to consider the GNSS position as a subset of the position plus the accuracy given by the network. On this regard, BQ field testing aims to confirm the reliability and stability of all the wireless communications, including calls, Wi-Fi and data browsing, GNSS navigation, and Bluetooth usage. According to the previous experience on that field, the position accuracy when using only cell network is in the range of 100-500 m, but could achieve values of 800m in some cases. The accuracy is slightly improved when using the support of the Wi-Fi network to a range of about 30-100m. It depends on the coverage of the router or the Wi-Fi network, and it is also related to the MAC information mapped by Google, but normally in the range of 30-100m. Finally, the GNSS positioning system has the highest accuracy among the available options in Android, with an average value in the range of 4-10m. In Figure 13 you can find an example of a position in blue that will be rejected and in Figure 14 a position in green that will be accepted. The circles in red represent the position given by the GSM/Data network:

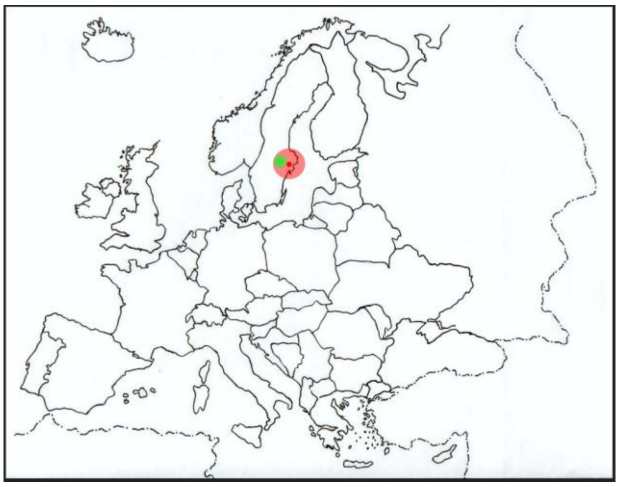


Figure 14. Example of position not authenticated (green position is a subset of red position)

4.2.1.1 Local Trust manager

This component is in charge of local (device-level) trust features and interacts with the platform e-security infrastructure components through the GOEASY enabled App, and more specifically through the Trust API offered by the Trusted Collection and Exchange of Position Information module.

4.2.1.2 GOEASY Library

In this library information needed to enable local trust features is stored and updated periodically.

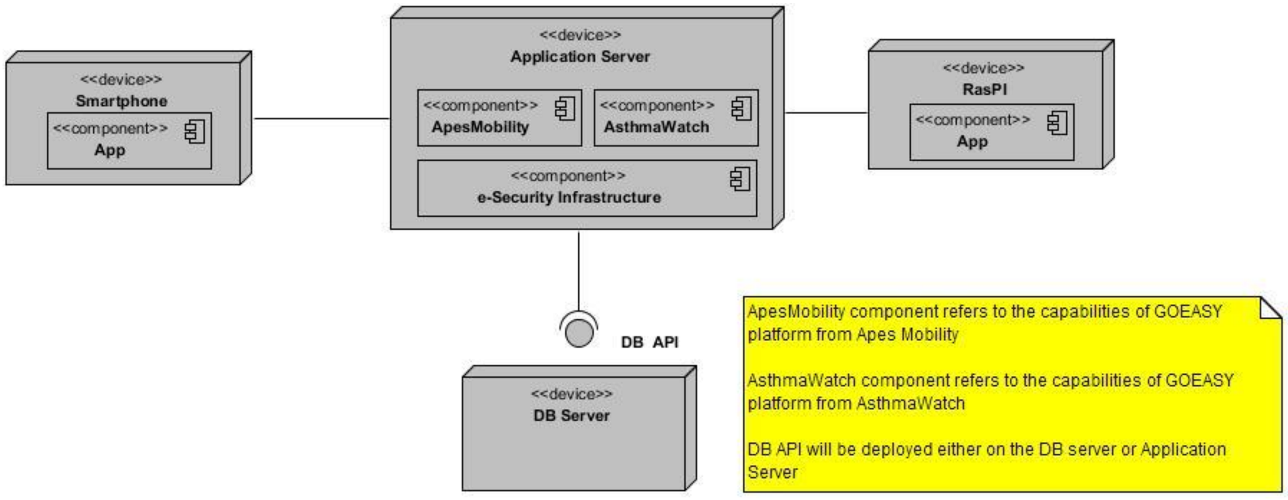
4.2.1.3 GOEASY enabled Apps

These are the mobile applications running on trusted GOEASY devices that consume one or more services offered by the GOEASY platform. At this moment, they are mainly three: the AsthmaWatch App, the ApesMobility App and the Raspberry Pi App. Inter-application communication between the mobile apps running on GOEASY trusted devices and the cloud-based version is possible without passing through the GOEASY platform (i.e. when performing operations that does not include the exchange of sensitive data or position information).

5 Deployment View

5.1 Overview

This section discusses aspects related to the GOEASY platform deployment. Subsequent paragraphs better detail such aspects tackling deployment of platform instances, of cloud-based solutions and federations.



<<device>> Smartphone
<<component>> App
<<device>> Application Server
<<component>> ApesMobility
<<component>> AsthmaWatch
<<component>> e-Security Infrastructure
<<device>> RasPI
<<component>> App
DB API
<<device>> DB Server
ApesMobility component refers to the capabilities of GOEASY platform from Apes Mobility
AsthmaWatch component refers to the capabilities of GOEASY platform from AsthmaWatch
DB API will be deployed either on the DB server or Application Server

Figure 15 Initial Deployment Model

GOEASY system is composed of client-side, server-side and database applications that interact with each other via interfaces. Figure 15 depicts how the components are deployed. The client-side applications are deployed either on smartphone or Raspberry Pi devices.

The server-side application, which consists of the components like ApesMobility, AsthmaWatch and e-Security Infrastructure are deployed on application server that can be a single or multiple server depending on subsequent requirements.

The data is deployed to database server and is available to the application server via interfaces. Deployment of the database will be on the client-side, local server and the cloud depending on whether the data is private, semi-private or public respectively. The mapping of the private data to pseudonymous data will be done on the client-side in order to ensure the privacy of the user.

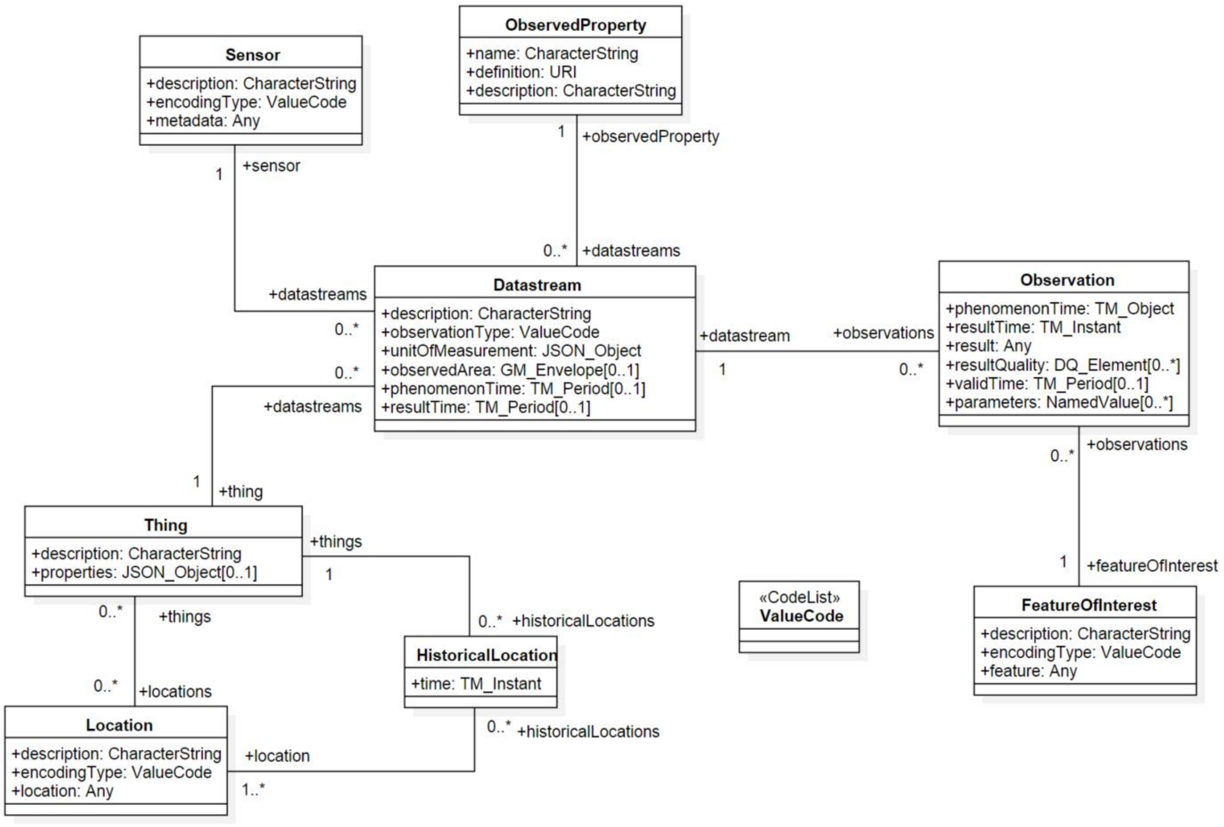
6 Information View

Information is a key enabler for all location-based services and as such, data must be considered the as one of the core assets of the GOEASY platform. Exchanging position information together with sensitive data generated by real people in a secure and privacy-preserving way however, is not a trivial task.

This section provides a focused view on data handled by the GOEASY platform, with a particular attention to high-level, shared models exploited to effectively handle data streams generated from users and sensors . First, an overview of currently adopted data models is presented, helping the reader in framing the context of the section. Then a view on data flow handled by the platform is provided.

6.1.1 OGC SensorThings Data Model

The SensorThings Data Model[2] from the OGC[3] has been selected as the generic representation of data in the GOEASY platform architecture (see Figure 16 for the corresponding data model).



Sensor
+description: CharacterString
+encodingType: ValueCode
+metadata: Any
ObservedProperty
+name: CharacterString
+definition: URI
+description: CharacterString
Datastream
+description: CharacterString
+observationType: ValueCode
+unitOfMeasurement: JSON_Object
+observedArea: GM_Envelope[0..1]
+phenomenonTime: TM_Period[0..1]
+resultTime: TM_Period[0..1]
Observation
+phenomenonTime: TM_Object
+resultTime: TM_Instant
+result: Any
+resultQuality: DQ_Element[0..*]
+validTime: TM_Period[0..1]
+parameters: NamedValue[0..*]
Thing
+description: CharacterString
+properties: JSON_Object[0..1]
HistoricalLocation
+time: TM_Instant
«CodeList» ValueCode
FeatureOfInterest
+description: CharacterString
+encodingType: ValueCode
+feature: Any
Location
+description: CharacterString
+encodingType: ValueCode
+location: Any
+sensor
+observedProperty
+datastreams
+datastream
+observations
+thing
+things
+historicalLocations
+locations
+location
+featureOfInterest

Figure 16. OGC SensorThings Data Model

The OGC SensorThings API data model consists of the Sensing and Tasking profiles.

The Sensing profile allows IoT devices and applications to CREATE, READ, UPDATE, and DELETE (i.e., HTTP POST, GET, PATCH, and DELETE) IoT data and metadata in a *Thing* service. Managing and retrieving observations and metadata from IoT sensor systems is one of the most common use cases. As a

[2] http://ogc-iot.github.io/ogc-iot-api/datamodel.html
[3] http://www.opengeospatial.org/

result, the Sensing profile is designed based on the ISO/OGC Observation and Measurement (O&M) model[4](OGC and ISO 19156:2011[5]).

The key to the model is that an *Observation* is modelled as an act that produces a result whose value is an estimation of a property of the observation target or *FeatureOfInterest*. An Observation instance is classified by its event time (e.g., *resultTime* and *phenomenonTime*), FeatureOfInterest, ObservedProperty, and the procedure used (often corresponding to a *Sensor*). Things are also modeled in the SensorThings API, together with the historical set of their geographical positions

More specifically, in the Sensing profile, a *Thing* has *Location*s and *HistoricalLocation*s. It can also have multiple *Datastream*s associated. A Datastream is a collection of Observations grouped by the same *ObservedProperty* and *Sensor*. An Observation is an event performed by a Sensor that produces a result whose value is an estimate of an ObservedProperty of the FeatureOfInterest.

Following subsections better detail the single data model entries.

Thing

The OGC SensorThings API follows the ITU-T definition, i.e., with regard to the Internet of Things, a thing is an object of the physical world (physical things) or the information world (virtual things) that is capable of being identified and integrated into communication networks (ITU-T Y.2060])

Location

The Location entity locates the Thing or the Things it is associated with. A Thing's Location entity is defined as the last known location of the Thing.

HistoricalLocation

A Thing's HistoricalLocation entity set provides the current (i.e. last known) and previous locations of the Thing with their time.

Datastream

A Datastream groups a collection of Observations and the Observations in a Datastream measure the same ObservedProperty and are produced by the same Sensor

Sensor

A Sensor is an instrument that observes a property or phenomenon with the goal of producing an estimate of the value of the property.

ObservedProperty

An ObservedProperty specifies the phenomenon of an Observation.

Observation

An Observation is an act of measuring or otherwise determining the value of a property (OGC and ISO 19156:2011).

FeatureOfInterest

An Observation results in a value being assigned to a phenomenon. The phenomenon is a property of a feature, the latter being the FeatureOfInterest of the Observation (OGC and ISO 19156:2001). In the context of the Internet of Things, many Observations' FeatureOfInterest can be the Location of the Thing. For

[4] http://www.opengeospatial.org/standards/om
[5] https://www.iso.org/standard/32574.html

example, the FeatureOfInterest of a wifi-connect thermostat can be the Location of the thermostat (i.e. the living room where the thermostat is located in). In the case of remote sensing, the FeatureOfInterest can be the geographical

6.2 Data flow in GOEASY

GOEASY Platform mainly addresses two categories of data streams: data originated by the GOEASY trusted devices, both at the monitoring/sensing level and at the "management" level, and data requested or injected by third parties exploiting the GOEASY platform (i.e. federation services). The two categories have quite different peculiarities and follow distinct, but crossing, paths inside the platform. For now, this document will focus on the first category: data generated from users using GOEASY trusted devices.

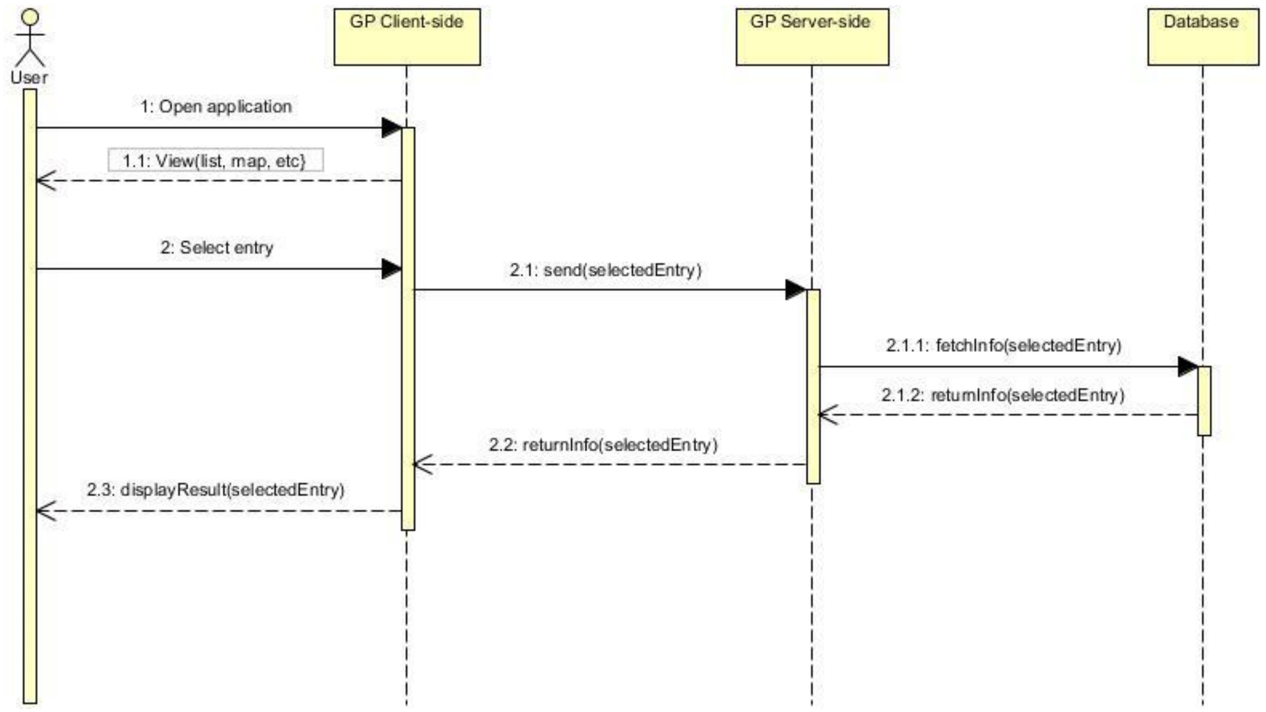

User
GP Client-side
GP Server-side
Database
1: Open application
1.1: View(list, map, etc)
2: Select entry
2.1: send(selectedEntry)
2.1.1: fetchInfo(selectedEntry)
2.1.2: returnInfo(selectedEntry)
2.2: returnInfo(selectedEntry)
2.3: displayResult(selectedEntry)

Figure 17 GOEASY Information Flow

Figure 17 shows the conceptual level information flow for GOEASY platform. In this platform, the client-side application is initiated by the user (when using the smartphone) or automatically (when using Raspberry Pi client), though this figure only depicts the flow for the former. When the user opens an application on the smartphone, the application provides an interface that enables the user to select the views into which the data is entered.

The data entered is sent to the backend in order to be used as criteria to fetch related information from the database and returned back to the user.

7 GOEASY Pilots Technical Use-case Instantiation

7.1 Overview of the GOEASY Pilots

The GOEASY project will consider 2 scenarios as stated in the DoA, which are on the field of sustainable mobility (ApesMobility) and e-health (AsthmaWatch). In the case of ApesMobility scenario (Figure 18), citizens using GNSS-enabled devices are invited to join a game, where their sustainable mobility behaviours will be automatically detected and evaluated, allowing them to get rewards.

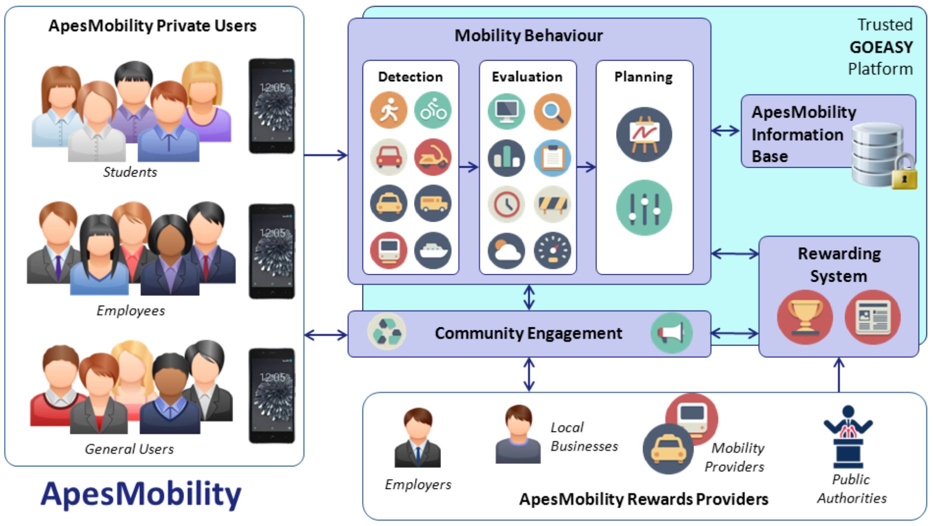


Figure 18 - The ApesMobility Scenario

In this case, the GOEASY project will build the ApesMobility upon the existing GREENAPES platform, which will be enhanced through:

- Integration of new functions for tracking and validation of citizens' mobility choices (i.e. to design and test a tracking of pilot-users through speed and positioning);
- Features to allow mobility statistics reporting on the user end;
- Gamification techniques to engage pilot-users in connection with the existing GREENAPES community in the city of Torino

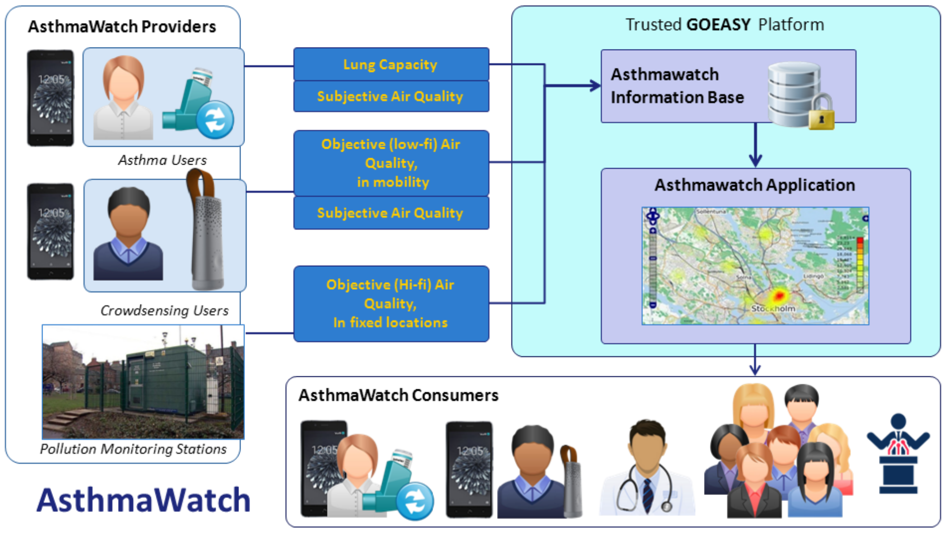


Figure 19 - The AsthmaWatch Scenario

In the AsthmaWatch scenario instead, the focus is on creating a mobile pollution sensing approach using the high precision and anti-spoofing system provided by Galileo and GOEASY, to generate a dynamic "air pollution map" based on the sensor values and the precise GNSS-based position (Figure 19).

In the next paragraphs an overview of two use case instantiations related to the 2 use pilots considered by the GOEASY project will be given to provide a clear idea of how the platform would work in each one of these scenarios.

7.2 Technical Use case Instantiation

Instantiation of the GOEASY platform is when the platform is used in specific use case, e.g., when the platform is used to detect the air quality in a certain environment. The GOEASY client-side application running on Raspberry Pi collects information about a certain location, i.e., location of the Raspberry Pi. The client application sends id of the device, position and air pollution to the GOEASY platform's server-side application.

The server-side application then performs computation on the input data, e.g., conversion, aggregation, etc. The resulting information such as health parameters, values used for input to rules that provide suggestions or heatmap of data points in the location of user's interest. This information is relevant for decision making. For example, whether the user has to go to that location to perform exercise.

7.2.1 Example 1: Data collection with sensor

The sensor system (Raspberry Pi App), continuously collects air pollution data linked to the location where it is collected. This data is transmitted (in the interval of time frame not yet specified) and anonymously stored in the GOEASY platform. Figure 20shows the instantiation of the data collection as a sequence of operations.

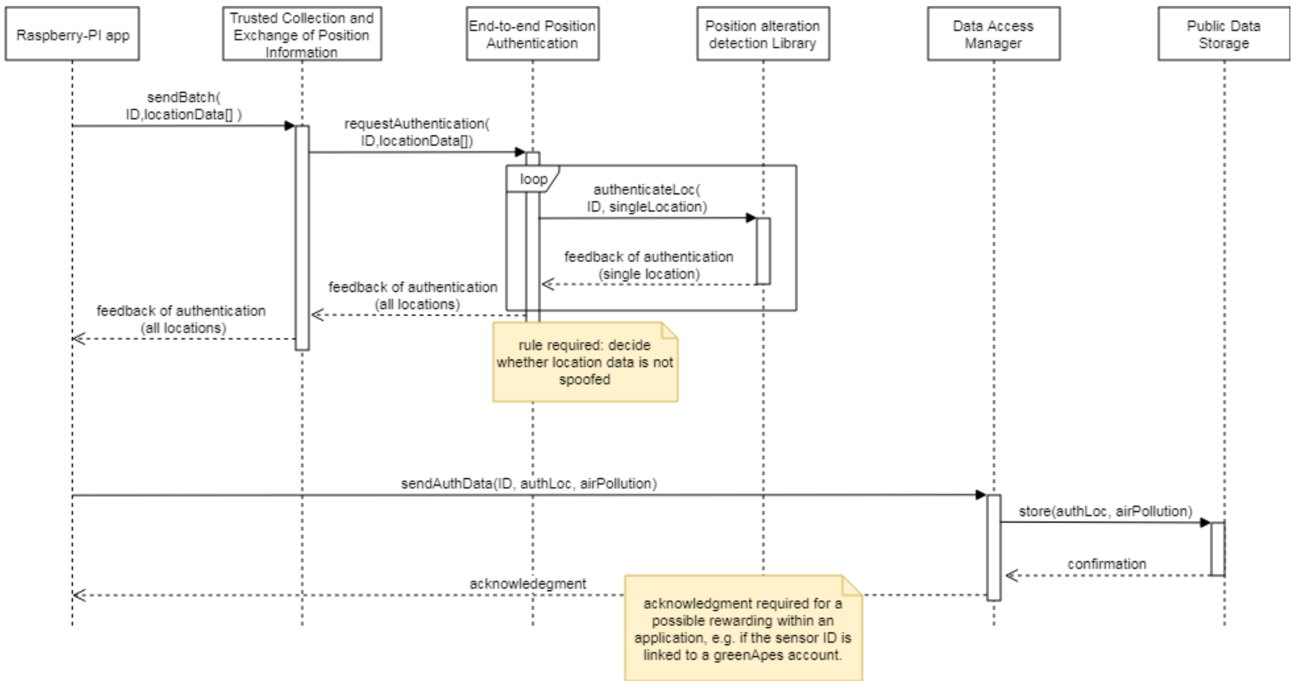

Raspberry-PI app
Trusted Collection and Exchange of Position Information
End-to-end Position Authentication
Position alteration detection Library
Data Access Manager
Public Data Storage
sendBatch(ID,locationData[])
requestAuthentication(ID,locationData[])
loop
authenticateLoc(ID, singleLocation)
feedback of authentication (single location)
feedback of authentication (all locations)
feedback of authentication (all locations)
rule required: decide whether location data is not spoofed
sendAuthData(ID, authLoc, airPollution)
store(authLoc, airPollution)
confirmation
acknowledgement
acknowledgment required for a possible rewarding within an application, e.g. if the sensor ID is linked to a greenApes account.

Figure 20 Use case instantiation of collecting data with Raspberry Pi sensor

7.2.2 Example 2: View conditions on map

Users of AsthmaWatch want to be able to view a map that highlights different levels of e.g. air pollution. The GOEASY platform models the appropriate conditions based on collected and anonymized sensor data and data from third party storages. Figure 21 shows the instantiation of this use case in a sequence diagram.

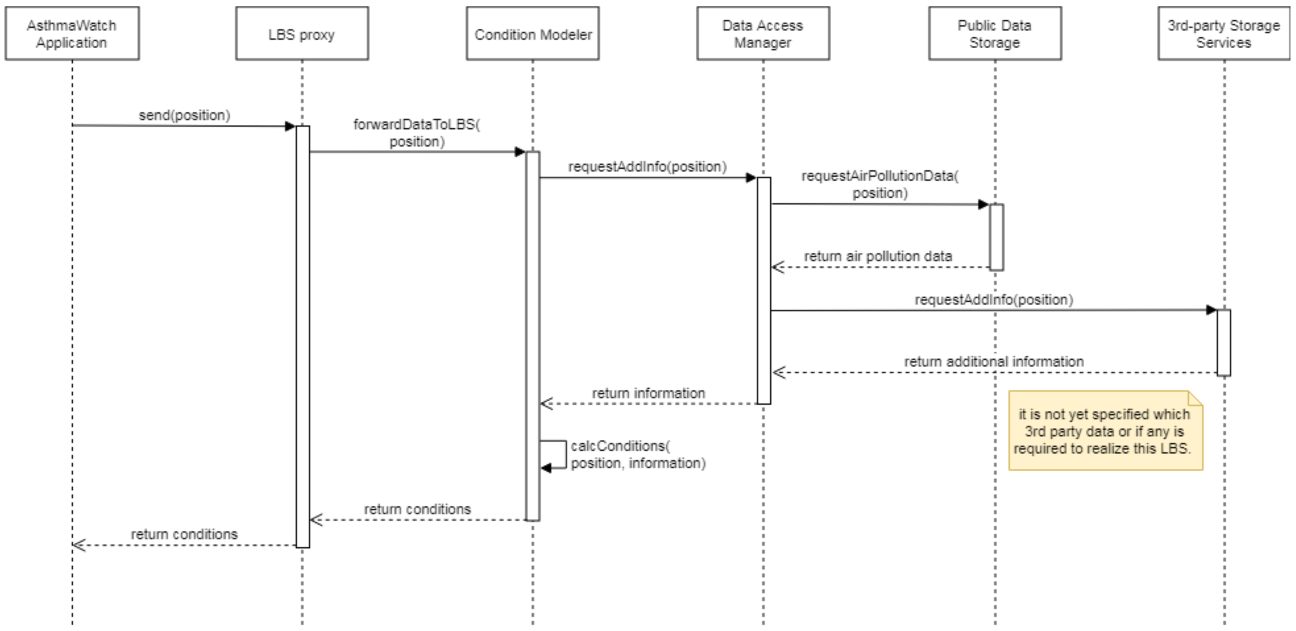

AsthmaWatch Application
LBS proxy
Condition Modeler
Data Access Manager
Public Data Storage
3rd-party Storage Services
send(position)
forwardDataToLBS(position)
requestAddInfo(position)
requestAirPollutionData(position)
return air pollution data
requestAddInfo(position)
return additional information
return information
it is not yet specified which 3rd party data or if any is required to realize this LBS.
calcConditions(position, information)
return conditions
return conditions

Figure 21 Use case instantiation of UC-AW-02: View conditions on map

8 Security Perspective

8.1 Overview

In the GOEASY platform architecture we distinguish between two areas: the trusted area and the untrusted area. The trusted area is demarcated by the boundaries of the platform, and can be accessed through points, as shown in Figure 22.

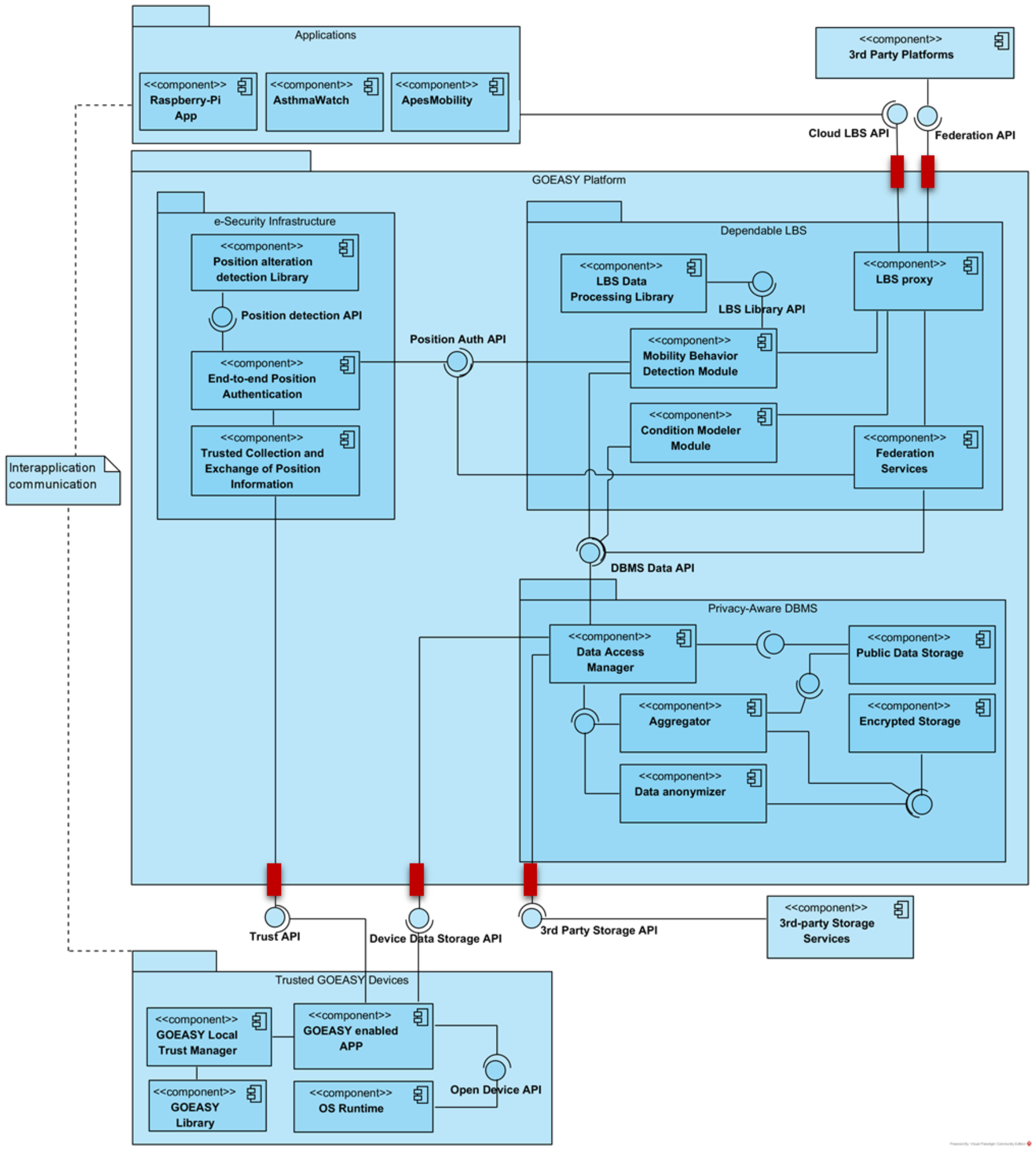

Applications
<<component>> Raspberry-Pi App
<<component>> AsthmaWatch
<<component>> ApesMobility
<<component>> 3rd Party Platforms
Cloud LBS API
Federation API
GOEASY Platform
e-Security Infrastructure
<<component>> Position alteration detection Library
Position detection API
<<component>> End-to-end Position Authentication
<<component>> Trusted Collection and Exchange of Position Information
Position Auth API
Dependable LBS
<<component>> LBS Data Processing Library
LBS Library API
<<component>> LBS proxy
<<component>> Mobility Behavior Detection Module
<<component>> Condition Modeler Module
<<component>> Federation Services
Interapplication communication
DBMS Data API
Privacy-Aware DBMS
<<component>> Data Access Manager
<<component>> Public Data Storage
<<component>> Aggregator
<<component>> Encrypted Storage
<<component>> Data anonymizer
Trust API
Device Data Storage API
3rd Party Storage API
<<component>> 3rd-party Storage Services
Trusted GOEASY Devices
<<component>> GOEASY Local Trust Manager
<<component>> GOEASY enabled APP
<<component>> GOEASY Library
<<component>> OS Runtime
Open Device API

Figure 22. Access points of the GOEASY platform trusted area

The untrusted area comprises external networks, e.g. Federated Networks such as CAPs, IoT platforms and Smart City platforms and 3rd party services. The components that act as a link between the untrusted area and the trusted area are the *DBMS aggregator*, the *Trusted collection and exchange of Position information* module and the *LBS proxy*. The DBMS aggregator connects the trusted area to the trusted GOEASY Devices and 3rd party storage services, together with the *Trusted collection and exchange of Position information* module that enables end-to-end position authentication and trusted exchange of position information features. The LBS proxy instead handles external accesses to the platform from Federated platforms and Cloud-based applications.

Each one of these entrances to the GOEASY platform is completely secure thanks to the Security Framework that provides Authentication and Authorization services (AAS) as it will explained in the following paragraphs.

8.2 Architecture design of the GOEASY Security Framework

The Security Framework goal is to secure components of the GOEASY platform that need to interact with the “untrusted” area, outside the platform or the trusted devices.

To serve this purpose, the Security Framework manages platform users’ profiles and decides what level of access is granted. In addition, the Security Framework protects the user’s information and guarantees that no one can impersonate a GOEASY platform user. It also allows developers and operators to create a role-based access control system, which can be adjusted to specific application requirements.

From a technical standpoint, the Security Framework is mainly composed of 2 main components for handling: i) access management and enforcing authentication, and ii) federation rules and agreements.

These components are referred to as the Access Manager and Identity Manager of the GOEASY Authentication and Authorization services. In order to access GOEASY services (i.e. Dependable LBS, Public/encrypted Storage services, etc.), users must be registered and authenticated prior to any platform interaction. Core elements of the AAS are complemented by Security Enforcement Points placed at the platform boundaries (i.e. in each of the points providing external access to the GOEASY platform as shown in Figure 22), according to the XACML and OpenID Connect reference architectures.

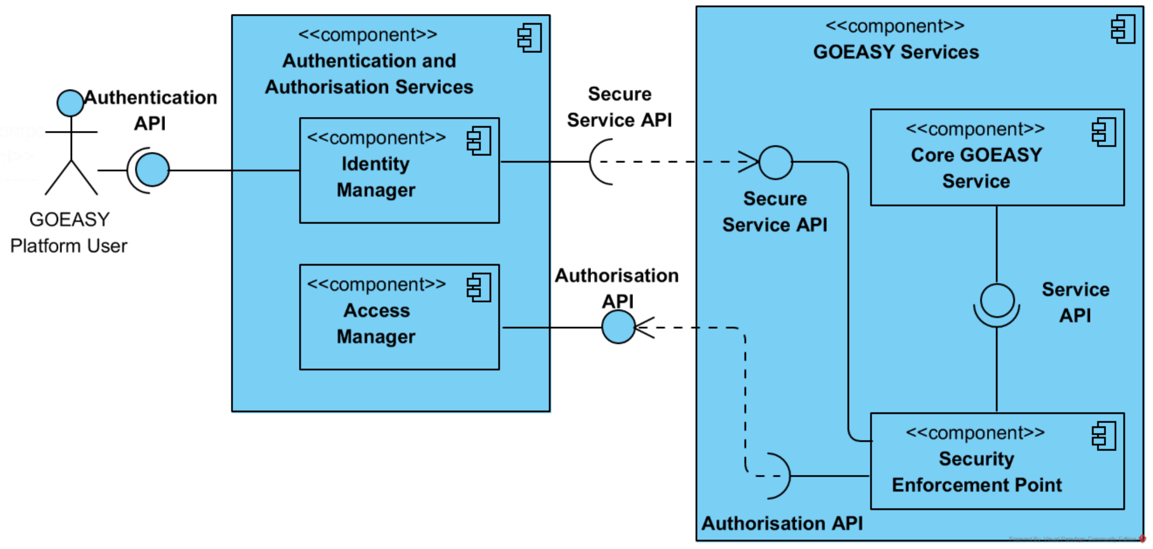

Authentication API
GOEASY Platform User
<<component>> Authentication and Authorisation Services
<<component>> Identity Manager
<<component>> Access Manager
Secure Service API
Authorisation API
<<component>> GOEASY Services
<<component>> Core GOEASY Service
Secure Service API
Service API
<<component>> Security Enforcement Point
Authorisation API

Figure 23. GOEASY Security Framework

Figure 23 shows the initial architecture of the GOEASY Security Framework consisting of the three main elements:

- Identity Manager (IM)
- Access Manager (AM)
- Security Enforcement Point (SEP)

8.2.1 Identity Manager

The GOEASY Identity Manager (IM) has mainly two purposes:

- Register the users of the GOEASY platform
- Provide a sign-in mechanism for federation users, based on OpenID Connect and/or OAuth 2.0.

8.2.2 Access Manager

The GOEASY Access Manager (AM) provides the following features

- Role-based access control, ensuring that only authorized users can access certain data, based on their role and, the properties of their intended operation.
- Centralized policy management to reduce the complexity associated with managing authorization policies separately across multiple platform instances and internally across platform components.

The AM operates following the XACML standard.

8.2.3 Security Enforcement Point

The Security Enforcement Point (SEP) is responsible to enforce security requirements of the components that need to exchange information with the untrusted area.

The SEP focused on enforcing authorization rules, taking into account confidentiality and authentication aspects. The components that will make use of the SEP are:

- GOEASY Dependable LBS
- GOEASY Public and Private Storage
- LinkSmart

All three modules share the requirements that:

- they have to protect the information they handle,
- a platform/federation user has to prove his identity before accessing the service, and
- it must be checked if the user has the proper authorization.

9 Scalability Perspective

Scalability and performance are of most importance in many ICT systems, no matter if they are used in monolithic applications or in applications for fully-distributed systems. While the GOEASY architecture might appear as a relatively vertical solution with few or no scalability issues, the project is adopting a general "design for scalability" approach.

Scalability considerations are being made for the selection of technologies supporting the GOEASY platform, preferring standardized protocols, and considering aspects that influence the scalability of IoT-based cloud solution such as:

- Management and processing of a huge number of incoming message and/or observations. This often causes problems for the back-end processes to elaborate and store the data. Event Hubs and message brokers like RabbitMQ and ActiveMQ are often used to divide data into different worker queues allowing back-end processes to go through the data at their own speed. The normal approach is horizontal scaling where more brokers and back-end processes can be added if needed.

- Storage of the huge amount of sensor data. There are a number of Big data technologies like NoSQL approaches such as document databases and column stores, as well as NoDB approaches where data is not formatted until it is loaded for handling of incoming IoT data streams with high resolution in cloud environments. For scaling the actual data processing and analytics the state-of-the-art approach is to use Map-Reduce systems where the problem is broken down into smaller sub problems processed by individual nodes. There are many applications where a centralised cloud solution will not be able to meet performance requirements. Therefore, new architectures like fog computing and edge analytics could be implemented, where part of the processing is done on the "edge" of the cloud gateways. By pre-processing data on the sensing nodes before sending it to the cloud good performance can be achieved.

Furthermore, to be more scalable, the GOEASY platform will adopt existing federation techniques that have been developed by ALMANAC in the area of Smart City / IoT platforms federation, extending the scope of 3rd party platforms to provide interoperability with CAPs and other platforms providing location-based services.

10 References

[1] IEEE, *ISO/IEC/IEEE 42010:2011, Systems and software engineering — Architecture description,* 2011.

[2] N. Rozanski and E. Woods, Software Systems Architecture: Working With Stakeholders Using Viewpoints and Perspectives, Addison-Wesley Professional, 2011.

[3] E. Commission, "REGULATION (EU) 2016/679 OF THE EUROPEAN PARLIAMENT AND OF THE COUNCIL of 27 April 2016on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing Directive 95/46/EC (General Dat," April 2016. [Online]. Available: https://eur-lex.europa.eu/legal-content/IT/TXT/?uri=CELEX%3A32016R0679.

[4] A. Hern, "The Guardian," 28th January 2018. [Online]. Available: https://www.theguardian.com/world/2018/jan/28/fitness-tracking-app-gives-away-location-of-secret-us-army-bases.

[5] Raw Mwasurements Task force, "Using Gnss Raw Measurements," GSA, Prague, 2017.

Acronyms

Acronym	Explanation
DoA	Description of Action
GEP	GOEASY platform
LBS	Location-Based Services
API	Application Programming Interface
IOT	Internet of Things
AD	Architectural Description
CAP	Collective Awareness Platform
GNSS	Global Navigation Satellite System
DBMS	Database Management System
GDPR	General Data Protection Regulation
DPO	Data Protection Officer
DPA	Data Protection Authority
PII	Personal Identifiable Information
OTP	Event-based One-Time Password
TOTP	Time-based One-Time Password
TDS	Transaction Data Signing
XMPP	Extensible Messaging and Presence Protocol
SASL	Simple Authentication and Security Layer
TSL	Transport Layer Security
OS	Operating System
OGC	Open Geospatial Consortium
AAS	Authentication and Authorization services
IM	Identity Manager
AM	Access Manager
SEP	Security Enforcement Point
XACML	eXtensible Access Control Markup Language

List of figures

List of tables

Appendix A: Application requirements – ApesMobility

Key	Summary	Status	Labels
GOEAS-65	As a citizen, I want to be able to delete identifiable location data to protect my privacy.	OPEN	ApesMobility UC-APES-08
GOEAS-64	As a citizen, I want to be able to enable and disable location tracking on a regular basic automatically to protect my privacy.	OPEN	ApesMobility UC-APES-08
GOEAS-63	As a citizen, I want to be able to enable location tracking only for a specific time frame to protect my privacy.	OPEN	ApesMobility UC-APES-08
GOEAS-62	As a citizen, I want to be able to en-/disable location tracking manually to protect my privacy.	OPEN	ApesMobility UC-APES-08
GOEAS-61	As a citizen, I want to subscribe to specific areas or topics only in order to get relevant information only.	OPEN	ApesMobility UC-APES-07
GOEAS-60	As a citizen, I want to be notified about new challenges in order to improve my sustainable behavior.	OPEN	ApesMobility UC-APES-06 UC-APES-07
GOEAS-59	As a citizen, I want to see additional information about registered locations/ events in order to decide whether to go there or not.	OPEN	ApesMobility UC-APES-06 UC-APES-07
GOEAS-58	As a citizen, I want to be able to find registered POIs and challenges in order to collect additional rewards by check-in (and share).	OPEN	ApesMobility UC-APES-06 UC-APES-07
GOEAS-57	As a citizen, I want to be notified about a detected, certified activity in order to not miss additional rewards.	OPEN	ApesMobility UC-APES-01
GOEAS-55	As a citizen, I want to be able to share my certified activities in order to motivate fellow citizens.	OPEN	ApesMobility UC-APES-01
GOEAS-20	As a citizen, I want to have full control of the location tracking in order to protect my privacy.	OPEN	ApesMobility UC-APES-01 UC-APES-08
GOEAS-15	As a citizen, I want to join challenges to get motivated and to support the community.	OPEN	ApesMobility UC-APES-07
GOEAS-14	As an organization, I want to offer challenges for users in order to familiarize them with sustainable topics.	OPEN	ApesMobility UC-APES-06
GOEAS-13	As a citizen, I want to get my location certified to get additional rewards.	OPEN	ApesMobility

Key	Summary	Status	Labels
			UC-APES-05
GOEAS-12	As a citizen, I want to share my location to inspire others and/or be additionally rewarded.	CLOSED-DUPLICATED	ApesMobility UC-APES-04
GOEAS-11	As a citizen, I want to join activities shared by others to get inspired for sustainable behavior.	CLOSED OUTOFSCOPE	ApesMobility UC-APES-03
GOEAS-10	As a citizen, I want to get my activity certified to get additional rewards.	OPEN	ApesMobility UC-APES-02
GOEAS-9	As a citizen, I want to be able to confirm my certified activity to be additionally rewarded.	OPEN	ApesMobility UC-APES-01

Appendix B: Application requirements – AsthmaWatch

Key	Summary	Status	Labels
GOEAS-16	As a user I want to know about the condition in an area to decide if I will go there or not.	OPEN	AsthmaWatch UC-AW-01 UC-AW-02
GOEAS-17	As a user I want to know the alternative routes based on a specific criteria in order to avoid an asthma attack.	OPEN	AsthmaWatch UC-AW-03 UC-AW-04 UC-AW-05 UC-AW-06
GOEAS-18	As a user I want to keep track of my health in order to react in time on changes.	OPEN	AsthmaWatch UC-AW-07 UC-AW-08
GOEAS-19	As a user I want to inform specific persons about an asthma attack in order to get immediate help.	OPEN	AsthmaWatch UC-AW-09 UC-AW-10 UC-AW-11
GOEAS-21	As a user I want to define rules for a specific criteria in order to be warned in relevant situations.	OPEN	AsthmaWatch UC-AW-01
GOEAS-22	As a user I want to be notified when I am within a certain range of a dangerous spot.	OPEN	AsthmaWatch UC-AW-01

Key	Summary	Status	Labels
			UC-AW-02
GOEAS-23	As a user I want to be able to view information of a single measuring station to decide if I will go to this area or not.	OPEN	AsthmaWatch UC-AW-02
GOEAS-24	As a user I want to be able to define a preferred route in order to get advised whether to take it or not.	OPEN	AsthmaWatch UC-AW-03
GOEAS-25	As a user I want to be able to import a route from 3rd party app(s) in order to reduce redundant tasks (defining the same route in different applications).	OPEN	AsthmaWatch UC-AW-03
GOEAS-26	As a user I want to get alternative routes suggested automatically if conditions along my preferred route exceed my rules in order to avoid an asthma attack.	OPEN	AsthmaWatch UC-AW-05 UC-AW-06
GOEAS-27	As a user I want to be able to view details of any suggested route in order to avoid an asthma attack.	OPEN	AsthmaWatch UC-AW-06
GOEAS-28	As a user I want to be able to export a picked route in order to start navigation with a 3rd party app.	OPEN	AsthmaWatch UC-AW-06
GOEAS-29	As a user I want to be able to enter health information manually to keep track of my health.	OPEN	AsthmaWatch UC-AW-07
GOEAS-30	As a user I want to be able to add health information automatically by a connected device to keep track of my health.	OPEN	AsthmaWatch UC-AW-07
GOEAS-31	As a user I want to be able to enter health information to the application in order to keep track of my health.	OPEN	AsthmaWatch UC-AW-07
GOEAS-32	As a user I want to get immediate feedback after adding health information in order to react in time on changes.	OPEN	AsthmaWatch UC-AW-07
GOEAS-33	As a user I want to be able to view historical data in order to know about the development of my health.	OPEN	AsthmaWatch UC-AW-08
GOEAS-35	As a user I want to control the map by known interaction patterns to get the required information as fast and easy as possible.	OPEN	AsthmaWatch UC-AW-02 UC-AW-03 UC-AW-06
GOEAS-36	As a user I want to be able to specify multiple persons as emergency contacts to increase the probability to get help.	OPEN	AsthmaWatch UC-AW-09
GOEAS-37	As a user I want to be able to activate an alarm in case of an asthma attack in order to get immediate help.	OPEN	AsthmaWatch UC-AW-10

Key	Summary	Status	Labels
GOEAS-38	As a user I want to inform my emergency contacts about my position when I activated the alarm to get help as fast as possible.	OPEN	AsthmaWatch UC-AW-10 UC-AW-11
GOEAS-39	As a user I want my phone to play an alerting sound when I activated the alarm to get immediate help by people around me.	OPEN	AsthmaWatch UC-AW-10
GOEAS-40	As a user I want to view all outgoing calls in the app due to an activated alarm to know who was informed and who I should update about my situation now.	OPEN	AsthmaWatch UC-AW-10 UC-AW-11
GOEAS-41	As a user I want to be bale to cancel a call at any time to avoid unnecessary attempts of contacts.	OPEN	AsthmaWatch UC-AW-10 UC-AW-11
GOEAS-42	As a user I want to be able to select text messages to send to all emergency contacts in case I cancelled a call in order to let them know about my current situation.	OPEN	AsthmaWatch UC-AW-10 UC-AW-11
GOEAS-43	As a user I want to be able to create and update templates for text messages to inform people faster about my current situation.	OPEN	AsthmaWatch UC-AW-10 UC-AW-11
GOEAS-44	As a user I want to be able to determine the order in which contacts are contacted to prioritize based on criteria.	OPEN	AsthmaWatch UC-AW-09