



# CommuniCity

Innovative Solutions Responding to  
the Needs of Cities & Communities

## **D4.3 Pilot support and technical validation report**



Funded by  
the European Union



## D4.3 Pilot support and technical validation report

Project acronym: **CommuniCity**

Project full title: **Innovative Solutions Responding to the Needs of Cities & Communities - CommuniCity**

Grant agreement no.: 101070325

<b>Due Date:</b>	06/06/2024
<b>Lead Partner:</b>	ENG
<b>Editor:</b>	Martino Maggio, Giuseppe Ciulla (ENG)
<b>Reviewers:</b>	Anna Björk (DH) Coen Antens (CVC)
<b>Dissemination Level:</b>	PU
<b>Version:</b>	V1.1



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

## DOCUMENT INFO

Date and version number	Author	Comments
24/07/2023	Martino Maggio (ENG)	Table of Content
04/08/2023	Edgar Gracia Larrosa (CVC), Martino Maggio (ENG)	First contribution on Toolbox section, First version of chapter 1
11/08/2023	João Bastos (PD), Gonçalo Fonseca (PD),	First draft of chapter 3 and section about sandbox
08/09/2023	Martino Maggio (ENG), Edgar Gracia Larrosa (CVC),	Second version of Chapter 1 and 2
22/09/2023	João Bastos (PD), Martino Maggio (ENG)	Improvement of chapter 3, Final draft for internal review
27/09/2023	Anna Björk (DH), Coen Antens (CVC)	Reviewed version
29/09/2023	Martino Maggio (ENG)	Final version
06/06/2024 v1.1	Adeeb Sidani(PD), Giuseppe Ciulla(ENG), Coen Antens (CVC)	Processing review remarks in V1.1

## GLOSSARY

<b>APIs</b>	Application Programming Interfaces
<b>AI</b>	Artificial Intelligence
<b>DTDl</b>	Digital Twin Definition Language
<b>ETSI</b>	European Telecommunications Standards Institute
<b>GE</b>	Generic Enabler
<b>OASC</b>	Open & Agile Smart Cities
<b>SAREF</b>	Smart Appliances REference
<b>VM</b>	Virtual Machine

## Table of contents

<b>1. Introduction.....</b>	<b>5</b>
<b>2. Knowledge base and learning resources.....</b>	<b>7</b>
2.1 CommuniCity documents and tutorials	7
2.2 Other relevant resources: OASC MIMs website	8
2.3 Technical webinars	9
<b>3. Technical assets.....</b>	<b>10</b>
3.1 Sandbox: Docker compose pre-defined containers	10
3.2 Toolbox: Online demos	13
3.3 External tools and components	15
3.3.1 FIWARE Catalogue	15
3.3.2 Ontologies and common data models	16
<b>4. Online support system.....</b>	<b>18</b>
4.1 Opening a Ticket	20
<b>5. Conclusions .....</b>	<b>23</b>
<b>6. References .....</b>	<b>24</b>

## Index of Figures

Figure 1 - CommuniCity on-line technical documentation	7
Figure 2: Partial view of the docker-composed file used to instantiate the sandbox. Configuration of the Orion Context Broker.	10
Figure 3: Partial view of the docker-composed file used to instantiate the sandbox. Configuration of the Timescale and Mintaka.	11
Figure 4 - Toolbox architecture	13
Figure 5 - FIWARE Catalogue website	15
Figure 6 -Example of data models (Parking data models) of Smart Data Model initiative	16
Figure 7: Example flowchart of a request for a new sandbox	17
Figure 8: Example flow for the process to change the parameters of the sandbox.	18
Figure 9: Example flow for the process to remove a sandbox.	19
Figure 10 - Support system official mail	20
Figure 11 - Ticketing system login	20
Figure 12 - Ticketing System – queue list	21
Figure 13 - Ticketing system – ticket list	21

# 1. Introduction

This deliverable includes information related to valuable resources and technical support for open call solution developers in compliance with the CommuniCity technical framework including technical assets and specifications as identified in Deliverable 4.1 [1]. This document can be considered a practical guide and the main reference for the participants of the 2<sup>nd</sup> and 3<sup>rd</sup> open calls, providing a complete overview of the resources provided to the developers by CommuniCity in the first-year project. The documentation and assets referenced in this deliverable can be considered the baseline to support the development of interoperable replicable applications following the CommuniCity approach.

This deliverable is mainly focused on the three pillars of the pilot support:

- Chapter 1 describes the knowledge batch and learning resources for developers including CommuniCity technical deliverables, the official online documentation and other useful websites related to the standards and technologies adopted in CommuniCity. The last part of the chapter describes the technical webinars, a set of online events for open call developers and pilot cities to present the CommuniCity technical resources and approaches.
- Chapter 2 provides a general overview of the technical assets available in CommuniCity: the toolbox with some ready to use demos based on AI for image recognition/processing compliant with CommuniCity specifications. The chapter also includes a description of resources accessible through the Sandbox (preconfigured VM to help developers in the validation of their solutions). At the end of the chapter, a set of external technical components and standards compliant with the CommuniCity specification that can be reused as building blocks for the development of open call services are listed.
- Chapter 3 describes the process of acquiring support of the CommuniCity technical team concerning the project's technical framework and tools. In particular, the developers that need help for the usage of a specific tool or, for instance, want to request access to the sandbox, can open via email a ticket that will be managed through a dedicated platform by the project team.

The chapters mentioned above provide an overall description of the different supporting resources. Additional contents are included in the documents and websites referenced in the different sections and listed in the Reference section.

For V1.1, we implemented updates to clarify various aspects of the ethical requirements for AI. Our toolbox incorporates standard algorithms and neural networks derived from extensive computer vision research. These models are publicly available and are used in accordance with their respective licences, such as the MIT License or the Apache License.

Instead of training new models from the ground up, we utilise pre-existing models that have already been developed and validated by the research community. Our modifications focus on adapting these models to ensure compatibility with MIM1 and MIM2, enhancing their interaction within our toolbox.

The deliverable “D4.2 Pilot Toolbox and validation sandbox” provides clear information about which models are used, their source code repositories, and their licensing information. This transparency is

essential for maintaining ethical standards and informing users about the tools they are employing. This dedication to ethical practices and transparency fosters trust among our users and supports the responsible development of AI.

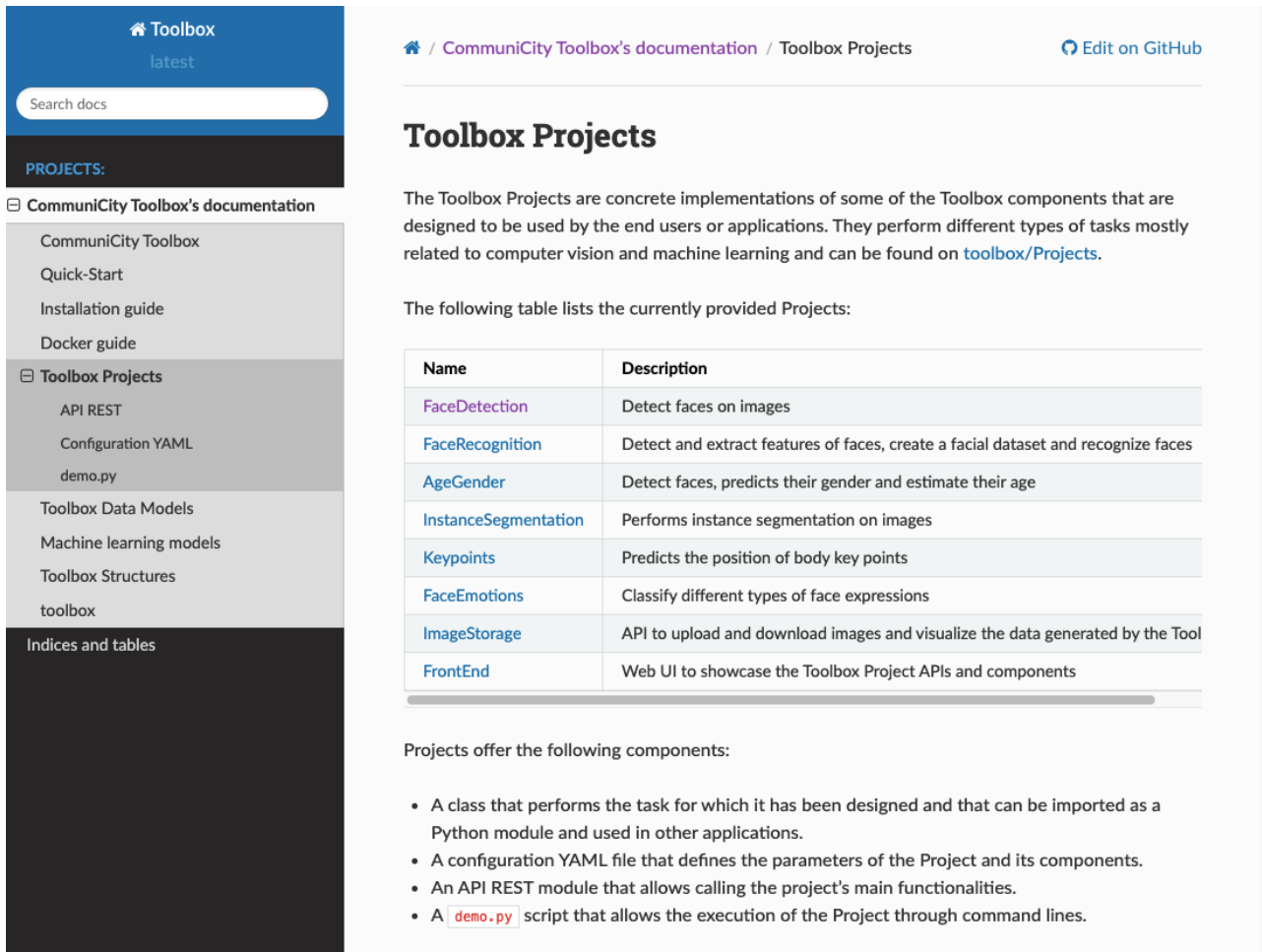
## 2. Knowledge base and learning resources

One of the main resources for the developers and, in general, the stakeholders interested in understanding the approach of the CommuniCity technical framework is the knowledge base that consists of the official documentation of the project and external resources. In this chapter, the most relevant documents are referenced.

### 2.1 CommuniCity documents and tutorials

Technical deliverables: they are the documents produced inside the work package 4 “Technical Common Ground”, in particular:

- **D4.1 - Technical architecture and APIs** [1]: it includes the description of technical specifications and the architectural elements of the CommuniCity project. The scope of this document is to provide high level specifications to the open call developers that will implement innovative solutions for the Cities involved in the project. More specifically, the document introduces the most relevant initiatives that constitute the main technical reference for CommuniCity, in particular, the OASC MIMs universal tools for achieving interoperability of data, systems, and services between cities and suppliers around the world. This deliverable also includes an overview of the CommuniCity project technical framework introducing its main elements: (1) The specification and API layer that represents the logical and functional specifications of the interoperability components of the architecture based on OASC MIMs, (2) the toolbox, a set of technical tools (open software, data models, services etc.) to support the open call developers in building CommuniCity compliant services and (3) the sandbox, a cloud environment, provided to the open call developers, to test the compliance of their services with the CommuniCity specifications. The core of D4.1 is the detailed description of the main specifications and API that will be considered the baseline of the CommuniCity technical framework. The compliance with these specifications will facilitate the interoperability of the open call solutions with third party systems (e.g. City IT systems) and their possible replicability.
- **D4.2 - “Pilot Toolbox and validation sandbox** [2] provides details about the concrete instances of the CommuniCity Toolbox and Sandbox. Regarding the Toolbox, it contains information about the software available, the endpoints, and description of some demo services with examples. Moreover, in this document are included details about the structure of the Sandbox and the way to access it.
- **Online documentation:** to provide a more detailed and up-date technical documentation [3] CommuniCity offers an online Knowledge base managed via GitHub repository. The documentation includes a tutorial for installation and configuration of software components, demo examples and will also describe the specific procedure to access the Sandbox and the online ticketing system. Moreover, it will be provided a dedicated FAQ section to answer to the most relevant technical questions of the open call developers.



The screenshot displays the 'CommuniCity Toolbox's documentation' website. The left sidebar contains a navigation menu with categories like 'CommuniCity Toolbox', 'Quick-Start', 'Installation guide', 'Docker guide', 'Toolbox Projects' (selected), 'Toolbox Data Models', 'Machine learning models', 'Toolbox Structures', 'toolbox', and 'Indices and tables'. The main content area is titled 'Toolbox Projects' and includes a search bar, a breadcrumb trail, and an 'Edit on GitHub' link. The text explains that these projects are concrete implementations of toolbox components. A table lists the following projects:

Name	Description
<a href="#">FaceDetection</a>	Detect faces on images
<a href="#">FaceRecognition</a>	Detect and extract features of faces, create a facial dataset and recognize faces
<a href="#">AgeGender</a>	Detect faces, predicts their gender and estimate their age
<a href="#">InstanceSegmentation</a>	Performs instance segmentation on images
<a href="#">Keypoints</a>	Predicts the position of body key points
<a href="#">FaceEmotions</a>	Classify different types of face expressions
<a href="#">ImageStorage</a>	API to upload and download images and visualize the data generated by the Tool
<a href="#">FrontEnd</a>	Web UI to showcase the Toolbox Project APIs and components

Below the table, it states that projects offer the following components:

- A class that performs the task for which it has been designed and that can be imported as a Python module and used in other applications.
- A configuration YAML file that defines the parameters of the Project and its components.
- An API REST module that allows calling the project's main functionalities.
- A `demo.py` script that allows the execution of the Project through command lines.

Figure 1 - CommuniCity on-line technical documentation

## 2.2 Other relevant resources: OASC MIMs website

**OASC MIMs website** [4]: reports the information related to the MIMs (version 2022), including overall description capabilities, reference to APIs and standards. Considering that some MIMs are not fully defined in terms of specification, this website represents the main reference to follow their evolution. Moreover, it is important to highlight that, during 2023, the way in which the MIMs are described and documented is under revision and in the near future a newer version of them will be released. Either way the technical specifications and standards related to the original MIMs remain valid. More information about this process can be found here [5].

## 2.3 Technical webinars

Another important supporting resource will be the Technical Webinars, online sessions dedicated to the open call developers, cities technical staff and other stakeholders interested in improving knowledge about the CommuniCity technical framework. The webinars will take place at the beginning of the 2<sup>nd</sup> and 3<sup>rd</sup> open call project execution. A specific plan will be defined in the last quarter of 2023 in relation to the specific needs of the open call projects and stakeholders.

The objectives of the technical webinars are the following:

- Presentation of technical framework and approach of the CommuniCity project
- Presentation of the most relevant aspects of technical tools with practical examples and hands-on sessions
- Identification of the most useful resources for the specific use cases of the open call projects
- Discuss the benefits and challenges for the adoption of CommuniCity Technical approach in public administrations
- Answer to questions and collection of feedback about the technical framework and the support process

The topics of the webinars will be different in order to cover the more interesting aspects for the open call project participant. A tentative list of the webinars is reported below:

- **CommuniCity technical framework overview:** this webinar will provide a general presentation of the main elements of the CommuniCity technical framework, including specifications, the Toolbox, the Sandbox and the support tools
- **CommuniCity Toolbox and Sandbox hands-on:** a practical webinar in which will demonstrate some components part of the CommuniCity toolbox and the usage of the sandbox for testing the compliance with CommuniCity specification.
- **MIMs introduction:** a webinar dedicated to the presentation of the OASC MIMs (in particular MIM1-2-3-4-5), their objectives and concrete examples of their adoptions.
- **Context Management and related tech tools:** this webinar will be about the role of context broker in the CommuniCity framework, NGSI-LD standard and concrete examples of its usage (focusing on FIWARE implementations)
- **MIM5 – FAIR and Transparent AI:** during the webinar will be presented the status of to the work in progress to define MIM5 “Fair and Transparent Artificial Intelligence” in order to identify a set of capabilities required to check that the algorithmic systems offered by suppliers to make decisions related to the services offered to the citizens are fair, trustworthy and transparent.

Other possible topics will be identified during the project based on suggestions and feedback coming from the CommuniCity open call participants and project partners.

### 3. Technical assets

In this section, we will describe the technical tools that can be used by the open call developers to simplify the development and testing of services.

#### 3.1 Sandbox: Docker compose pre-defined containers

A sandbox represents a safe environment for pilots to test their solutions and to validate them against the standard data model and components (following the OASC MIMs). The sandbox developed in this context currently implements a [MIM1](#) complaint deployment environment.

Each sandbox is built/implemented as a *virtual machine* with a set of fixed resources and equipped with a **Context Broker** and *TimescaleDB* time-series **historical database**. The Orion Context Broker [6] is MIM1 compliant and implements an ETSI **NGSI-LD** [7] protocol for data exchange.

NGSI-LD is an information model and API to collect, retrieve and subscribe to context information. Together with the use of standard data models such as SmartDataModels, a city can implement a robust system to gather data that is dynamic and relevant to its context and access and share it easily.

Each sandbox is instantiated on demand when pilots request their creation. This is done by submitting a request that is received via a ticketing system. The ability to create these environments on request relies on the use of a predefined recipe for the creation of the virtual machines and MIM1 infrastructure. This is done using Docker as a background engine and a predefined docker-compose file, as shown in Figure 2 and Figure 3. While Figure 2 shows the configuration parameters, in docker specification, for the Orion Context Broker, Figure 3 depicts the configuration parameters for the Timescale and Mintaka components that implement the temporal aspects of the NGSI-LD protocol.

To ensure compliance with data protection and privacy principles, such as data minimization, all data managed within these sandbox environments is handled with strict confidentiality and security measures. Data is used solely for the purpose of testing and validation within the sandbox and is erased once the virtual machine's designated period expires, ensuring that no residual data remains accessible. By adhering to these principles, we demonstrate our commitment to ethical standards and the responsible use of data within our sandbox environments.

```

version: "3.5"
services:
  # Orion is the context broker
  orion:
    image: fiware/orion-ld:1.1.0
    hostname: orion
    container_name: fiware-orion
    restart: always
    environment:
      - ORIONLD_TROE=TRUE
      - ORIONLD_TROE_USER=orion
      - ORIONLD_TROE_PWD=####
      - ORIONLD_TROE_HOST=timescale
      - ORIONLD_MONGO_HOST=mongo-db
    depends_on:
      - mongo-db
      - timescale
    networks:
      - default
    ports:
      - "1026:1026"
    command: -logLevel DEBUG
    healthcheck:
      test: curl --fail -s http://orion:1026/version || exit 1
      interval: 5s

  # Databases
  mongo-db:
    image: mongo:4.0
    hostname: mongo-db
    container_name: db-mongo
    expose:
      - "27017"
    ports:
      - "27017:27017" # localhost:27017
    networks:
      - default
    command: --nojournal
    volumes:
      - mongo-db:/data
    healthcheck:
      test: |
        host=`hostname --ip-address || echo '127.0.0.1'`;
        mongo --quiet $host/test --eval 'quit(db.runCommand({ ping: 1 }).ok ? 0 : 2)' && echo 0 || echo 1
      interval: 5s

```

**Figure 2: Partial view of the docker-composed file used to instantiate the sandbox. Configuration of the Orion Context Broker.**

```
timescale:
  image: timescale/timescaledb-postgis:1.7.5-pg12
  hostname: timescale
  container_name: timescale
  healthcheck:
    test: [ "CMD-SHELL", "pg_isready -U orion" ]
    interval: 15s
    timeout: 15s
    retries: 5
    start_period: 60s
  environment:
    - POSTGRES_USER=orion
    - POSTGRES_PASSWORD=orion
    - POSTGRES_HOST_AUTH_METHOD=trust
  expose:
    - "5432"
  ports:
    - "5432:5432"
  networks:
    - default

mintaka:
  image: fiware/mintaka:0.4.3
  hostname: mintaka
  restart: always
  container_name: mintaka
  environment:
    - DATASOURCES_DEFAULT_HOST=timescale
    - DATASOURCES_DEFAULT_USERNAME=orion
    - DATASOURCES_DEFAULT_PASSWORD=orion
    - DATASOURCES_DEFAULT_DATABASE=orion
  expose:
    - "8080"
  ports:
    - "8080:8080"
  networks:
    - default

volumes:
  mongo-db: ~
```

*Figure 3: Partial view of the docker-composed file used to instantiate the sandbox. Configuration of the Timescale and Mintaka.*

## 3.2 Toolbox: Online demos

The goal of work package 4 is to provide a set of tools but, more importantly, an example of a paradigm on how to incorporate these tools in applications that are compliant with the MIMs.

In order to demonstrate the use of the toolbox on a more practical level, we chose to implement a number of demos using this paradigm of the MIMs. We chose to implement demos that are based on Artificial Intelligence to showcase some examples of what can be done by using these technologies. The paradigm can be used for any application.

In the end, the toolbox is made available on GitHub together with extensive information that can be found on ReadTheDocs. In order to get a feeling for the toolbox, we chose to make it available as a service as well. In this way, the users can access the demos from the browser without installing anything.

The toolbox itself is provided as a **Git** repository available at [8]. This approach is meant for developers who prefer to have full control over their applications. They have direct access to the library to incorporate the functions directly into their applications. For developers that first want to try the library, the toolbox is also available in the form of a **Sandbox** that has the toolbox components (pre)installed so that developers can start developing right away. Finally, the toolbox is also implemented as a **service**, with multiple APIs and a front-end to interact with the toolbox components, available at [9]

The initial version of the toolbox includes the following modules:

- **Models:** Contains a collection of computer-vision machine-learning models.
- **Structures:** Defines a set of data structures for representing the machine learning models' output.
- **Context:** Contains a set of context data management utilities.
- **Data Models:** Contains the defined data models used by the toolbox.
- **Projects:** Contains the implementation of different projects that are the final applications or demos that the users will use.
- **Visualisation** : Contains a set of modules for the visualisation of the data models.
- **Tools:** Contains some command-line tools.
- **Docs:** Contains the documentation of the toolbox and some Python Notebook tutorials.

Each one of the toolbox projects implements an API service that exposes all its functionalities and gives remote access to clients and applications. These APIs process input images and return a data model as a response. The returned data models are also posted in a context broker, so that any other application can use this data. The APIs can also receive notifications from the context broker and run some tasks when certain conditions are met. They are based on the FastAPI Python framework that follows the open standards for APIs OpenAPI (Swagger) and JSON Schema. It also offers auto generated and interactive documentation and clear error handling.

In addition to the computer vision demos, two more services are offered:

- An Image Storage API that allows the upload and download of images. It can be used to temporarily store images that will be available to all the toolbox components and to retrieve generated images with the models' output data visualized.
- A front-end web page that allows users to interact with and try the toolbox components without any previous technical knowledge.

The toolbox components can run in a distributed way with a consumer-producer pattern, using a Context Broker to share and publish data. Some components can be set up to produce data to the context broker, while other components can create subscriptions to the context broker and process this data as soon as it is posted.

As an example, we could have a component extracting facial features from a camera at the entrance of a building and uploading this data to a context broker. Then a face recognition component can be set up somewhere else to recognise the faces using this data, also posting its output to the Context Broker and triggering other applications based on who is recognised.

These components can be launched isolated from each other using Docker. Docker Compose can also be used to orchestrate the deployment of a set of components. Each one will serve its own API on a different port.

The current version of the toolbox includes the following demos:

- Face detection
- Face recognition
- Face emotions classification
- Age and gender estimation
- Instance segmentation
- Body key-points detection

More implementations will be included in future iterations of the toolbox.

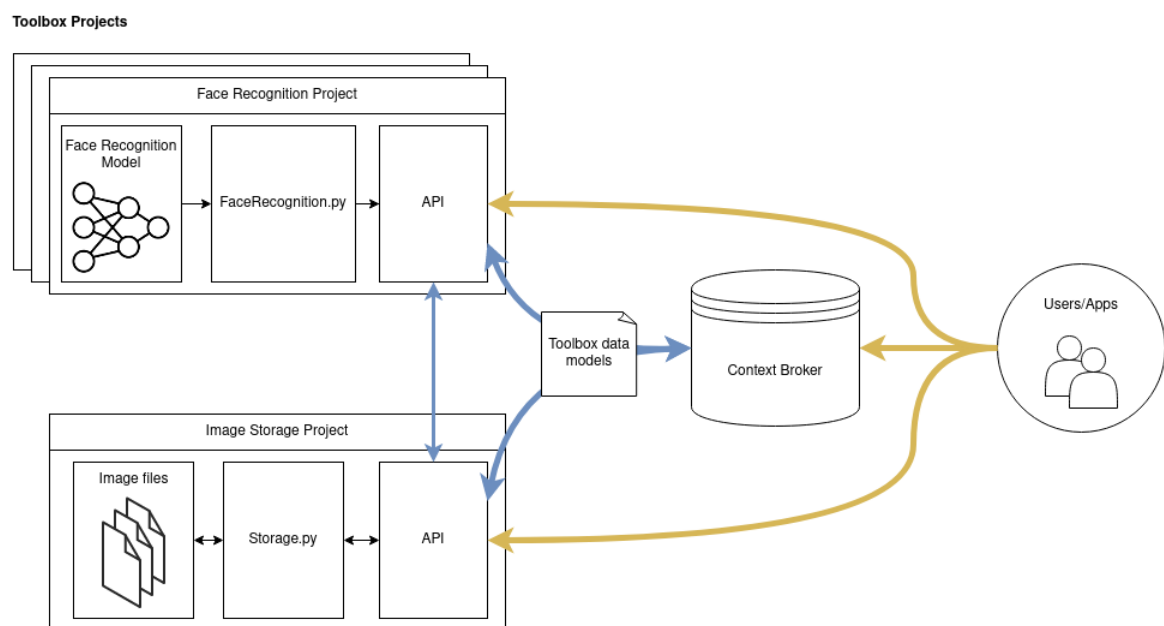


Figure 4 - Toolbox architecture

### 3.3 External tools and components

Open call developers can also benefit from external tools and components that are compliant with MIMs and, in general, standards related to the CommuniCity specification – some of them are referenced in this section.

#### 3.3.1 FIWARE Catalogue

- FIWARE Catalogue [10] is a curated Open-Source framework of software platform components using a common API based on Context Data Management, managed by FIWARE Foundation. The FIWARE Catalog is an important resource for the developers because it includes several general-purpose open-source components that can be used to build smart applications. The components are classified based on their capabilities and role in the FIWARE logical architecture. The main component categories (“chapters”) are:
- **Core Context Management:** this chapter includes “Context Brokers” and “data connectors”. Context Broker component is the core and mandatory component of any FIWARE. It enables the management of context information in a highly decentralised and large-scale manner.
- **Context Processing, Analysis and Visualisation:** this chapter includes components to support decision making and data analysis and visualisation from context information.
- **Interface with IoT, Robots and Third-Party Systems:** this chapter is related to software components that provide interfaces between context information and sensors/robots allowing them to get information, but also to be triggered in response to context updates.
- **Context Data/API Management, Publication and Monetization:** in this chapter are included the components for the data/asset publication and monetization (i.e. marketplace) but also related to security and API management
- **Deployment Tools:** this chapter includes supporting software for the containerization and deployment of the FIWARE components and system at scale.

Some of the FIWARE components can be considered more relevant in CommuniCity for their compliance with the MIMs specifications and standards:

- **NGSI-LD Context Brokers (MIM1):** In the FIWARE catalogue several implementations of NGSI-LD [7] compliant context brokers (MIM1) are published and implemented in different programming languages and suitable for different types of deployments (Orion-LD [6], Scorpio [11], Stellio [12]). These components allow managing and requesting context information. Context Producers can manage their context creating, updating, appending and deleting context information. Usually, the context brokers support both synchronous query-response, and an asynchronous subscribe / notify, where notifications can be based on a change in property or relationship, or on a fixed time interval.

It is also useful to mention, as part of the context management resources, the ETSI NGSI-LD API Conformance Test Suite [13], a robot framework test suite defined by ETSI ISG CIM, that can be used to validate the compliance of a Context Broker or any component using with the official API specifications.

- **Business API Ecosystem** [14] (MIM3): this component allows the monetization of different kinds of assets (both digital and physical) during the whole service life cycle, from offering creation to its charging, accounting and revenue settlement and sharing. The Business API Ecosystem provides sellers the means for managing, publishing, and generating revenue of their products, apps, data, and services. The Business API Ecosystem GE is a joint component made up by integrating the FIWARE Business Framework with a set of standard APIs (and its reference implementations) provided by the TMForum in its TMF API ecosystem.

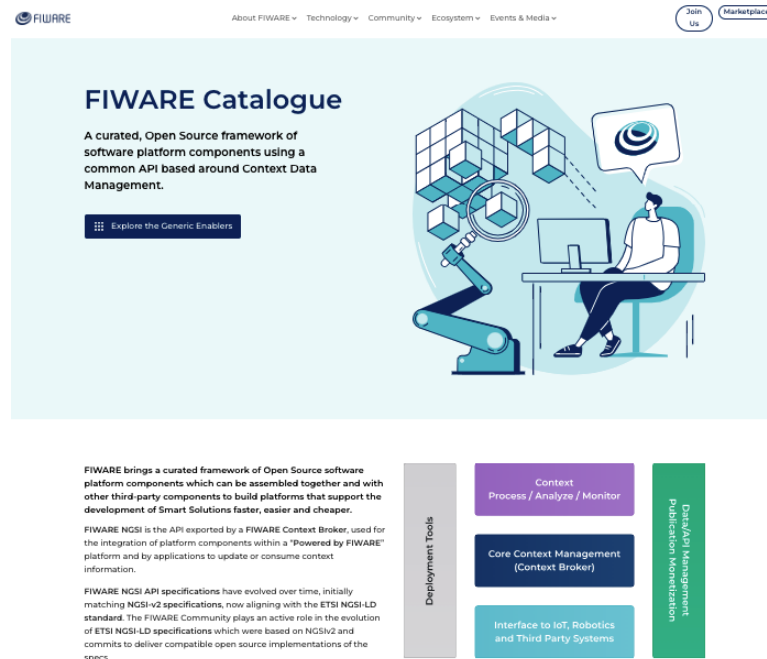


Figure 5 - FIWARE Catalogue website

### 3.3.2 Ontologies and common data models

The use of common data models in different systems and domains is identified as a key interoperability feature ensuring applications' replicability. As already mentioned in D4.1 CommuniCity Framework refers to MIM2 for the definition and implementation of this aspects suggesting a set of data models and ontology ready to be reused including: NGSI-LD compliant data models of the Smart Data model initiative, SAREF (Smart Applications REFERENCE ontology [15]), oneM2M base ontology [16], Core vocabularies of ISA like Core Public Service Vocabulary Application Profile [17], DTDL is the Digital twin Definition Language [18] developed by Microsoft.

All these initiatives include extensive repositories of standard data models and ontologies specific for the most relevant applicative domains giving to the developers the possibility to reuse or easily extend these resources for the design and implementation of their smart applications.

☰ README.md

## dataModel.Parking

These data models are intended to model entities relevant for parking use cases in smart cities scenarios. When feasible these models reuse types, properties and enumerations from [DATEX II version 2.3](#). A data dictionary for DATEX II terms can be found at <http://datexbrowser.tamtamresearch.com/>. Nonetheless, these data models are intended to NGSI-based systems and many simplifications has been made with respect to DATEX II version 2.3.

### List of data models

The following entity types are available:

- [OffStreetParking](#). Off street parking
- [OnStreetParking](#). A site, open space zone, on street, (metered or not) with direct access from a road, intended to park vehicles.
- [ParkingAccess](#). Parking Access - TODO: Provide a complete Schema
- [ParkingGroup](#). Parking Group
- [ParkingSpot](#). A parking spot is an area well delimited where one vehicle can be parked.

### Contributors

[Link](#) to the 7 current contributors of the data models of this Subject.

### Contribution

You can raise an [issue](#) or submit your [PR](#) on existing data models

*Figure 6 -Example of data models (Parking data models) of Smart Data Model initiative*

## 4. Online support system

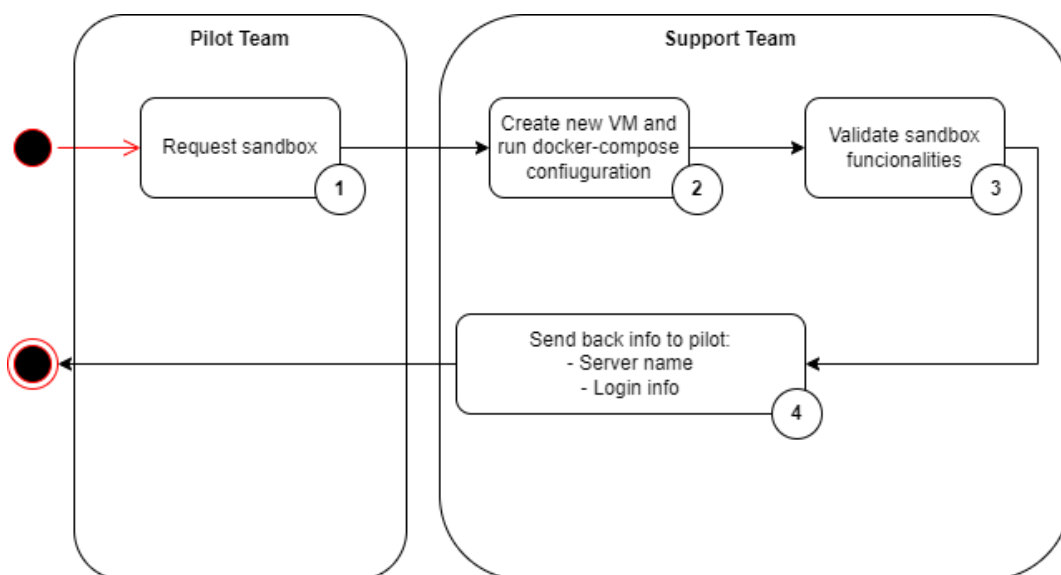
To assist the open call pilots in requesting access to sandboxes or troubleshooting an online ticketing system is used in CommuniCity. This ticketing system is managed by the support team of CommuniCity and allows:

- Requesting a sandbox;
- Requesting changes to the resources of the sandbox;
- Deleting a sandbox;
- Report issues regarding the sandbox or toolbox;
- General support questions regarding the sandbox or toolbox.

Mechanisms for human oversight, transparency, and auditability are built into the AI system, ensuring that the system is designed to avoid bias in input data and algorithmic design.

Additionally, the support system complies with data protection and privacy principles, such as data minimization. All data managed within the ticketing system is handled with strict confidentiality and is used solely for the purpose of providing support. The system ensures that any data provided by users is securely stored and erased in accordance with data protection guidelines.

Furthermore, the impact of the developed and/or used AI system on individuals, society, and the environment is carefully evaluated to avoid any possible risks of harm. This approach ensures that our online support system is not only efficient and responsive but also ethically sound and socially responsible.

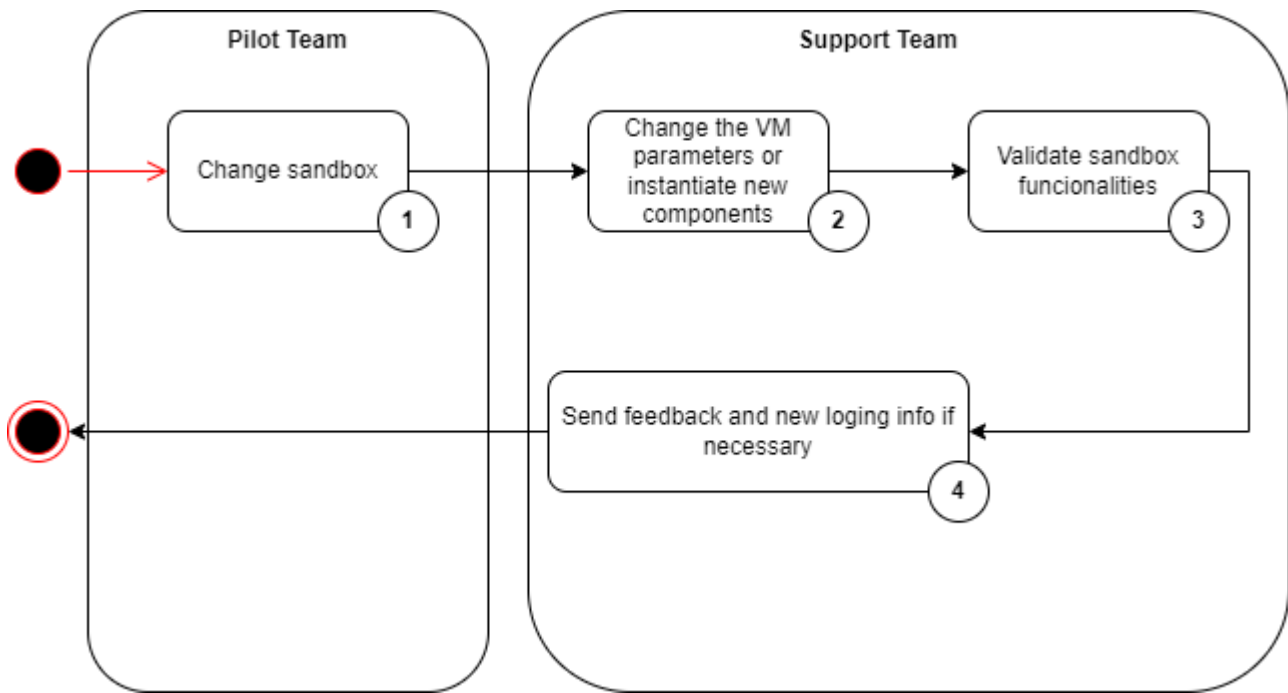


*Figure 7: Example flowchart of a request for a new sandbox*

Figure 7 show an example process flow for the request of a sandbox. This is a 4-step process involving both the pilot and support teams. The steps are:

- 1) Requesting the sandbox: the pilot team submits a ticket to request a sandbox as shown in Figure 10.

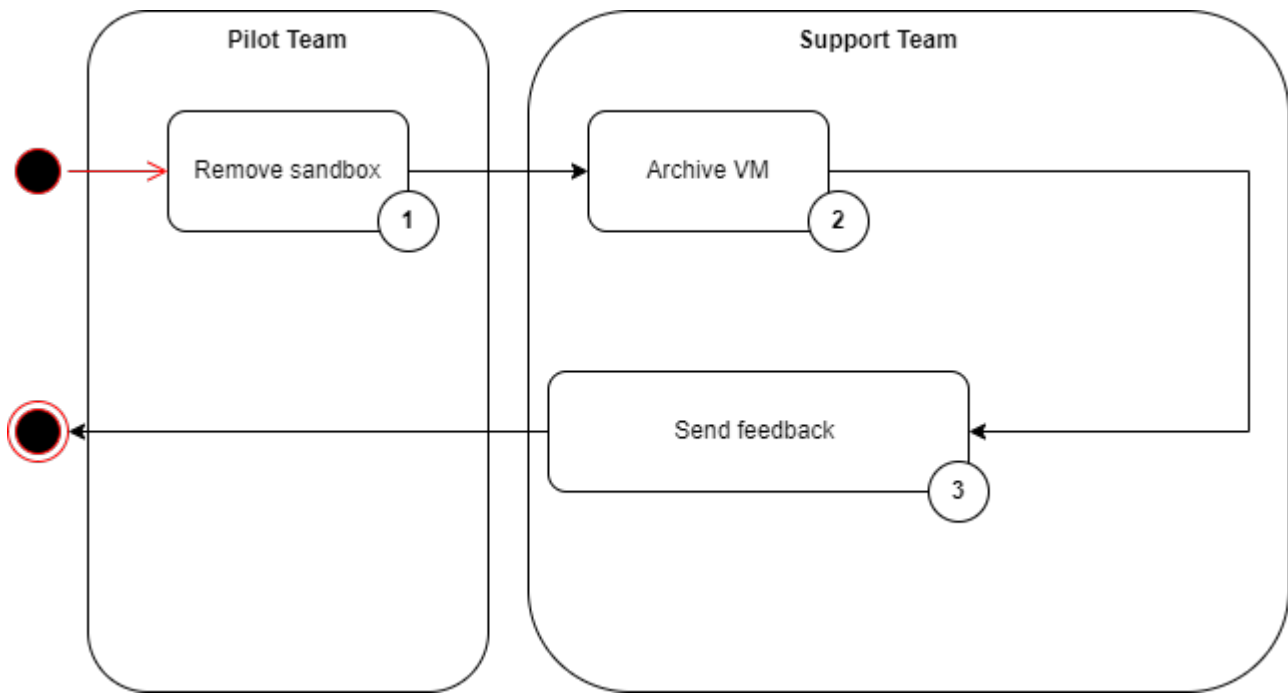
- 2) Creating the sandbox: the support team creates a new virtual machine (VM) and instantiates the necessary components by running a docker-compose recipe, as shown in Figure 2 and 3.
- 3) Validate the sandbox: the support team then validates the VM by testing the endpoints and component status.
- 4) Returning the access and login info: if the tests are successful the support team, then replies to the ticket submitted with the necessary information for the pilot team to access the sandbox.



*Figure 8: Example flow for the process to change the parameters of the sandbox.*

Figure 8 shows an example process flow for the change of the parameters of a sandbox. This is a 4-step process involving both the pilot and support teams. The steps are:

- 1) Requesting changes to the sandbox: the pilot team submits a ticket to changes to the sandbox (similar to the request shown in Figure 10).
- 2) Changing the sandbox: the support team alters the configuration of the VM or the components instantiated (for example installing new packages or software in the VM).
- 3) Validate the sandbox: the support team then validates the VM by testing the endpoints and component status.
- 4) Returning the access and login info: if the tests are successful the support team, then replies to the ticket submitted with the necessary information for the pilot team to access the sandbox and a summary of the changes executed.



*Figure 9: Example flow for the process to remove a sandbox.*

Figure 9 shows an example process flow for the change of the parameters of a sandbox. This is a 3-step process involving both the pilot and support teams. The steps are:

- 1) Requesting to remove the sandbox: the pilot team submits a ticket to remove a sandbox (similar to the request shown in Figure 10).
- 2) Removing the sandbox: the support team archives the VM, it will only be removed after the pilot period is finished.
- 3) Sending feedback: Once the sandbox is archived, the pilot team is informed.

For other general support and requests, the process flow is more flexible as each request may require a different approach. However, all processes start with the submission of a ticket which is received at the support ticketing system. Daily the support team will address new tickets when possible and send feedback as soon as issues are resolved. In the remainder of this section, we explain the steps taken by the support team on the ticketing system.

## 4.1 Opening a Ticket

To open a ticket in the support system, one needs to send an email to [communities@portodigital.pt](mailto:communities@portodigital.pt) and specify in the Subject the topic you require assistance with and in the Body of the email describe the problem as best as possible. This would be the procedure to start any assistance request to the support team of CommuniCity.

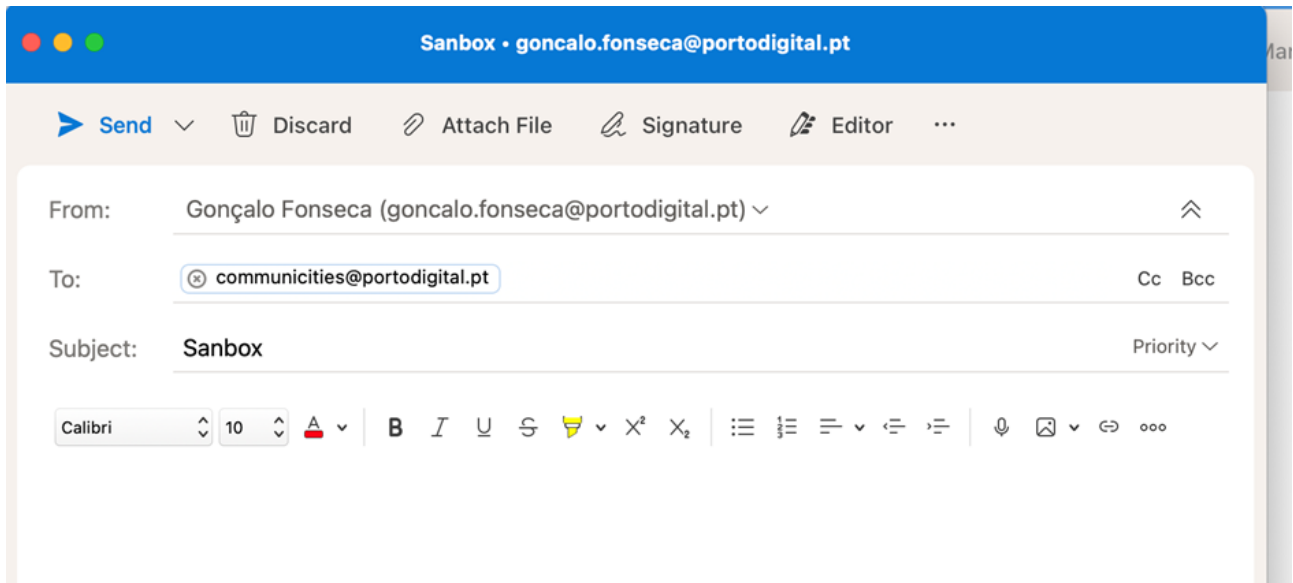


Figure 10 - Support system official mail

## Managing Tickets

For the support team to manage tickets, it is necessary to access the ticketing system. This is hosted at <https://servicedesk.portodigital.pt/>:

A screenshot of a login page titled "Entrar" with version "5.0.1" in the top right. It features a login form with "Username:" and "Password:" labels. The username field contains "goncalo.fonseca@portodigital.pt" and the password field is filled with dots. A blue "Entrar" button is at the bottom right of the form. Below the form, it says "For local help, please contact servicedesk@portodigital.pt".

Figure 11 - Ticketing system login

Once logged in, one can select the *queue* related to the CommuniCity project called Communities :

^ Queue list 

Queue	novo	aberto	pendente
APD Internal	-	5	-
CITYFLOW - CROA	-	-	-
Communities	1	-	-

*Figure 12 - Ticketing System – queue list*

Once within the list, one can view, comment and resolved any available issues.

Found 1 ticket Editar Pesquisa Avançado Mostrar Resultados Atualização em bloco 6

#	Assunto	Estado	Queue	Dono	Prioridade
	Requerente	Criado	atualizado	Última atualização	Tempo disponível
3243	Re: Sanbox goncalo.fonseca@portodigital.pt	novo 3 minutes ago	Communities	Nobody in particular 16 seconds ago	Low

*Figure 13 - Ticketing system – ticket list*

Each ticket will contain the information sent via email , as explained in **Opening a Ticket**. At this point, a support person can reply to ask for more information or provide feedback on an issue or select an update to the status of the ticket, for example, to resolve it.

## 5. Conclusions

Deliverable D4.3 reported the main resources for the support of open calls projects developers in understanding the technical approach of the CommuniCity project and to develop and validate interoperable and replicable solutions across different cities and domains. The support resources are based on three main pillars: (1) the knowledge base and learning resources, including official project documentation (and external references) and the online technical webinars, (2) technical assets to simplify development and test of (AI based) services in compliance with the CommuniCity technical framework; (3) the online ticketing system to allow developers to require access to dedicated services and support for technical issues.

The information included in this document is also aimed at the pilot cities participating in the project to clearly understand the potential benefits of the CommuniCity technical framework and the way to adopt and integrate it with the City IT systems during the piloting phase. Moreover, the documents and resources referred to in D4.3 will be also useful for external stakeholders that want to understand and apply the approach of CommuniCity in building standards and interoperability (AI based) services.

Even if this deliverable does not have any other official update, most of the content will also be provided as online documentation (on the project web site [19]) in order to be further improved and updated, to follow the project evolutions, providing up to date information to the developers of the future open calls.

## 6. References

[1]	C. Project, «D4.1 - Technical architecture and APIs,» [Online].
[2]	C. Project, «D4.2 "Pilot Toolbox and Validation Sandbox",» [Online].
[3]	«CommuniCity technical documentation,» [Online]. Available: <a href="https://communicity-docs.readthedocs.io/">https://communicity-docs.readthedocs.io/</a> .
[4]	«OASC MIMs,» [Online]. Available: <a href="https://mims.oascities.org/">https://mims.oascities.org/</a> .
[5]	«MIMs version 2023,» [Online]. Available: <a href="https://docs.google.com/document/d/1nC5wSulop4lg5xWtZ4nsmfbMs4mfc1eQ/edit">https://docs.google.com/document/d/1nC5wSulop4lg5xWtZ4nsmfbMs4mfc1eQ/edit</a> .
[6]	«Orion-LD,» [Online]. Available: <a href="https://github.com/FIWARE/context.Orion-LD/">https://github.com/FIWARE/context.Orion-LD/</a> .
[7]	«ETSI NGSI-LD specifications,» [Online]. Available: <a href="https://www.etsi.org/deliver/etsi_gs/CIM/001_099/009/01.07.01_60/gs_cim009v010701p.pdf">https://www.etsi.org/deliver/etsi_gs/CIM/001_099/009/01.07.01_60/gs_cim009v010701p.pdf</a> .
[8]	«CommuniCity Toolbox,» [Online]. Available: <a href="https://github.com/CommuniCityProject/communicity_toolbox">https://github.com/CommuniCityProject/communicity_toolbox</a> .
[9]	«CommuniCity Toolbox online demos,» [Online]. Available: <a href="https://communicity.portodigital.pt/">https://communicity.portodigital.pt/</a> .
[10]	«FIWARE Catalogue,» [Online]. Available: <a href="https://www.fiware.org/catalogue/">https://www.fiware.org/catalogue/</a> .
[11]	«Scorpio Broker,» [Online]. Available: <a href="https://github.com/ScorpioBroker/ScorpioBroker">https://github.com/ScorpioBroker/ScorpioBroker</a> .
[12]	«Stellio Broker,» [Online]. Available: <a href="https://github.com/stellio-hub/stellio-context-broker">https://github.com/stellio-hub/stellio-context-broker</a> .
[13]	«NGSI-LD Test Suite,» [Online]. Available: <a href="https://forge.etsi.org/rep/cim/ngsi-ls-test-suite">https://forge.etsi.org/rep/cim/ngsi-ls-test-suite</a> .
[14]	«Business API Ecosystem,» [Online]. Available: <a href="https://github.com/FIWARE-TMForum/Business-API-Ecosystem/">https://github.com/FIWARE-TMForum/Business-API-Ecosystem/</a> .
[15]	«SAREF: the Smart Applications REference ontology,» [Online]. Available: <a href="https://saref.etsi.org/core/v3.1.1/">https://saref.etsi.org/core/v3.1.1/</a> .
[16]	oneM2M, «oneM2M Base Ontology,» [Online]. Available: <a href="https://www.onem2m.org/images/pdf/TS-0012-Base_Ontology-V3_7_3.pdf">https://www.onem2m.org/images/pdf/TS-0012-Base_Ontology-V3_7_3.pdf</a> .
[17]	«Core Public Service Vocabulary Application Profile (CPSV-AP),» [Online]. Available: <a href="https://ec.europa.eu/isa2/solutions/core-public-service-vocabulary-application-profile-cpsv-ap_en/">https://ec.europa.eu/isa2/solutions/core-public-service-vocabulary-application-profile-cpsv-ap_en/</a> .
[18]	«Digital Twins Definition Language (DTDL),» [Online]. Available: <a href="https://azure.github.io/opendigitaltwins-dtdl/DTDL/v3/DTDL.v3.html">https://azure.github.io/opendigitaltwins-dtdl/DTDL/v3/DTDL.v3.html</a> .
[19]	«CommuniCity Website,» [Online]. Available: <a href="https://communicity-project.eu/">https://communicity-project.eu/</a> .