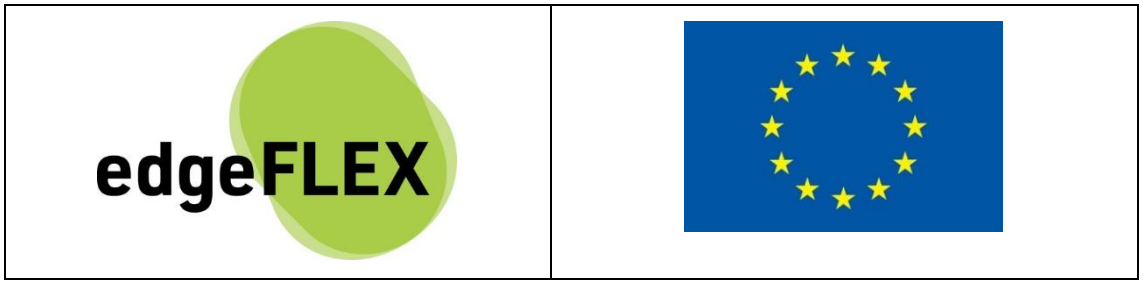


edgeFLEX

D4.3

Description of internal interfaces for control services

The research leading to these results has received funding from the European Union's Horizon 2020 Research and Innovation Programme, under Grant Agreement no 883710.

Project Name	edgeFLEX
Contractual Delivery Date:	30.03.2022
Actual Delivery Date:	30.03.2022
Author(s):	WIT
Workpackage:	WP4 – Platform and Services for Dynamically Controlled VPP Solutions
Security:	P
Nature:	R
Version:	1.0
Total number of pages:	38

Abstract

The goal of the edgeFLEX project is to advance the role of the VPP with the use of advanced grid management techniques, effective optimisation, flexibility provision and trading combined with enabling solutions such as Service Level Agreement Monitoring tools, edgePMU devices and 5G capabilities. This report details how the MVP developed in the first phase of the project has been further enhanced, with a particular view of the internal interfaces which allow the edgeFLEX platform to interact with Grid Control services and also external services such as VPP optimisation and engaging with Flexibility trading platforms. The report describes these interfaces in terms of functional architecture, technologies and how Policy Based Grid Management and 5G communication features support the operation of these interfaces to enable the advancement of the VPP.

Keyword list
Virtual Power Plant, Optimisation, Flexibility Provisioning, Control, enhanced measuring, grid management, 5G, Automatic SLA Monitoring, Architectures, Interfaces

Executive Summary

The edgeFLEX platform is a core element of edgeFLEX, which enables the operation of the Grid Control, and VPP optimisation and interaction with Flexibility Trading entities in an effort to advance the role of the VPP.

During the first phase of the edgeFLEX project, an effort was made to gather detailed requirements from the perspective of the Grid Management, VPP Optimisation and Flexibility Trading, and finally from the field trials. Based on the requirements gathered in phase 1, architectures and services were defined for the development of the edgeFLEX platform; this led to the development of a version of the edgeFLEX platform that was built to serve as a Minimum Viable Product (MVP) and was used to gain insight from the customer, the outputs of which were used to improve the edgeFLEX platform. The requirements gathering and platform design process are described in detail in D4.1 – Description of the edgeFLEX Platform Design.

This report describes the data interfaces defined in phase 1 of edgeFLEX, based on the functional and non-functional requirements gathered, and how these interfaces enable data communication between components of the edgeFLEX platform. These data interfaces will be described in a number of contexts, such as how they support the overall platform architecture and the communication protocols and technologies utilised to implement these interfaces, in particular MQTT and REST.

The report also considers the context of security in relation to the edgeFLEX platform interfaces, as security has been of increasingly large concern in recent years due to the level of cyber-security threats faced by ICT systems, and particularly in relation to vital infrastructure such as power systems. Therefore, consideration was made for how data interfaces and components were secured in terms of networking, access credentials, data access and sharing constraints between components.

Related to this, the report also considers the context of the Policy Based Grid Management (PBGM) system and how user defined policies are utilised to enable the PBGM component to provide observability via data sharing, and in addition, to enable flexibility requests and interact with grid assets.

This report also considers 5G and its capabilities to enable fast, secure data sharing for the edgeFLEX platform's internal interfaces. This is described through the usage of the 5G Device Management API and how it may be used by the DSO to manage a large number of devices which are deployed on the grid and act as data sources, and how it could become an enabler for more secure and reliable data sharing between devices and services when combined with a system such as the edgeFLEX PBGM component.

One of the key goals of the edgeFLEX project is the advancement of the VPP. In Phase 1 of the project, the gathered requirements were used to define architectures, components, and services to realise this goal. Phase 2 of the edgeFLEX project involved the development of the edgeFLEX MVP, a tool which has been used to develop and improve the edgeFLEX platform components further. In the last phase of the project, the culmination of the work described in the previous deliverables will be demonstrated on the trials and subsequent deliverables and presentations.

Authors

Partner	Name	e-mail
WIT		
	David Ryan	david.ryan@waltoninstitute.ie
	Darren Leniston	darren.leniston@waltoninstitute.ie
	Jack Jackman	jack.jackman@waltoninstitute.ie
RWTH		
	Edoardo De Din	ededin@eonerc.rwth-aachen.de
	Gianluca Lipari	glipari@eonerc.rwth-aachen.de
	Diala Nouti	dnouti@eonerc.rwth-aachen.de
EDD		
	Robert Farac	robert.farac@ericsson.com
	Tuana Ensezgin	tuana.ensezgin@ericsson.com

Table of Contents

1. Introduction

One of the core outputs of the edgeFLEX project is the edgeFLEX platform, which hosts the Grid Control and VPP Optimisation services and engages with the Flexibility Trading elements of the project. A functional requirement of this is the need to have internal interfaces that can allow the services to communicate with other components, both internal to the platform and external, so that data can be accessed, and control messages sent in a coherent, efficient, and secure way. In the early stages of the edgeFLEX project, a set of functional and non-functional requirements were gathered from the partners developing the Grid Management, VPP Optimisation and Flexibility Trading components of the project, and when coupled with those gathered from the field trials, the interfaces were defined. The analysis of the requirements focused mainly on what data needed was needed, where was it gotten from, what was the likely format and what were the constraints and security concerns around accessing it, with the aim of developing the interfaces that are both internally and externally facing.

This report will present these interfaces in the context of:

- the overall architecture and the services they enable.
- the protocols and technologies used.
- the security measures and schemes employed.
- how Policy Based Grid Management is used to manage the interfaces.
- how external data streams are managed within the platform.
- how the features of 5G can provide secure and reliable communications for the interfaces to use.

1.1 Related Work

This report is based on work carried out in the first phase of the project and the early stage of the second phase of the project. The precursor to this report is D4.1 Description of edgeFLEX platform design, where the platform was described from a technical perspective, with the individual architectures focused on the trial site implementation and the data flows between the services and the trial site components. This report focuses on taking the architectures developed in Task 4.1, and based on the requirements gathered from WP1 and WP2, both functional and non-functional, describes the interfaces as defined and developed. The grid management services from WP1 and WP2, the VPP Optimisation, the edgePMU and the 5G work from WP3 are also considered in defining the interfaces along with the Flexibility Trading, the interaction with the GOFLEX platform (Task 4.5) and the SLA Monitoring Tool (Task 4.4). The work in this report also considers the trials in WP5 when detailing the interfaces as a critical component of the operation of the services and how well they integrate into the existing systems, like the KIBERNet system for example. Figure 1 details graphically the task linkages from the technical tasks of the project and the Trials in WP5, and how they feed into the definition and development of the edgeFLEX platform internal and external interfaces.

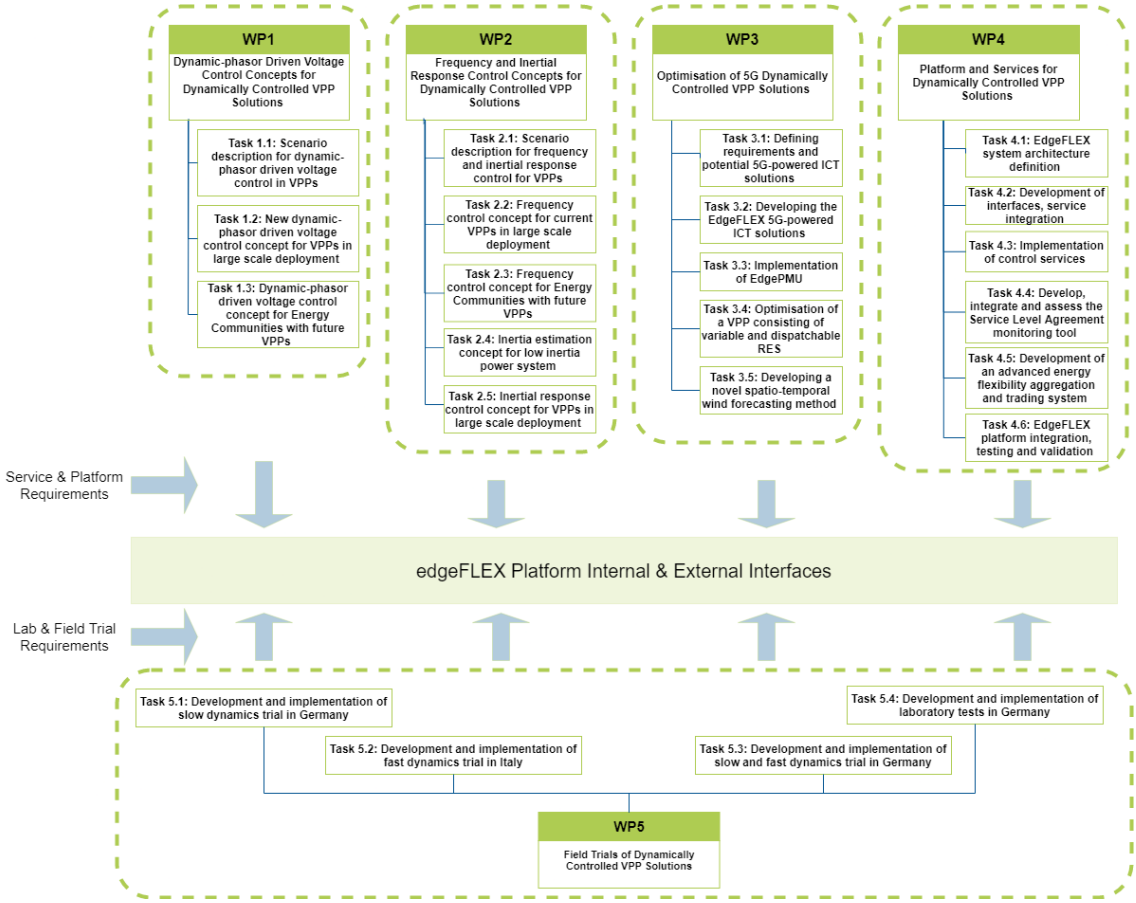


Figure 1 - edgeFLEX requirements defining the interfaces

1.2 Objectives of the Report

The main objective of this report is to describe the edgeFLEX internal interfaces that will be used by the control services. Given the modularity and portability of the edgeFLEX platform, it is also important to detail the external interfaces. The aim is to give the reader a clear picture of how the services are enabled in a secure and efficient way by describing all of the relevant interfaces used by the platform.

1.3 Outline of the Report

This report (in Section 2) briefly describes the edgeFLEX platform, its architecture and how it is deployed, with particular focus on where the interfaces within and to the platform are situated. The interface technologies and the protocols used are then detailed in Section 3. Section 4 details the security aspects of the interfaces, with consideration around how they are accessed, communicated with, networked and how the data is shared throughout the platform. The report, next, details the requirements for the Grid Management services in terms of data access and device control (Section 5). Section 6 describes how Policy Based Grid Management (PBGM) and its deployment within the platform enables and controls the exposure of the interfaces, manages data sharing and facilitates the connection to external interfaces for data access (further described in Section 7) and flexibility acquisition. 5G and the use of its features are considered when developing the interfaces, and this is detailed in Section 8.

1.4 How to Read this Document

This document follows up on the description of the platform in D4.1, and in the initial section summarises briefly the platform, the services and the architecture. In there, we mention such concepts as Voltage Control, Inertia Estimation, VPP Optimisation and Frequency Control and

while D4.1 provides details of their implementation, a more in-depth view of the research is contained in the following deliverables.

- D1.1: Scenario description for dynamic-phasor driven voltage control for VPPs and D1.2: Dynamic-phasor driven voltage control concept for current VPPs
- D2.2: Frequency control concept for current VPPs D2.4: Inertia estimation concept for low inertia power system
- D3.2: Report on VPP optimisation

All other concepts that are not in a deliverable from another work package will be explained in a comprehensive way, or linkages to prior research from other projects will be cited and referenced where appropriate.

2. edgeFLEX Architecture Description

The edgeFLEX architecture is one which has been designed with the aim of providing flexibility in terms of configuration and deployment. The edgeFLEX architecture can be described as a loosely coupled set of services which may be deployed together or in isolation according to the needs of the System Operator or specific use-case.

To support this aim, a functional architecture was designed to support the flexibility required and utilised to implement the edgeFLEX platform MVP, described in greater detail in D4.2 – Description of the edgeFLEX MVP. This architecture captures all elements of the edgeFLEX platform, data collection and communications, and communications between internal and external services and entities. This section will focus specifically on the edgeFLEX backbone tools and services, highlighted in Figure 2 of the edgeFLEX functional architecture.

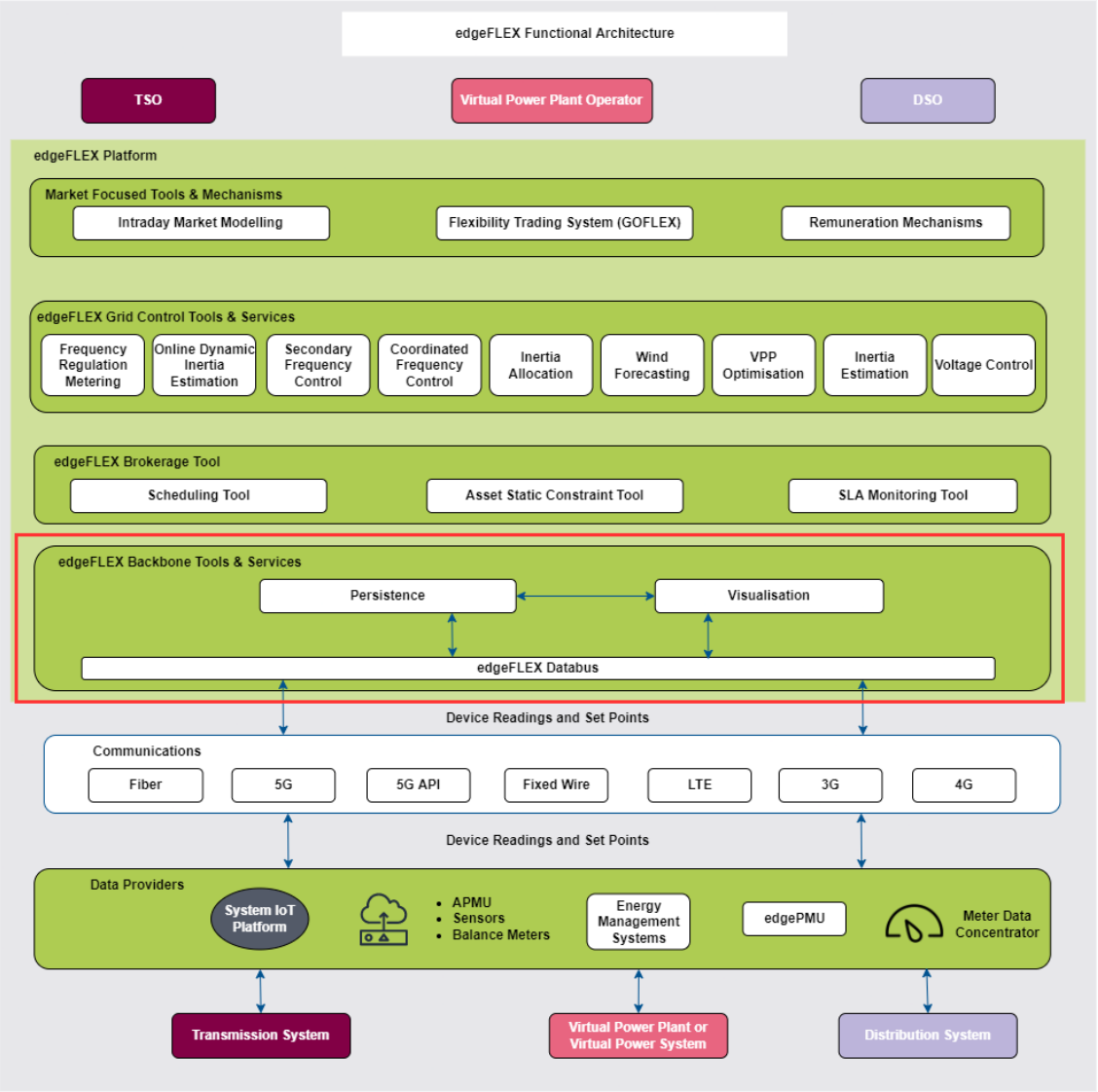


Figure 2 - edgeFLEX Functional Architecture

The edgeFLEX backbone is a key component of the platform, providing services and interfaces for the collection, storage, and visualisation of data from data providers such as the grid and control services and the edgePMU. The backbone services follow the same flexible approach as the greater edgeFLEX platform by enabling the instantiation of persistence and visualisation on a per service basis. For example, the Inertia Estimation service requires persistence for the scheduled analysis of stored data, while this is not a requirement for other services. Similarly, the visualisation functionality provided by Grafana, can be deployed on a per-service basis, giving

the System Operator a greater level of control over which services and functionality they wish to deploy within their environment.

The edgeFLEX databus provides the core interface for both upstream and downstream data into the edgeFLEX backbone and facilitates the collection of data via MQTT communications technology. MQTT operates on a publish/subscribe model; in this way data from devices (e.g. the edgePMU) and services such as voltage control can be either passed or consumed via specific MQTT topics. The use of specific topics with a standardised naming format for different data types ensures that services can access the required data from the edgeFLEX databus.

As described above, the edgeFLEX architecture contains several components that interact with both internal services and external systems. To facilitate these interactions, the edgeFLEX architecture implements several interfaces to ingest and return data. Taking the Voltage Control service as an example, the edgeFLEX backbone provides the edgeFLEX databus, which provides the core component for communications between the voltage control service, edgePMU, policy loader for interaction with the policy-based network management system and external KIBERNet system for flexibility requests made on behalf of the Voltage Control service.

Each of these systems and components contain their own interfaces for consuming and passing data, which are implemented using a combination of MQTT and REST technologies. For example, the Voltage Control service and edgePMU communicate via MQTT directly with the edgeFLEX databus. However, the policy-based network management system (i.e., SLA monitoring tool) exposes policies via a REST interface, and this is also the case for the KIBERNet system. Therefore, the policy loader component in the edgeFLEX backbone serves to both provide a method of requesting data from both the policy-based network management system and KIBERNet via REST interface, and to also translate the response received to be communicated via MQTT through the edgeFLEX databus to the Voltage Control service as illustrated in Figure 3.

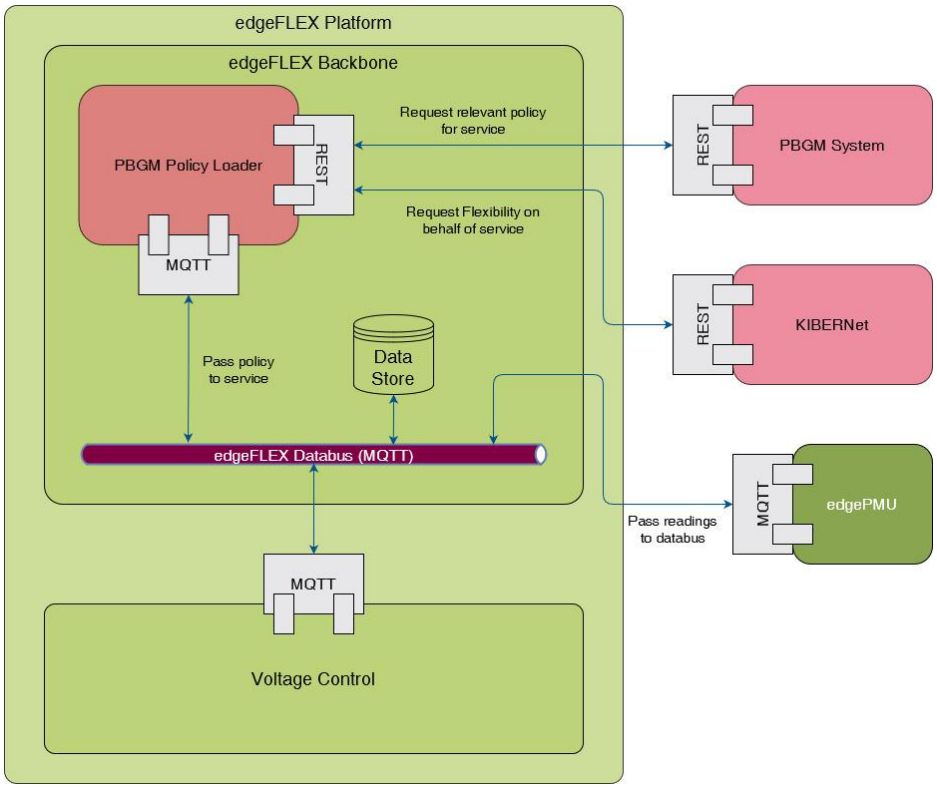


Figure 3 - edgeFLEX Architecture Interfaces in Voltage Control Use Case

2.1 edgeFLEX Architecture Deployments

As each use case is unique, it is important for the edgeFLEX system to have the capability to operate within a variety of deployment architectures to support as many use cases as possible. Due to the loosely coupled nature of the edgeFLEX services and the flexibility afforded through the overall edgeFLEX architecture design, several potential deployment architectures are supported. This flexibility is also enabled through the configurable services and interfaces within the edgeFLEX platform and backbone services such as the databus; such configuration provides the capability for System Operators to deploy parts of the system on the edge, or to connect them to existing systems within their environment. For example, there may already be a databus in place, therefore the System Operator may not need to deploy the edgeFLEX databus and may instead configure the grid control service(s), persistence, and visualisation to communicate with the existing databus. Likewise, the same process is supported if an existing, compatible persistence service or visualisation service is in place, offering the System Operator the flexibility to utilise the parts of the edgeFLEX platform that they need or wish to use on a use case dependent basis. Further detail on the configurable nature of the edgeFLEX services and components can be found in section 8 In this section, three potential architectures will be explored, namely; centralised, decentralised and hybrid mobile edge.

2.1.1 Centralised Deployment

Within the example centralised deployment proposed in Figure 4, the System Operator is deploying the Voltage Control service and edgeFLEX backbone in its entirety within their cloud environment, with communications outside the cloud environment limited to external systems such as the PBGM and KIBERNet systems. This configuration minimises the communication latency as compared with a distributed deployment, as the backbone and services are contained within the same environment which mitigates some of the security concerns when dealing with distributed resources and communications from outside the system.

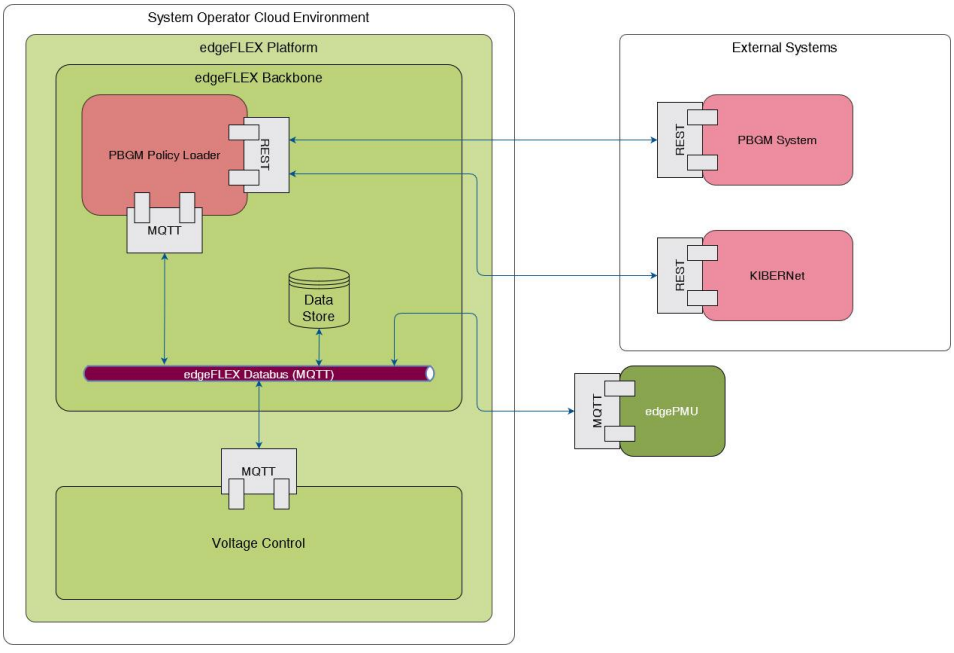


Figure 4 - Centralised Deployment Example

2.1.2 Decentralised Deployment

The deployment presented in Figure 5 illustrates a proposed decentralised deployment. Depending on the needs of the System Operator, parts of the edgeFLEX system, such as the control services (in this example, Voltage Control) may need to be deployed in a separate location, for example on a machine or even Raspberry Pi within a substation. The edgeFLEX components are configurable in such a way to make this possible. This example also

demonstrates the capability of using an existing resource, such as a databus, as an alternative to deploying the edgeFLEX databus if it is not needed. In this example, the System Operator has configured the Voltage Control, edgePMU and policy loader to interface with the existing databus.

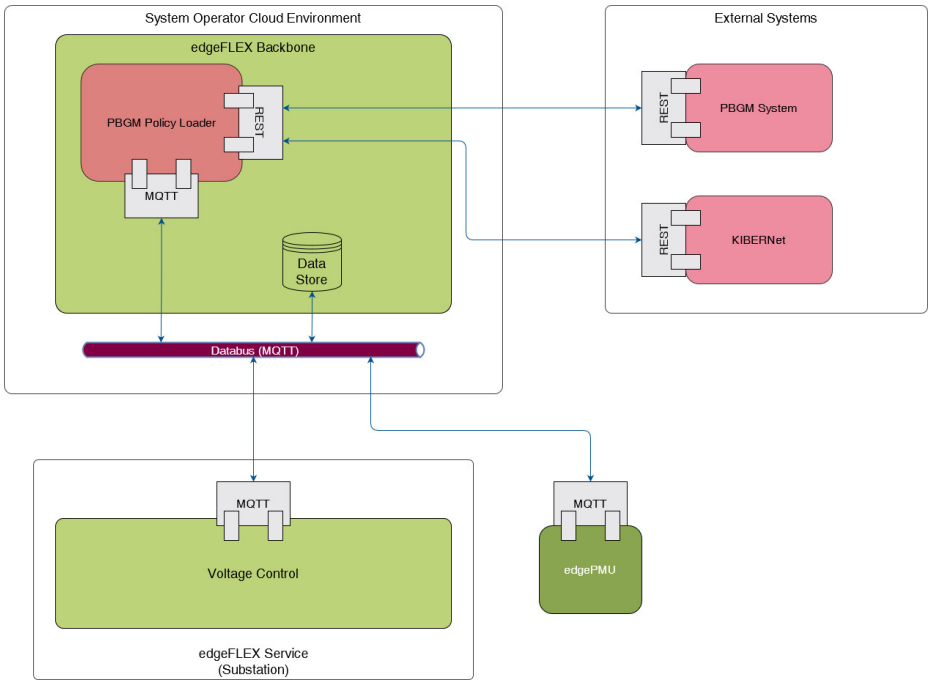


Figure 5 - Decentralised Deployment Example

2.1.3 Hybrid Edge Deployment

The final example illustrated in Figure 6 proposes a hybrid edge deployment which leverages the edge cloud and 5G. Communications across the 5G network provide several advantages from network slicing, latency and security, which massively curtails issues with operating devices on the edge. Within this configuration, the Voltage Control service is operating at the network edge, utilising the edge cloud and 5G communications to enable effective operation. Such capability offers increased flexibility to the System Operator and supports a wider range of use cases.

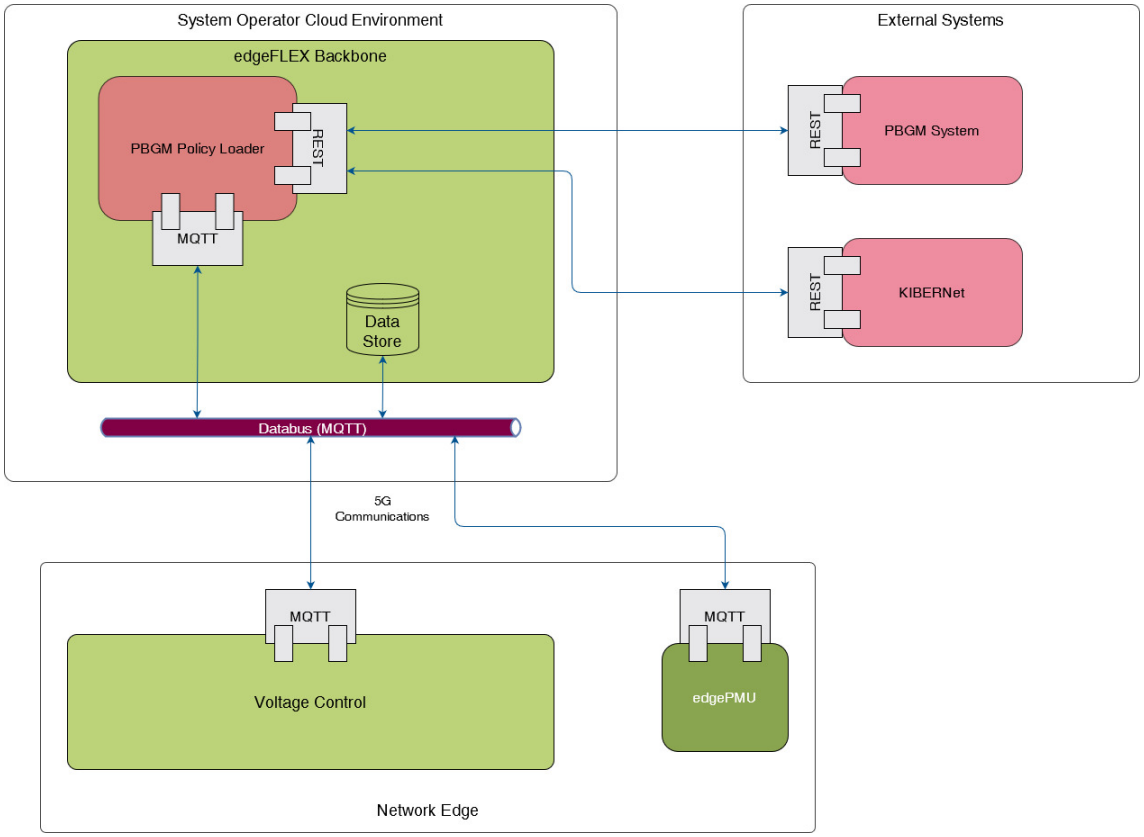


Figure 6 - Hybrid Edge Deployment Example

3. Interface Technologies & Protocols

The technologies and protocols used in the edgeFLEX platform include MQTT, REST and UDP. MQTT is a bi-directional messaging protocol that can be used to retrieve telemetry from devices in the field. It is lightweight, has a small code footprint and requires minimal network bandwidth, making it ideal for communicating with devices in remote locations [1]. MQTT makes use of an MQTT broker, which is a central software entity that handles messages between MQTT clients. The broker first allows devices to make a connection request, then authenticates the device's connection credentials, then ensures that the device can send and receive messages using encryption, before finally storing messages to prevent data loss. The advantages of a centralised broker are that it performs most of the processing, allowing client devices to operate with minimal processing and bandwidth, and that client devices are decoupled from one another, making it easy to add additional devices without affecting existing ones [2]. MQTT clients can publish and/or subscribe to an MQTT broker. Messages are published over a topic set by the client publishing that message, and clients that are subscribed to that topic will receive the message. In the edgeFLEX platform, telemetry from field devices will be published over the edgeFLEX databus, which is extended from the open-source DockerMQTT developed as part of the RESERVE project.

REST is an architectural style for designing loosely coupled applications over HTTP. An API or web service that conforms to the constraints of REST is known as a RESTful API. These constraints are as follows [3]:

- Must conform to a client-server architecture consisting of clients, servers and resources.
- The API must be stateless, meaning no client information is stored between requests.
- Data must be cacheable enabling the client applications to reuse response data to streamline client-server interactions.
- Uniform interface between components.
- Layered system that organises each type of server involved in the retrieval of requested information into hierarchies.
- Optional requirement that the system has the ability to pass executable code from the server to the client.

REST clients can access data in various formats such as HTML, XML, plain text, PDF, JPEG and JSON. REST APIs can be used to formally request data over HTTP so that requests can be logged and handled appropriately, as opposed to MQTT's publish-subscribe model. In the edgeFLEX platform, REST APIs are used primarily for retrieving policies from the SLA Monitoring Tool. The platform makes a request for a given policy by querying the policy endpoint with the given policy ID and policy group in the query parameters. The SLA Monitoring Tool validates the request and returns the policy in JSON format if the request is valid. The client will then direct the policy data for processing and policy enactment.

UDP is a lightweight data transport protocol used to send packets over IP. UDP can detect corrupt data in packets, but unlike TCP, it does not correct for packets received out of order or lost during transmission. UDP may be preferred in IoT applications for its lower latency, reduced network resource requirements and the fact that it does not require a constant connection between endpoints [4]. Samples from the edgePMU will be packed in UDP packages for transmission.

4. Interface and Platform Security

This section will describe, and detail security considerations made for the edgeFLEX platform and what measures have been taken to ensure that communications between interfaces and services have been made secure. With IoT communications and software automation becoming more and more ubiquitous in recent years across many industries, the need for robust security measures to ensure data privacy and reliable operations has only become more necessary [5].

Cyberattacks are now a common and constant threat to organisations and industries, leading to data theft and disruption. When considering vital utilities such as the electrical grid, the need for effective security becomes apparent [6], and the edgeFLEX platform services and interfaces operate with these factors in mind.

4.1 Secure Communications & Networking

The edgeFLEX platform, services and brokerage contain several REST and MQTT based interfaces. At a minimum, each interface implements Secure Socket Layer (SSL), Transport Layer Security (TLS) or HTTPS on the communication layer where appropriate. This ensures that data passed between interfaces on the edgeFLEX platform and brokerage is unreadable to any outside entities by encrypting data in transit via RSA encryption [7].

In addition to securing communications via encryption, security has been considered via networking configuration. In practice, the network configuration will depend on the System Operator's environment, however, the flexibility afforded through the edgeFLEX architecture, deployment and configuration enables several options. This has been tested in WIT by deploying each component of the edgeFLEX platform and brokerage in the cloud across several separate virtual instances and minimising traffic to the required ports and direct SSH access to instances via specific IP addresses, adding additional security by locking off any avenues of access for third parties.

4.2 Authenticated Access

In addition to data encryption, interfaces are secured with authentication to ensure that only permitted services have access to data from the edgeFLEX platform services, database, and brokerage. For example, the edgeFLEX databus is secured via TLS and authentication credentials which are set on deployment via configuration. Any services which require access to the databus must be supplied with the correct credentials. Similarly, the edgeFLEX brokerage is secured with authentication credentials to access the policy GUI.

The edgeFLEX brokerage system provides access to policies via REST endpoints. While communications from these endpoints are secured via SSL, authenticated access is managed via JSON Web Tokens (JWT). JWTs are a three-part string containing a header, payload and signature, and they provide a compact way of verifying requests to a resource from an outside entity [8]. In practice, when an entity makes a request for a policy a valid JWT token must be supplied in the request to the edgeFLEX brokerage. This token is then verified by the brokerage system. If a valid JWT token is not supplied, then no policy is returned. In this way only trusted and verified entities may access policy data.

4.3 Data Access & Sharing Constraints

The edgeFLEX brokerage system provides functionality for more fine-grained control of access to policy data. The policy system GUI contains options for specific policies to be accessible only to certain users and groups. This access control is managed by the system administrator who may create users, groups and define access for each. For a service or entity to request a specific policy, they must provide a valid asset ID or policy ID depending on the type of policy. These are defined when the policy is created via the GUI. To provide an extra layer of security, all IDs are binary-to-text base 64 encoded. When a request is made to the edgeFLEX brokerage system, the base 64 ID is decoded and verified against the requested policy type. If the request is valid, the policy is enforced, otherwise an invalid request message is returned. Additional verification is

enabled for FlexOffer requests, which are further validated based on the time and duration of the offer. Within the greater edgeFLEX platform, grid control services are supplied endpoints and MQTT topics for the data they require via configuration. In this way, the data that these services have access to is controlled on deployment.

The policies themselves as defined in the PBGM system by the user are another means by which data access and data constraints are managed for devices, grid services and flexibility. By utilising policies, devices may be configured to interact with specific endpoints and flexibility requests with duration and time for validation. At present, many of the edgeFLEX components are configured prior to deployment via configuration file. In future versions, this may be replaced with platform specific policies which will remove the need for configuration files. This not only provides a means for defining policy-based rules around data access and constraints, but also enables these to be updated without requiring re-configuration and re-deployment of components and services.

5. Policy Based Grid Management enabling external and internal interfaces

Policy Based Grid Management (of which the edgeFLEX SLA Monitoring Tool is a component) plays the role of the interface between the components of the edgeFLEX platform, the systems and the services that may be owned by an external entity. This interface is used to make flexibility requests to an Energy Community, supply the operational limits of a grid asset to an Energy Community or provide bi-directional observability by way of data sharing in an agreed and automated way.

PBGM, Figure 7, operates based on having a set of user defined policies stored in a Policy Repository where they are accessed via the Policy Access Point for use by the Policy Decision Point to create actions that are enforced by the Policy Enforcement Point. These policies could be created by a System Operator, a VPP Manager or a manager in an Energy Community, most likely an operative in the organisation where the platform is deployed and would be centred on providing the functionality for an external entity to engage with their systems in a monitored, managed, and automated way.

A core component of the PBGM system is the event service, which provides the interface to the systems that wish to interact with system that the policies are created to manage.

It contains a set of REST Interfaces and MQTT subscribers it works in two ways,

- as an event listener that monitors traffic, data requests and grid state changes and based on these events, looks for a corresponding policy and acts accordingly. Such actions may be the returning of data, the running of a Grid Management Service, such as Inertia Estimation, or the revoking of access.
- as a REST interface that serves the policy to the calling service. The policy would potentially contain such things as FlexOffer templates, connection details for external endpoints for data access, and the operational limits of assets.

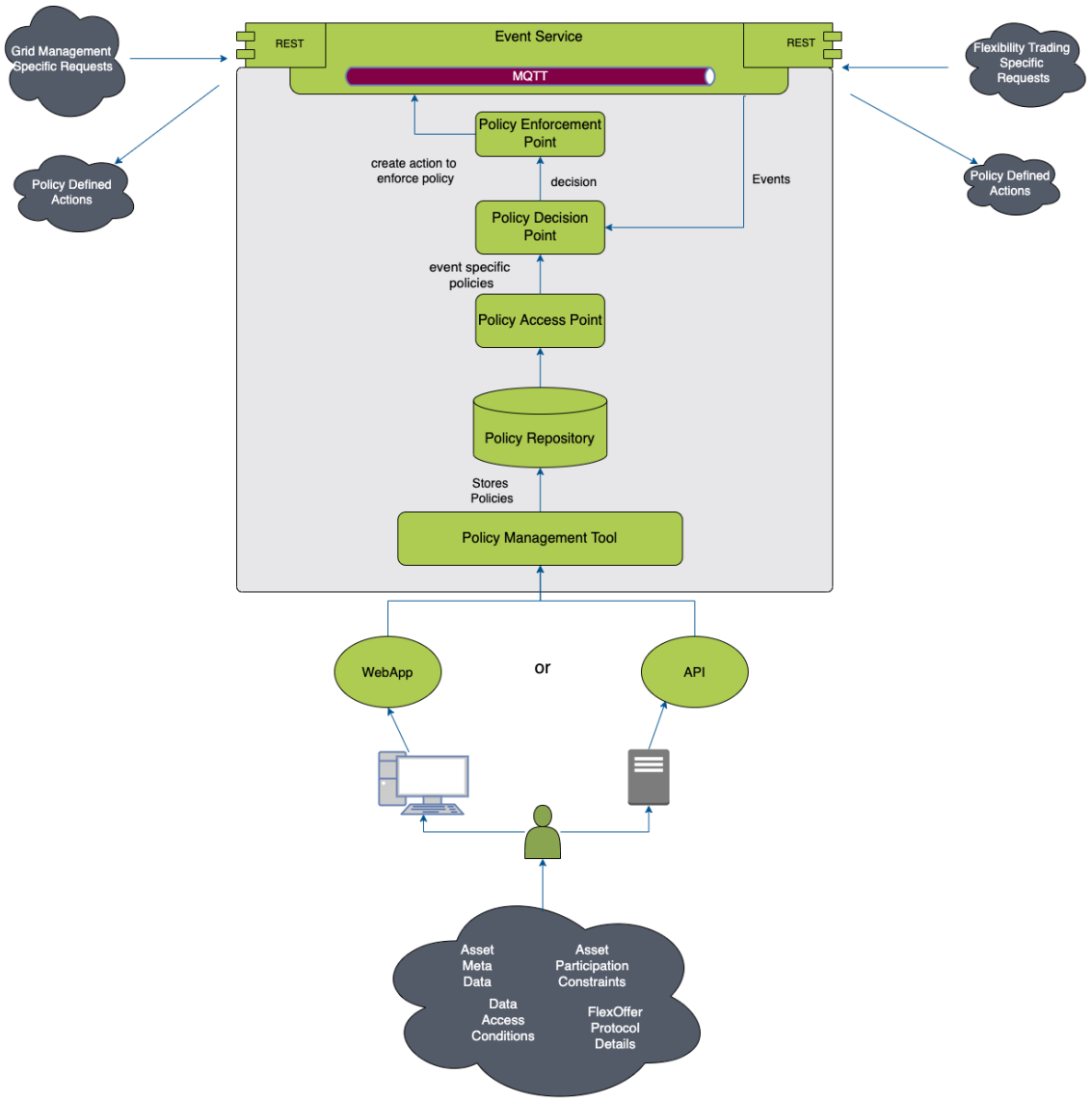


Figure 7 - PBGM Architecture

6. edgeFLEX Control Service Requirements

6.1 Inertia Estimation

The inertia estimation service developed within edgeFLEX serves as a monitoring tool for System Operators to accurately estimate the overall system inertia. The inertia estimation is achieved by collecting frequency and active power measurements of the generating units' substations and then performing parametric regression based on post-processing of these measurements. Consequently, the inertia estimation service has specific functional requirements, which are supported by edgeFLEX backbone services such as:

- Persistence for the post-processing of stored measurements.
- The event service, as part of the PBGM system, which acts as an event listener to monitor data traffics and detect the needed excitation (rate of change of frequency) to run the estimation algorithm and ensure the convergence of the algorithm.
- Visualisation, provided by Grafana, which helps System Operators make informed decisions efficiently.

The inertia estimation service together with above mentioned edgeFLEX backbone services are to be deployed in a centralised manner, with communications outside the cloud environment limited to external systems such as the PBGM and the generation substations manager platform. The internal and external communication is carried out via MQTT, and REST API interfaces as illustrated in Figure 8.

The edgeFLEX databus collects the external data streams of measurements sent from the generation substations IoT manager platforms and stores them directly into the persistence database via MQTT. Furthermore, the edgeFLEX databus ensures that both the inertia estimation service and the PBGM policy loader can access the required data from the persistence database. The event listener in the PBGM loader monitors the grid state changes represented by the rate of change of frequency measurements, and based on these events and the corresponding policy, triggers the inertia estimation service accordingly.

As for the communication within the PBGM network, the PBGM policy loader requests the inertia estimation triggering policy defined in the PBGM system by the system operator via REST interface.

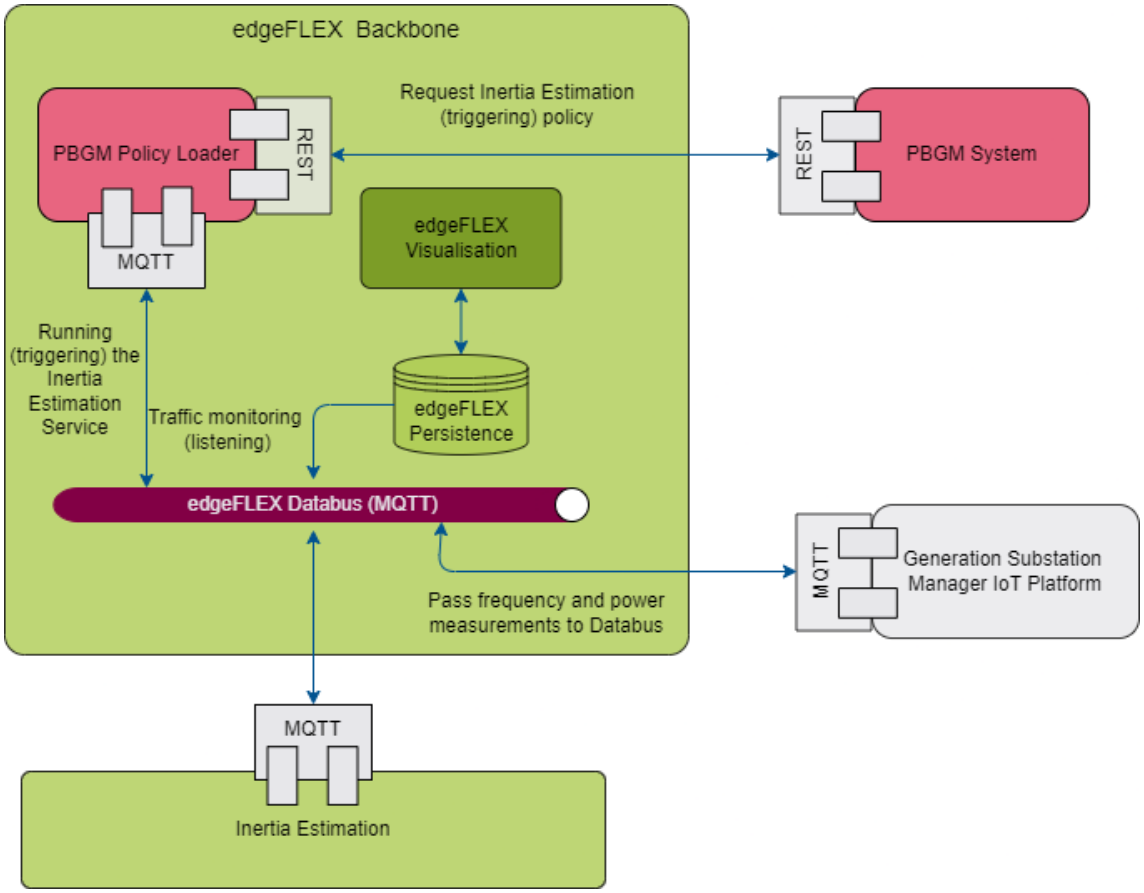


Figure 8 - Inertia Estimation

6.2 Frequency Control Service

As detailed in D2.2 and D4.1, the Frequency Control service developed within the edgeFLEX project operates as part of the edgeFLEX platform and is composed of three algorithms which each perform discrete operations for the VPP. A summary of these algorithms' operation aims is as follows:

- **VPP coordinated frequency control**: Aim is to improve the fast frequency response of the VPP. The control amplifies the sensitivity of fast frequency control in reference to local frequency deviation by a coefficient that is proportional to the power generated by the VPP.
- **VPP automatic generation control**: Aim is to restore the grid frequency to and keep the VPP power injection at the scheduled value. The control coordinates the frequency control loops of VPP resources proportionally to their droops.
- **Frequency regulation metering**: Aim is to implement a practical criterion to distinguish between devices which modify the frequency from those which do not. The provision of frequency regulation of the device is evaluated according to its corresponding Rate-of-Change-of-Power (RoCoP).

The implementation of the Frequency Control service within the edgeFLEX platform builds on the requirements gathered, architectures and interfaces defined in D4.1. In operation, the Frequency Control service interacts with the edgeFLEX backbone services and interfaces to support the operation of the algorithms. The edgeFLEX databus is utilised to publish and consume data to the service, for example capturing voltage and power readings from the field and publishing active power setpoints to assets. The edgeFLEX persistence enables the storage of values, setpoints and results, which may then be visualised by the edgeFLEX visualisation service.

The Frequency Control service also interacts with the PBGM system to enhance the operation of the service. With user-defined policies, the operational bounds of the Frequency Control algorithms can be updated and changed. These policies are passed to the Frequency Control via the PBGM policy loader which periodically analyses relevant policies and detects updates to the policy by a user. These updated operational bounds are then passed to the Frequency Control service and applied. The interactions between the Frequency Control service, edgeFLEX platform services, PBGM system and field assets is illustrated in Figure 9.

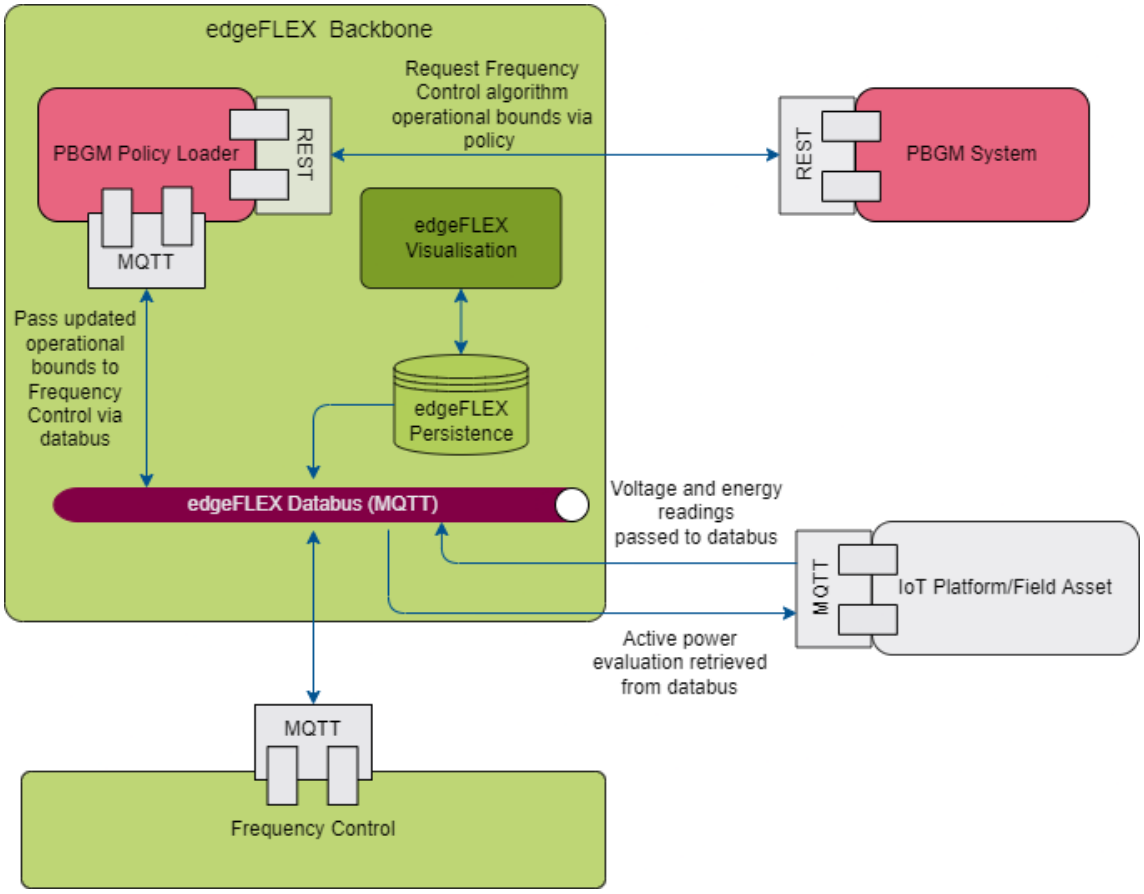


Figure 9 - Frequency Control Service & Platform Interfaces

6.3 Voltage Control Service

As described in D1.2, one of the main aspects of the voltage control service is to gather measurement data from the field and send control set-points to the controllable devices installed in the field. The service requires the connection to the MQTT broker of the platform to do this.

In addition to the interface with the MQTT broker, the control requires a proper classification of the MQTT topics, which depends on the type of data and device. When running the control in a simulated environment, interfaced with the powerflow service (also described in D1.2), the MQTT topics are divided into the publishing and subscribing topics:

The topics:

- /voltage_control/control/active_power
- /voltage_control/control/reactive_power
- /voltage_control/control/active_power_ESS

are used to publish the control set-points messages, so that the powerflow can subscribe to them and apply the control actions in the simulation.

The topics:

- /voltage_control/measuremnts/voltage
- /voltage_control/measuremnts/pv

are the ones to which the voltage control subscribes to receive the measurements from the simulation.

6.4 Model Predictive Voltage Control Service

The second voltage service is the Model Predictive Voltage Control for low voltage energy communities to deal with the local flexibility market. The full detail of the algorithm is presented in D1.3, however some basic information is provided here to understand the algorithm. The proposed Model Predictive Control (MPC) for Voltage Control represents the second drop of WP1, which has been developed for the integration of the Energy Communities in supporting the control of the voltage in distribution grid.

The control service integrates the inputs of the forecast to calculate the control outputs for a prediction horizon. The results of the MPC calculation are transmitted to the market platform as flexibility requests to perform the voltage control. However, the MPC is not directly interfaced with the market, since the requests need to be reformulated before submitting to the market. This is performed by the Policy Based Grid Management, which is the software element that links the two platforms.

Once the market has performed the calculation of the flexibility response, the output is then used by the MPC to track the reference values.

The JSON message sent by the MPC to the Policy Based Grid Management can be defined, for example as, as:

```
Flex_request = {{"variable": "variable", "nodes": [n1,n2,n3,…],
"flex_request":{"prediction 1":[k1,k2,k3,…], "prediction
2":[k1,k2,k3,…], "prediction 3":[k1,k2,k3,…], …},…}
```

The JSON message received by the MPC can be defined for example as:

```
Flex_response = {{"variable": "variable", "nodes": [n1,n2,n3,…],
"flex_response":{"prediction 1":[r1,r2,r3,…], "prediction
2":[r1,r2,r3,…], "prediction 3":[r1,r2,r3,…], …},…}
```

Where:

- `variable` : represents one of the possible controllable variable
- `nodes` : are the nodes for which the flexibility request have been defined
- `flex_request`: is the request specified for each prediction step
- `flex_response`: is the response specified for each prediction step

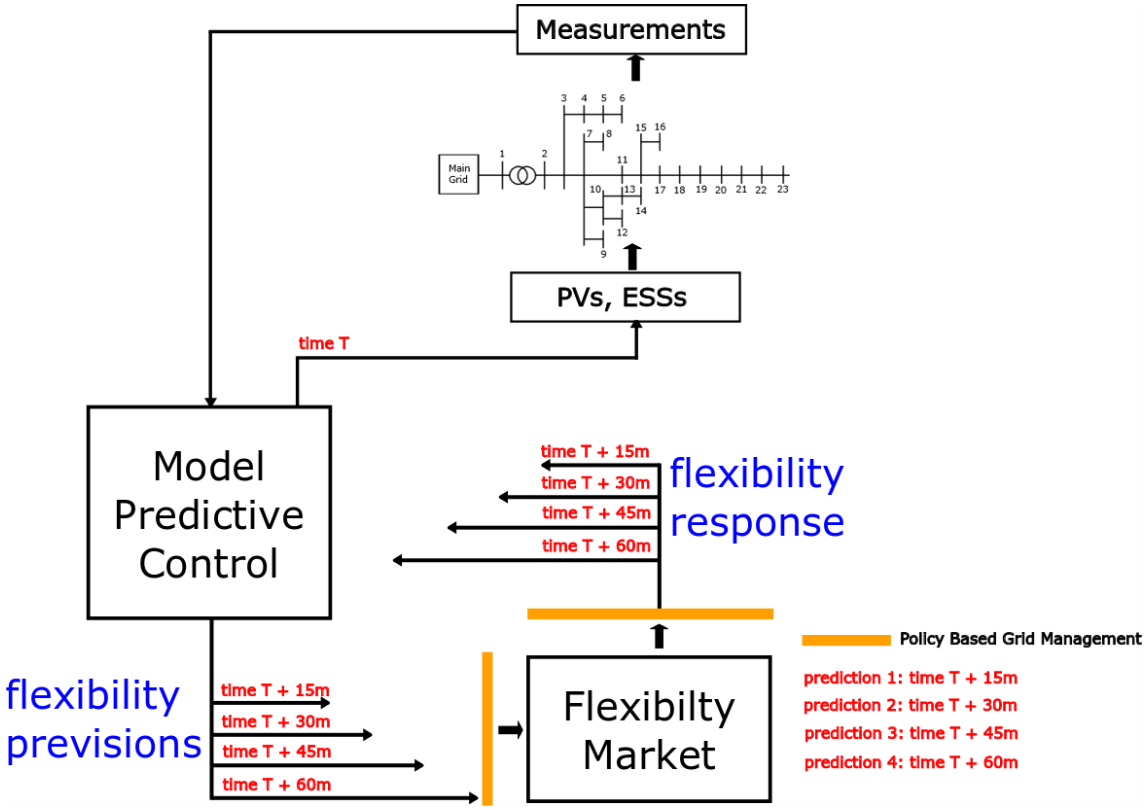


Figure 10: Graphical representation of the interface between the MPC and the Market

7. Using external data streams as control service interfaces

A core element of the edgeFLEX platform is its modular and portable design, whereby components can be taken and be deployed as single services or in tandem with others. This is enabled by the flexible design of the components, and part of this design is in how they can be dynamically configured when they are instantiated. These configurations serve the purpose of setting up the service to identify the input data needed, the origin of input data, and where the outputs of the service are sent to. This means that while the control services are built to get data from the edgeFLEX backbone, they can equally be configured to access data from existing sources, like message brokers, APIs, or databases in the systems where they are deployed.

There are two methods of configuration used within the edgeFLEX platform, one using docker environment variables at runtime and the other using the PBGM Policy API that dynamically requests and updates the policy when the service is running. Using the docker environment, as illustrated in Figure 11, involves the injection of a configuration file containing the required configuration details, and when the code is run, the details are inputted into the queries and connections to the input sources. While this is a flexible means of configuring the service, and both more robust and secure than hard coding the connection strings, it does not update. To do so, the service will require a service reboot to enact a configuration change.

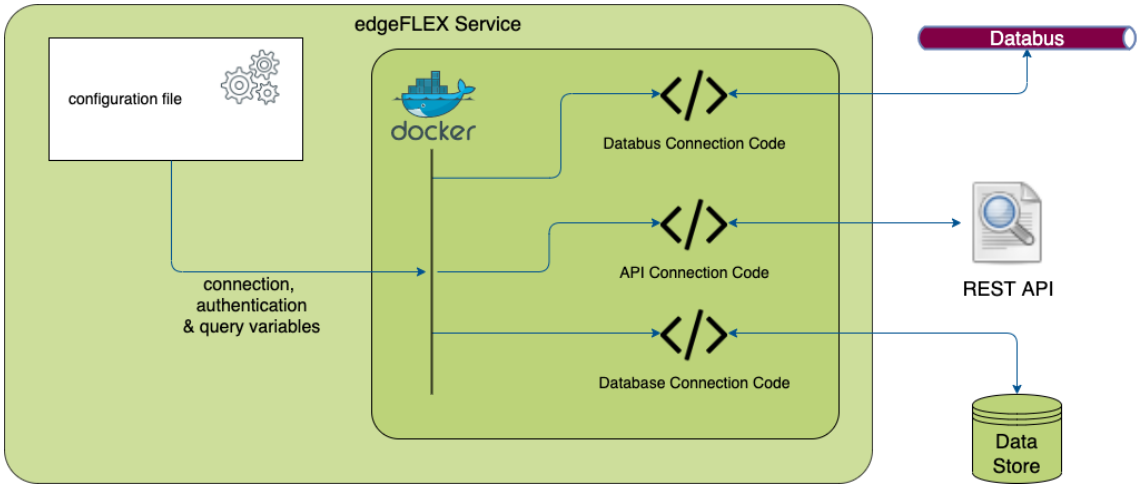


Figure 11 - Interface Configuration using Docker Environment

The PBGM solution, as illustrated in Figure 12, while still using a configuration file inputted at run time, takes the main configuration details for connecting to data sources from a policy API that is exposed by a cloud hosted PBGM instance. The configuration file in this configuration method only contains configurations needed to connect to a cloud deployed instance of PBGM and a list of the policies needed for the service operate. This configuration file also contains a refresh rate that will direct the policy loader deployed in the service to poll and update the configurations periodically. What this means is that while the service is running, the code can be configured to switch data sources or read different input variables from the data bus, the data store or from REST APIs. While this is a dynamic method of configuration, it relies on connection to the PBGM instance and may not be a viable configuration method if deployed in a system with limited outward connectivity.

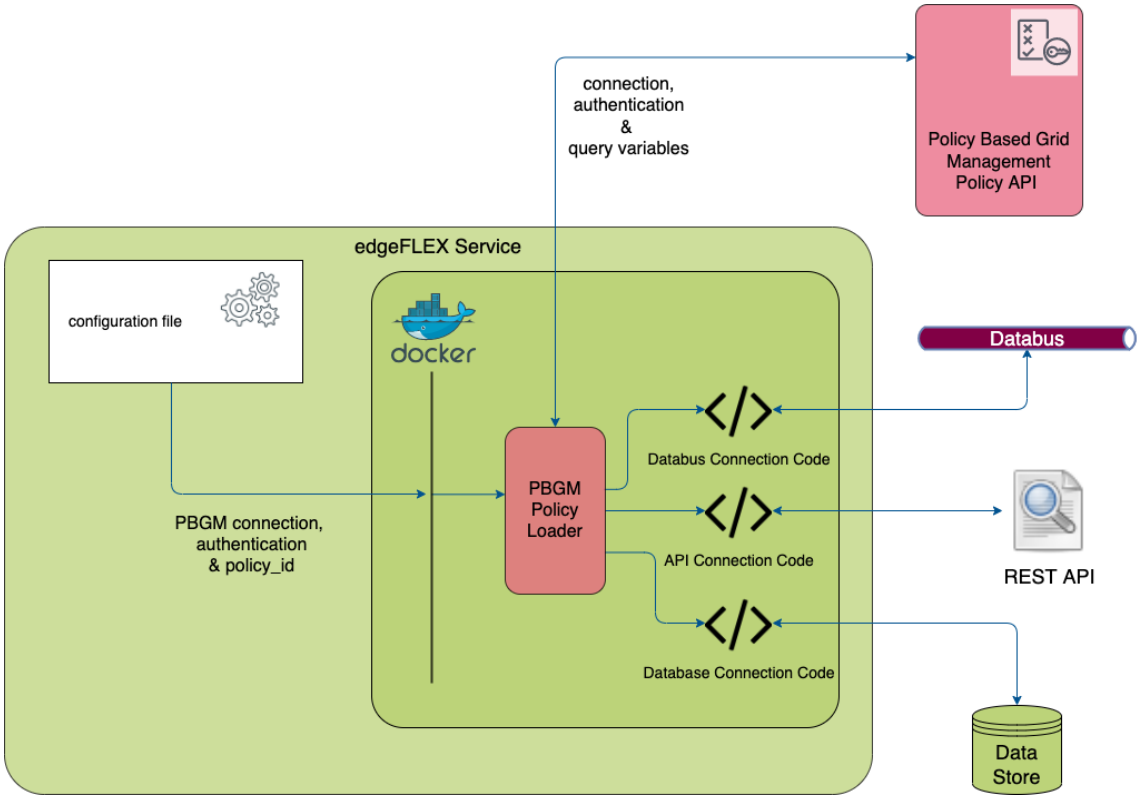


Figure 12 - Interface Configuration using PBGM

An example of how this works and how the services are configured is illustrated in Figure 13, which is representative of the architecture that forms part of the trial in Wunsiedel where the Voltage Control service and the edgeFLEX Backbone require data from the KIBERNet system, the existing data management platform, and the edgePMUs that are deployed onsite. The instantiation of the platform in this case uses both configuration methods, the PBGM driven method with a data management policy for accessing data from the KIBERNet system, and the Docker Environment configuration method for connecting to the databus, from which the edgePMU readings will be received, and to which the data will be sent for the storage of the outputs.

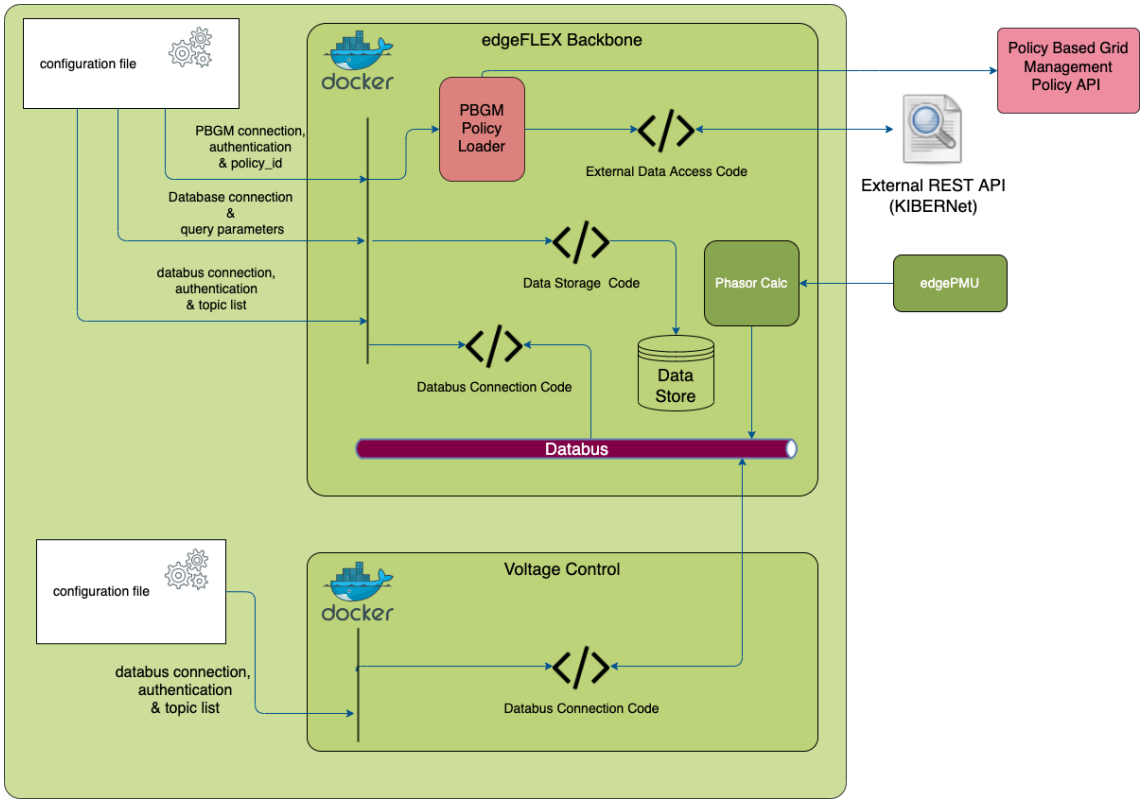


Figure 13 - Hybrid Service Configuration

While Figure 13 shows the data store and the databus being deployed as part of the platform, it must be noted that these services could already exist within the systems on site and in that case the configurations would be such as to connect to them. This flexibility of configuration allows the control service interfaces to connect to internal and external sources, meaning that the components of the platform can use external data streams, data stores and data serving APIs to be used as data for the control services.

8. 5G features providing secure, reliable and managed data sharing

In this chapter, a new 5G device management capabilities exposure API, termed the “5G Device Management API”, is introduced with the focus on its benefits to the edgeFLEX services and power system operators. Furthermore, its features addressing the requirements of the edgeFLEX services and enablers are described in detail in a table. To give a better understanding of the use of 5G Device Management API with the other components of edgeFLEX platform, an exemplary deployment is drawn similar to the diagrams drawn in Section 2 - edgeFLEX Architecture Description.

The 5G Device Management API is a new interface that is continuously being defined and partially standardised. The 5G Device Management API from Ericsson is the world’s first design and implementation of such an interface as a proof-of-concept. This new interface is designed to expose the device management capabilities of 5G networks openly to the users from any industrial sector, including power system operators. It allows authorised users to be able to directly manage the connectivity of their devices, to monitor the communication status of these devices remotely and to check the quality of the communication links created per device. Furthermore, it allows users to create isolated device groups and easily change the members of these groups remotely. All these operations can be performed through requests sent to the 5G Device Management API. Then, the API retranslates and forwards the requests to the corresponding network functions to execute the desired operation in the 5G network. As a result, the user can use an interface developed according to its needs to manage device connectivity related tasks on its own without the need to request them from a mobile network operator, saving time and effort for the user.

An example of a deployment option including the 5G Device Management API and other edgeFLEX platform components is illustrated in Figure 14. In Figure 13, an authorised user can use the 5G Device Management API to establish the communications link between edgePMU device and the voltage control service illustrated with red lines. The requests being sent from the user to the 5G Device Management API are illustrated with blue dashed lines, as they are the control messages rather than the actual data traffic. Then, these requests are forwarded to the relevant 5G network functions in the 5G network to establish the communications link for edgePMU devices and to enable data transfer from edgePMU device to the edgeFLEX databus and to the voltage control service.

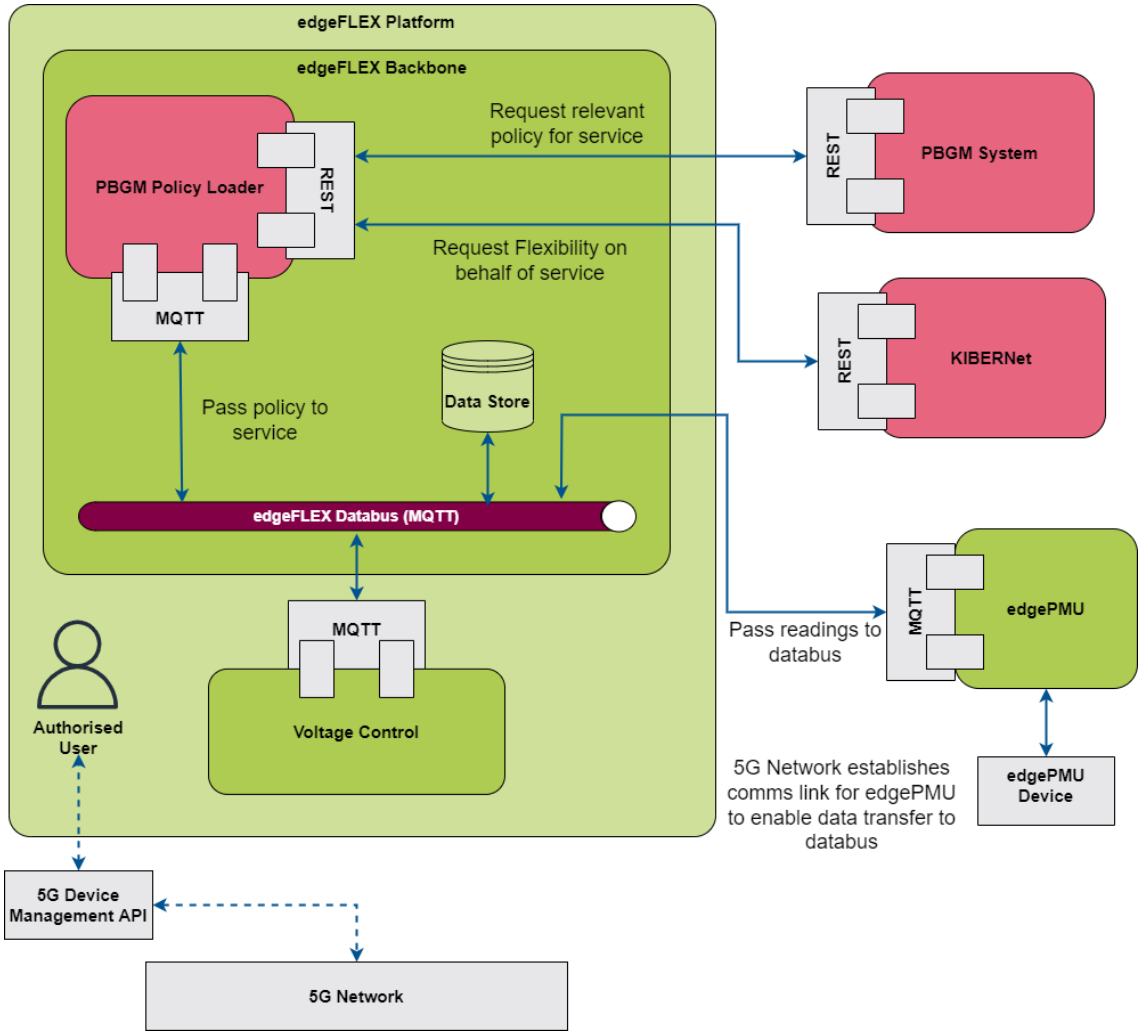


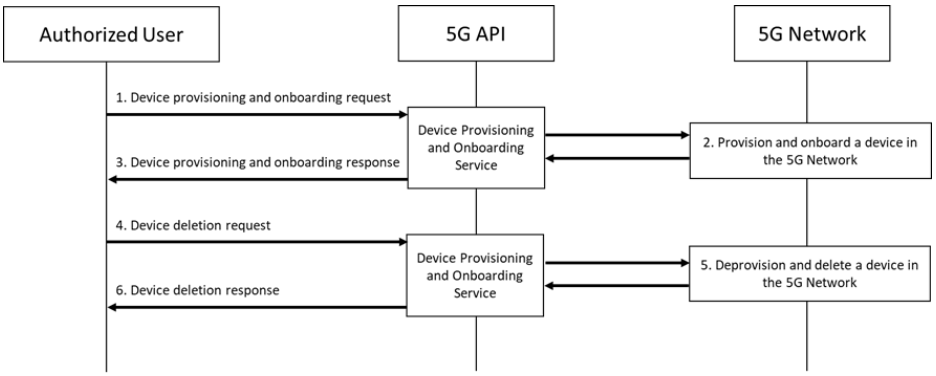
Figure 14 - 5G Device Management API with edgeFLEX platform

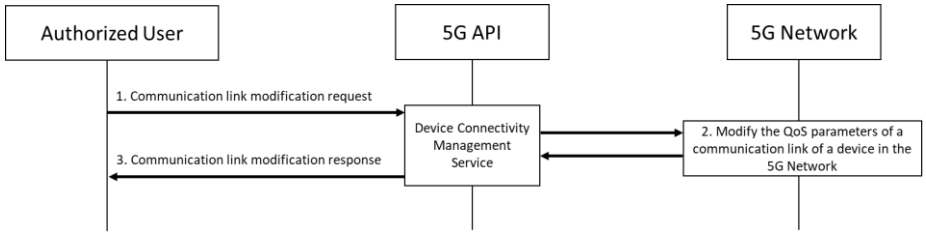
The figure above shows only one of the deployment options that is suitable to be used by a DSO. However, the 5G Device Management API can support all edgeFLEX services and actors:

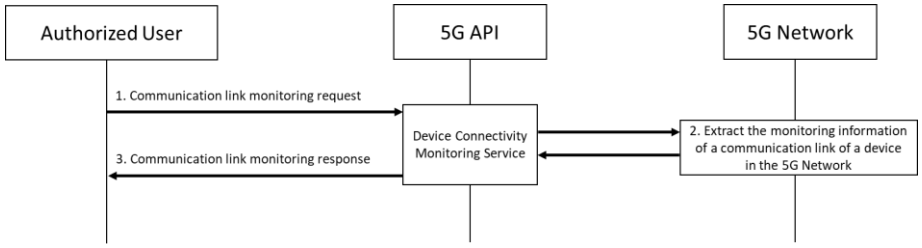
- DSO
  - Voltage control
- TSO
  - Inertia estimation
  - Frequency regulation metering
- VPP
  - VPP optimisation
  - Wind forecasting
  - Advanced energy flexibility aggregation and trading system
  - VPP coordinated frequency control
  - VPP automatic generation control

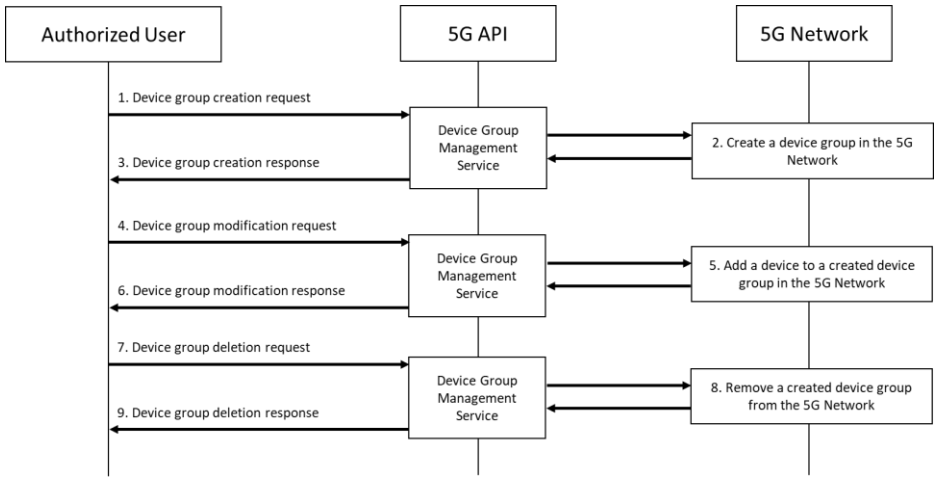
The 5G Device Management API features listed below can serve all edgeFLEX services and actors listed above. In the following list, the relevant and important features of the 5G Device

Management API [9] are given as well as the common requirements of the edgeFLEX services and enablers.

Requirement 1: Provision and onboard field devices to the 5G network remotely	
API Feature	Device Provisioning and Onboarding
Actors	• User from the power system operator • Field device connected to a 5G user equipment • 5G network • Device provisioning and onboarding service of the API
Process	The features of the 5G Device Management API can be classified as the "services" of the API, which are responsible for executing different operations requested by the users. The 5G Device Management API service that is responsible for provisioning and onboarding devices is called "device provisioning and onboarding service". The device provisioning and onboarding requests initiated by the users are received by this service. For this specific process, it is required that the device has the correct certificates available in place that will enable the successful authentication of the device in the 5G network, e.g., user has a card provided by the mobile operator to be inserted to the device or user downloaded the credentials and certificated from the mobile operator. • The authorised user sends a request to the device provisioning and onboarding service to provision and onboard a device to the 5G network. This request would include the unique identifier of the device such as GPSI number provided by the mobile operator. • The device provisioning and onboarding service forwards the request provided by the user to the 5G network. • The device provisioning and onboarding service sends a reply to the user indicating the success or failure of the provisioning and onboarding operation of the device. • The authorised user sends a request to the device provisioning and onboarding service to delete the device that is connected to the 5G network. • The device provisioning and onboarding service forwards this deletion request to the 5G network. • The device provisioning and onboarding service sends a reply to the user indicating the success or failure of the operation of removing the onboarded device from the 5G network.
Sequence diagram	

Result	• The field device is successfully provisioned and onboarded to the 5G network. • The field device has a default communication link created in the 5G network. • The field device is removed from the 5G network.
Requirement 2: Change the Quality-of-Service (QoS) parameters of the communication links of field devices	
API Feature	Device Connectivity Management
Actors	• User from the power system operator • Field device connected to a 5G user equipment • 5G network • Device connectivity management service of the API
Process	The features of the 5G Device Management API can be classified as the “services” of the API, which are responsible for executing different operations requested by the users. The 5G Device Management API service that is responsible for changing the QoS parameters of the communication links of devices is called “device connectivity management service”. The modification requests of the communication links initiated by the users are received by this service. • The authorised user sends a request to the device connectivity management service to modify the QoS parameters of the communication link of a device. • The device connectivity management service forwards the requested QoS parameter values provided by the user to the 5G network. • The device connectivity management service sends a reply to the user indicating the success or failure of the modification of the communication link operation.
Sequence Diagram	
Result	• The QoS parameters of a communication link of the field device is changed to higher/lower values in the 5G network. Thus, the communication link of the field device is optimized and set up according to the needs of the edgeFLEX service.
Requirement 3: Monitor the quality and status of the communication link of field devices	
API Feature	Device Connectivity Monitoring
Actors	• User from the power system operator • Field device connected to a 5G user equipment • 5G network • Device connectivity monitoring service of the API

Process	The features of the 5G Device Management API can be classified as the “services” of the API, which are responsible for executing different operations requested by the users. The 5G Device Management API service that is responsible for monitoring the quality and status of the communication link of devices is called “device connectivity monitoring service”. The monitoring requests of the communication links initiated by the users are received by this service. • The authorised user sends a request to the device connectivity monitoring service to monitor the status of the communication link for a device, that is already connected to the 5G network. This request would include the specific parameters that are desired to be monitored. • The device connectivity monitoring service forwards the request provided by the user to the 5G network. • The device connectivity monitoring service receives the monitoring information from the 5G network and provides this information to the user indicating the values for the requested monitoring parameters.
Sequence Diagram	
Result	• The field device has a communication link created in the 5G network. • The communication status of this link is observable to the authorised user from the power system operator. • The monitoring of e.g., QoS parameters of the device, the current reference signal received power etc. can be read by the authorised user.
Requirement 4: Create, modify, and remove device groups	
API Feature	Device Group Management
Actors	• User from the power system operator • Field device connected to a 5G user equipment • 5G network • Device group management service of the API
Process	The features of the 5G Device Management API can be classified as the “services” of the API, which are responsible for executing different operations requested by the users. The 5G Device Management API service that is responsible for creating, modifying, and removing of device groups is called “device group management service”. The group modification requests initiated by the users are received by this service. • The authorised user sends a request to the device group management service to create a device group. • The device group management service provides the device group data provided by the user to the 5G network. • The device group management service sends a reply to the user indicating the success or failure of the device group creation operation.

	In case of a success, the response will contain a unique device group identifier. • The authorised user sends a request to the device group management service to modify the members of a device group. • In case the request is about adding a device to a group, the user request will contain the device identifier and the unique device group identifier of the group to which the device will be added as a member. • The device group management service forwards this modification request to the 5G network. • The device group management service sends a reply to the user indicating the success or failure of the operation of adding a device to a group. • In case the request is about deleting the device group from the 5G network, the user request will contain the unique device group identifier of the group. • The device group management service forwards this modification request to the 5G network. • The device group management service sends a reply to the user indicating the success or failure of the operation of deleting the device group from the 5G network.
Sequence Diagram	
Result	• The device group is defined and created in the 5G network. • The field device is added as a member to the device group. • The device group is removed from the 5G network.

All of the features of the 5G Device Management API described above have been selected according to the needs of the edgeFLEX services. For example, a DSO having many devices as the data sources in the grid can onboard its devices to the 5G network whenever the measurements need to be collected from a particular area. In addition, the DSO can define various device groups with different purposes and add its devices to these groups. As a result, devices serving the same purpose can be managed as a group.

As a result, 5G Device Management API will address many requirements of the energy use cases and will make 5G easier to use for users in energy sector. With the combination of other edgeFLEX platform components such as Policy Based Grid Management, the 5G Device Management API can become an enabler for more secure and reliable data sharing between devices and services. The communication links can be defined and changed faster thanks to the use of 5G as a wireless technology, but also the use of this API, as the connectivity can be managed by sending simple requests to the API. The outcome of these exemplary use cases will

show that the power system operators will be able to quickly start using 5G without spending time learning about the details of 5G networks. Furthermore, the transition to the digitalisation will be smoother for the power system operators, as they can execute operations by using an easy-to-use interface from 5G.

9. Conclusion

The goal of this deliverable is to describe to the reader the edgeFLEX platform components and services in the context of interfaces, and how these interfaces allow for the sharing of data both within the platform itself and with external entities. These interfaces are a core component which enable the operation of the edgeFLEX platform and how it will advance the role of the VPP. This work builds on the earlier requirements gathering tasks in Phase 1 which lead to the development of an architecture for the edgeFLEX platform and services as well as functional and non-functional requirements based on input gathered from work packages WP1, WP2 and WP3 and the field trials in WP5.

As described in Section 2, the edgeFLEX platform architecture is one with modularity in mind, providing the means for the System Operator to deploy the platform components they need, in the manner which best matches their operating environment and specific use case. Three potential examples were given, centralised, de-centralised and hybrid mobile-edge deployment, intended to demonstrate the flexibility of the platform components and how the outputs from Phase 1 of the project allowed for the design and implementation of the data interfaces which enable this modularity and flexibility in terms of the edgeFLEX platform deployment. Section 3 details how the interfaces were implemented, and how the use of modern, standardised technologies and protocols such as MQTT and REST, enhance and grant the range of functionality needed for the interfaces to enable the platform to operate in a number of operating environments.

With more industries and utilities expanding their use of ICT and IoT technologies to give greater vision and control of their operations, security is of ever increasing concern. Section 4 details the security considerations made when designing and implementing the edgeFLEX platform interfaces and how they are secured within the contexts of data access, sharing, networking and communication between components. Key to this is the implementation of the PBGM system, described in Section 6. The PBGM system plays a key role in controlling the exposure of the interfaces, managing data sharing and facilitating connection to external data interfaces such as flexibility trading platforms. By providing a means for the SO to define and implement their own policies via the PBGM system, this gives a level of fine-grained control for data sharing and access which can be modified at any time by the user as required.

Finally, Section 8 of the report describes how the use of 5G, in the form of the 5G Device Management API can enable more reliable, secure and fast communication for the edgeFLEX services. This is demonstrated by the use cases described in this section, which detail the advantages to the SO in adopting 5G in terms of managing and scaling devices and data sources, and how the features of the 5G API make the transition to digitisation smoother for the SO via user-friendly interfaces.

For the remainder of the project, the flexibility and modularity of the platform will be demonstrated through the edgeFLEX platform deployments and field trials. At the time of writing, the edgeFLEX platform and PBGM system have been deployed in the cloud and are communicating with the KIBERNet system based on defined policies, demonstrating that the interfaces are working as expected. All outputs from these trials will be used to enhance and improve the edgeFLEX platform according to the edgeFLEX improvement model as defined in Phase 1 with an aim to demonstrate the value the project has in advancing the role of the VPP and facilitating a more dynamic grid. In addition, the descriptions of the functionality and interfaces described, including the users, actors and processes within this document, will provide a relevant source of information to further business modelling work in WP6.

10. Bibliography

[1] “MQTT: The Standard for IoT Messaging,” [Online]. Available: https://mqtt.org/. [Accessed 7 January 2022].

[2] “The Complete MQTT Broker Selection Guide,” [Online]. Available: https://www.catchpoint.com/network-admin-guide/mqtt-broker. [Accessed 7 January 2022].

[3] “What is a REST API?,” 8 May 2020. [Online]. Available: https://www.redhat.com/en/topics/api/what-is-a-rest-api. [Accessed 7 January 2022].

[4] “What Is User Datagram Protocol (UDP)?,” 22 December 2020. [Online]. Available: https://www.emnify.com/iot-glossary/udp. [Accessed 7 January 2022].

[5] J. Z. F. L. S. R. W. a. Z. Y. D. G. Liang, “A Review of False Data Injection Attacks Against Modern Power Systems,” *IEEE Transactions on Smart Grid*, vol. 8, no. 4, pp. 1630-1636, 2017.

[6] S. M. G. X. a. D. Y. X. Fang, “Smart Grid — The New and Improved Power Grid: A Survey,” *IEEE Communications Surveys & Tutorials*, vol. 14, no. 4, pp. 945-947, 2012.

[7] A. A. H. M. a. B. D. A. Liu, “Iotverif: Automatic Verification of SSL/TLS Certificate for IoT Applications,” *IEEE Access*, vol. 9, pp. 27038-27050, 2021.

[8] J. H. M. a. J. M. S. I. Adam, “RESTful Web Service Implementation on Unklab Information System Using JSON Web Token (JWT),” *2020 2nd International Conference on Cybernetics and Intelligent System (ICORIS)*, pp. 1-6, 2020.

[9] 5GACIA, “Exposure of 5G Capabilities for Connected Industries and Automation Applications,” ZVEI – German Electrical and Electronic Manufacturers’ Association, Frankfurt am Main, 2021.

11. List of Figures

12. List of Abbreviations

ADM	Architecture Domains Methodology
API	Application Programming Interface
ATP	Automated Trading Platform
CSV	Comma-separated values
DB	Database
DER	Distributed Energy Resource
DG	Distributed Generator
DMS	Data Management System
DOMS	Distribution Observability and Management System
DSO	Distribution System Operator
EEX	European Energy Exchange
EMS	Energy Management System
ESS	Energy Storage System
FA	FlexAgent
FMAN	Flexibility Management
FMAR	Flexibility Market
FO	FlexOffer
FOA	FlexOffer Agent
FTP	Flexibility Trading Platform
GMAN	Grid Management
GUI	Graphical User Interface
HEMRM	Harmonised Electricity Role Model
HTML	HyperText Markup Language
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure
ICT	Information and Communication Technology
IoT	Internet of Things
IP	Internet Protocol
IEEE	Institute of Electrical and Electronics Engineers

JPEG	Joint Photographic Experts Group
JSON	JavaScript Object Notation
KPI	Key Performance Indicator
LEC	Local Energy Community
LTE	Long Term Evolution
MBA	Market Balance Area
MQTT	Message Queue Telemetry Transport
MVP	Minimum Viable Product
PBNM	Policy based Network Management
PDF	Portable Document Format
PMU	Phasor Measurement Unit
PV	Photovoltaic
RES	Renewable Energy Source
REST	Representational state transfer
ROCOF	Rate of Change of Frequency
ROCOP	Rate of Change of Power
SLA	Service Level Agreement
SO	System Operator
SSL	Secure Sockets Layer
TCP	Transmission Control Protocol
TOGAF	The Open Group Architecture Framework
TSO	Transmission System Operator
UDP	User Datagram Protocol
UE	User Equipment
VNF	Virtual Network Function
VPN	Virtual Private Network
VPP	Virtual Power Plant
VPS	Virtual Power System
WP	Work Package
XML	Extensible Markup Language