Platone

PLATFORM FOR OPERATION OF DISTRIBUTION NETWORKS

Platone

PLATform for Operation of distribution NEtworks

D2.8

Platone DSO Technical Platform (v3)

The project PLATform for Operation of distribution NEtworks (Platone) receives funding from the European Union's Horizon 2020 research and innovation programme under Grant Agreement no 864300.

Project name	Platone
Contractual delivery date:	30/06/2023
Actual delivery date:	29.06.2023
Main responsible:	Siemens SPA
Work package:	WP2 – Platform Implementation and Data Handling
Security:	P
Nature:	DEM
Version:	V1.0
Total number of pages:	18

Abstract

The Platone Open Framework aims to create an open, flexible, and secure system that enables distribution grid flexibility/congestion management mechanisms, through innovative energy market models involving all the possible actors at many levels (DSOs, TSOs, customers, aggregators). The Platone Framework is an open-source framework based on blockchain technology that enables a secure and shared data management system, allows standard and flexible integration of external solutions (e.g. legacy solutions), and is open to integration of external services through standardized open Application Program Interfaces (APIs).

This document accompanies the third software delivery of the Platone DSO Technical Platform providing an architecture overview and a description of the additional services developed to enable the use cases from the demo sites.

Keyword list

Platone DSO Technical Platform, Platone Framework, Open Source, micro-service, control centre, Kubernetes

Executive Summary

"Innovation for the customers, innovation for the grid" is the vision of project Platone - Platform for Operation of distribution Networks. Within the H2020 programme "A single, smart European electricity grid", Platone addresses the topic "Flexibility and retail market options for the distribution grid". Modern power grids are moving away from centralised, infrastructure-heavy Transmission System Operators (TSOs) towards Distribution System Operators (DSOs) that are flexible and more capable of managing diverse renewable energy sources. DSOs require new ways of managing the increased number of producers, end users and more volatile power distribution systems of the future.

Platone is using blockchain technology to build the Platone Open Framework to meet the needs of modern DSO power systems, including data management. The Platone Open Framework aims to create an open, flexible and secure system that enables distribution grid flexibility/congestion management mechanisms, through innovative energy market models involving all the possible actors at many levels (DSOs, TSOs, customers, aggregators). It is an open source framework based on blockchain technology that enables a secure and shared data management system, allows standard and flexible integration of external solutions (e.g. legacy solutions), and is open to integration of external services through standardized open Application Program Interfaces (APIs). It is built with existing regulations in mind and will allow small power producers to be easily certified so that they can sell excess energy back to the grid. The Platone Open Framework will also incorporate an open-market system to link with traditional TSOs. The Platone Open Framework will be tested in three European demos and within the Canadian Distributed Energy Management Initiative (DEMI).

This Deliverable describes the third release of the Platone DSO Technical Platform and accompanies the related software delivery; this third version follows the consolidated development guidelines built for the previous versions evolving both on interfaces, communication mechanisms and internal services adapting to the requirement of the use cases tested in the demo sites.

Authors and Reviewers

Main responsible		
Partner	**Name**	**E-mail**
SIEMENS		
	Brunella Conte	brunella.conte@siemens,com
	Carlo Adriano Arrigoni	carlo.arrigoni@siemens.com
Author(s)/contributor(s)		
Partner	**Name**	
RWTH		
	Florian Oppermann	florian.oppermann@eonerc.rwth-aachen.de
	Juan Galeano	adolfo.galeano@eonerc.rwth-aachen.de
ENG		
	Ferdinando Bosco	Ferdinando.bosco@eng.it
Reviewer(s)		
Partner	**Name**	
AVACON		
	Benjamin Petters	benjamin-georg.petters@avacon.de
ENG		
	Ferdinando Bosco	ferdinando.bosco@eng.it
Approver(s)		
Partner	**Name**	
RWTH		
	Amir Ahmadifar	aahmadifar@eonerc.rwth-aachen.de

Table of Contents

1 Introduction

The project “PLATform for Operation of distribution Networks – Platone” aims to develop an architecture for testing and implementing a data acquisition system based on a two-layer Blockchain approach: an “Access Layer” to connect customers to the Distribution System Operator (DSO) and a “Service Layer” to link customers and DSO to the Flexibility Market environment (Market Place, Aggregators, …). The two layers are linked by a Shared Customer Database, containing all the data certified by Blockchain and made available to all the relevant stakeholders of the two layers. This Platone Open Framework architecture allows a greater stakeholder involvement and enables an efficient and smart network management. The tools used for this purpose will be based on platforms able to receive data from different sources, such as weather forecasting systems or distributed smart devices spread all over the urban area. These platforms, by talking to each other and exchanging data, will allow collecting and elaborating information useful for DSOs, transmission system operators (TSOs), Market, customers and aggregators. In particular, the DSOs will invest in a standard, open, non-discriminatory, blockchain-based, economic dispute settlement infrastructure, to give to both the customers and to the aggregator the possibility to more easily become flexibility market players. This solution will allow the DSO to acquire a new role as a market enabler for end users and a smarter observer of the distribution network. By defining this innovative two-layer architecture, Platone strongly contributes to aims to removing technical and economic barriers to the achievement of a carbon-free society by 2050 [1], creating the ecosystem for new market mechanisms for a rapid roll out among DSOs and for a large involvement of customers in the active management of grids and in the flexibility markets. The Platone platform will be tested in three European demos (Greece, Germany and Italy) and within the Distributed Energy Management Initiative (DEMI) in Canada. The Platone consortium aims to go for a commercial exploitation of the results after the project is finished. Within the H2020 programme “A single, smart European electricity grid” Platone addresses the topic “Flexibility and retail market options for the distribution grid”.

The overall Platone architecture is based on the Platone Open Framework (Figure 1), for more details refer to the previous Deliverables D2.6 [2] and D2.7 [3].

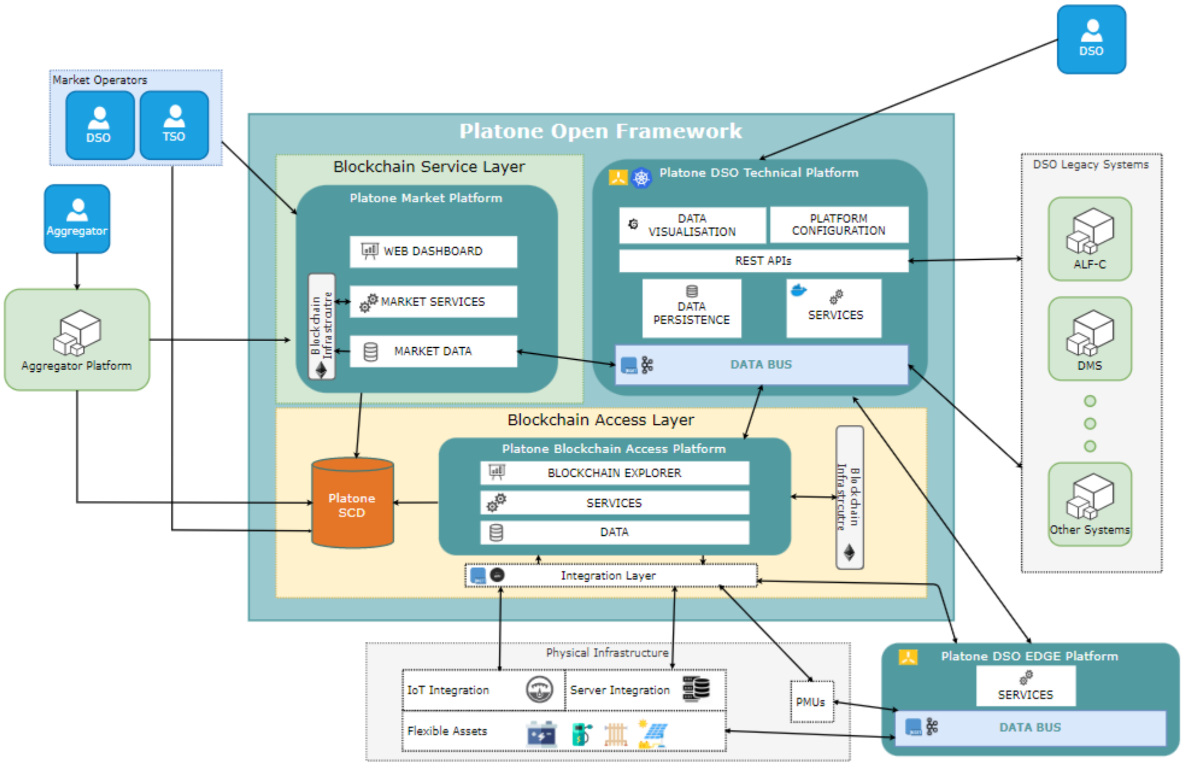

Figure 1 Platone Open Framework

This Deliverable describes the third version of the Platone DSO Technical Platform adding on the already existing platform the following:

- Greek demo State Estimation: transition from Matlab to Octave and visualization dashboard development
- Power Calculation Service extension: inclusion of voltages, currents and power values on the three phases; improvement of connection stability
- New plug-in to calculate the average values for current, voltage and power coming from the Power Calculation service
- Balancer Optimizer improvement to enable multiple optimization requests independently and with multiple flexibility providers (i.e. including also household storages) adapting also to the German demo uses cases
- Kafka Databus implementation and integration with all existing and new services
- MQTT (Message Queuing Telemetry Transport) Bridge adjustment to enable publishing of results to outside receivers (e.g. Azure IoT gateway)

1.1 Task 2.3

This Deliverable is related to Task 2.3 that is connected to the implementation of a DSO Technical Platform able to fulfil market requests and activating local flexibility requests while ensuring the reliability of the electrical network.

1.2 Objectives of the Work Reported in this Deliverable

The Platone description of action defines this Deliverable as a demonstrator; the objective of this document is to present the third release of the Platone DSO Technical Platform architecture and components accompanying the code repository.

1.3 Outline of the Deliverable

The second Chapter of this document describes the third version of the Platone DSO Technical Platform starting from the second implementation (refer to Deliverable D2.7 [3]) focusing on the new services, interfaces and communication mechanisms. Chapter 3 delivers a compilation of Languages, Technologies and External Tools used throughout the Platform. Chapter 4 is about the software delivery and provides the guidelines about installation, setup and configuration; Chapter 5 concludes this Deliverable.

1.4 How to Read this Document

This document reports the software delivery of the third release of the Platone DSO Technical Platform, which is part of the Platone Open Framework implemented within WP2 of Platone. Further information on the first and second releases are available in the Platone Deliverable D2.6 [2] and D2.7 [3]. For more insights on the open framework and on the requirements for the platform, refer to D2.1 [4] and D2.2 [5].

2 Platform updates

2.1 Architecture

The Platone DSO Technical Platform design builds on previous work done in the Horizon 2020 project SOGNO [6] and relies massively on a micro-service architecture.

The presented platform architecture aims at facilitating the transition to a modular, micro-services-based control center software solution for Distribution System Operators (DSO). The used architecture allows for a faster adjustment and independent development and deployment of components, which benefits from the isolation of services to improve maintenance preventing cascading failures between different components.
To address requirements such as high availability, scalability, and modularity from the very beginning, the DSO Technical Platform is designed to be deployed on Kubernetes [7] clusters. As a requirement, for all micro-services to be part of the platform, containerization using Docker [8] containers is required; this approach enables an easy deployment on a Kubernetes cluster. Among the benefits of using Kubernetes is the simplification of different deployment approaches: from edge- and public-cloud to on premise installation. Figure 2 illustrates the architecture of the DSO Technical Platform. The Databus is one of its core components and is implemented by means of a message broker to which all services can publish and subscribe in order to exchange data. This data exchange can be done internally between micro-services or externally with field devices and other platforms or systems.

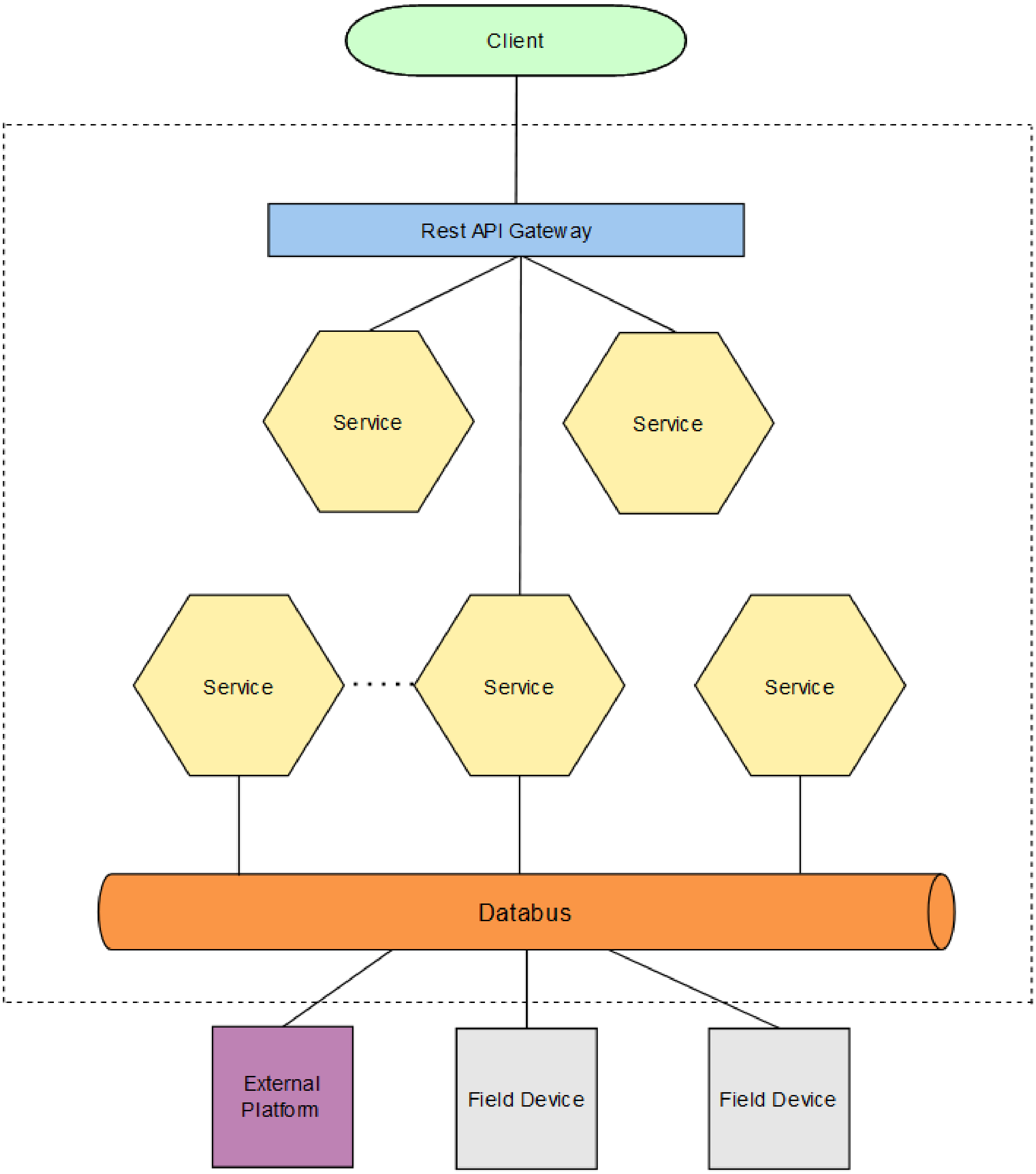

Figure 2 DSO Technical Platform architecture

2.2 Services

State Estimation

For the state estimation developed by the Greek demonstrator partners, the transition from Matlab to Octave was started as the initial steps for the deployment of this service. The change from MatLab to Octave defeats the requirement for a MatLab license. Initial testing indicates that this transition will be possible without much implementation overhead. Methods unavailable in Octave such as the string function “contains” were added. An initial version of the visualisation dashboard for the state estimation was developed, too. A graph representation of the network is generated from CIM data, which is the data format provided by the Greek demo. An initial static graph layout is created using a tool based on XML parsing and the NetworkX package [9]. The latter provides multiple layout methods including the one chose, the spring layout. This initial layout was imported into a custom react app using react-flow [10]. This will enable an intuitive access to the results of state estimation by giving an overview of the network as a whole. The network display will be interactable to allow navigation directly from the overview to detailed information about each node. The network structure is processed during build time and only the results are used during runtime while measurement data is fed in continuously via MQTT. The new components can be seen in green in Figure 3. The already existing components of the DSOTP are shown in Yellow.

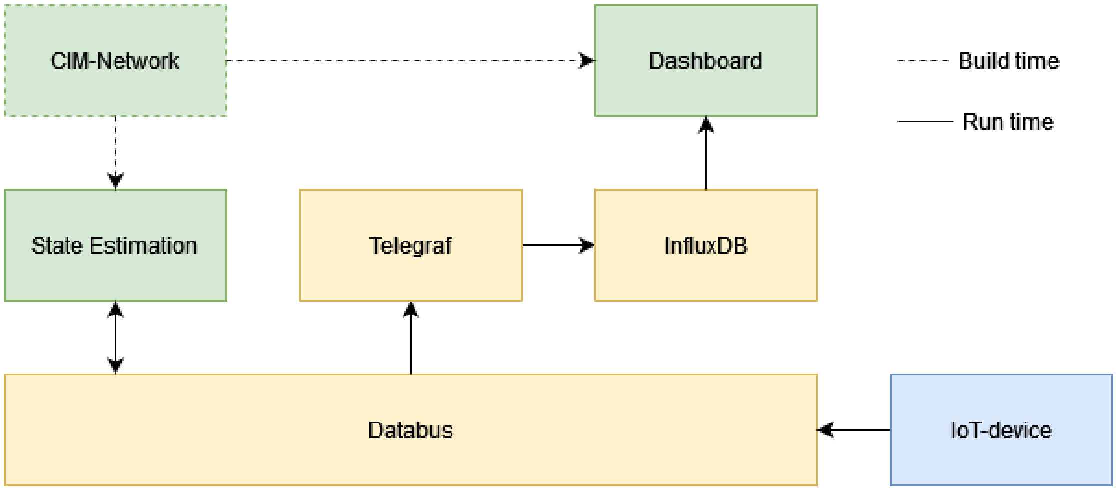

Figure 3 Deployment structure of the state-estimation tool

Power Calculation

The goal of the Power Calculation service is to enrich measurement data from Phasor Measurement Unit (PMU) field devices with additional power quantities; in this third release the Power Calculation service was extended and the processed messages now include the apparent Power for each of the three phases. Furthermore, the service also adds the total values of all three phases for current and voltage as well as for the power values. Additionally, the connection stability was improved by means of configuring the MQTT client.

Telegraf

A plug-in called “basic stats” [11] was added to calculate the average for current, voltage and power values coming from the power calculation service. This plug-in is used to calculate the averages of these values. This decouples reporting and measuring resolution and reduces storage needs if used.

Balancer Optimizer

With respect to the final version of the python microgrid flexibility management (referred to as pymfm) service which balances the total generation and consumption within community microgrids by means of optimization and rule-based control logics, the following updates have been conducted. FastAPI has been incorporated as a framework for the implementation of the REST (REpresentational State Transfer) API. Among the benefits that can be gained by the use of this framework are the easiness and order to read the code, the automatic documentation of endpoints, validation of the input format. The automatic generation of OpenAPI specifications and User Interface (UI) specifically improves the development work-flow and testing as well. The API has also been adjusted to follow the REST standard more closely. In version 3.3 of this service, it is now possible to request multiple optimizations independent of each other and results are stored in a RedisDB for later requesting of the results. Both URL paths and request methods have been adjusted to follow REST more closely. Additionally, it is now possible to run the optimization with multiple flexibility providers (i.e. batteries). In this regard, not only community storage units but also the household storage units could be scheduled in an optimized manner to achieve the virtual islanding optimization objective. Furthermore, the optimizer is capable of delivering (or receiving) a certain amount of bulk energy in form of bulk export or import for a certain period of bulk window within the use case time windows of the German use case field trials.

2.3 Interfaces and Communication mechanisms

This subsection briefly presents the functionalities of the different platform reference components.

Databus

There were no relevant changes after the second release and for the final third release.

MQTT Bridge

In the third release of the DSOTP, the MQTT Bridge was adjusted to also enable publishing of results to external receivers (e. g. Azure IoT gateway). This allows the continuous streaming of processed PMU data including the data from the power calculation. This especially results in a smooth transition from legacy systems that directly receive data via MQTT from the IoT devices. To facilitate this process, the topic structure in the databus was adjusted in such a way that the input data from the Blockchain Access Layer (BAL) (see Figure 4) is mapped to "bal/raw". The pre-processed data is taken from that topic and published to "bal/processed" and then published to a topic specific to the legacy system.

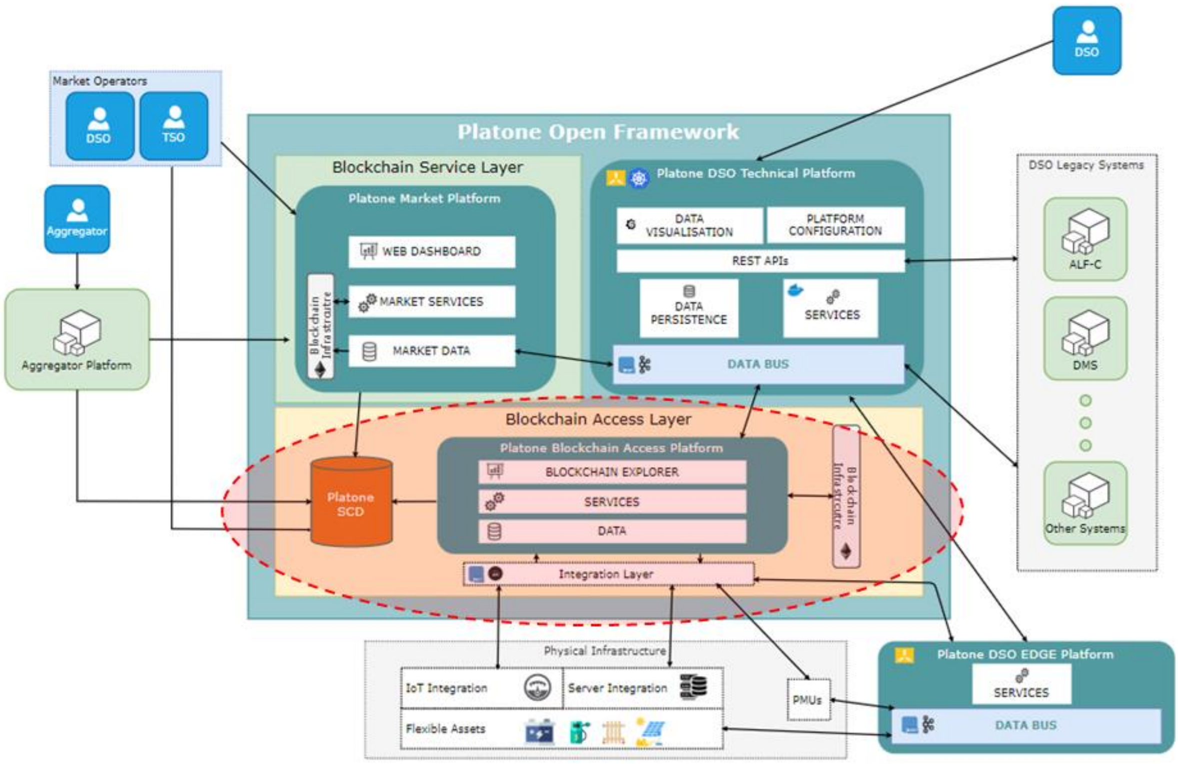

Figure 4 BAL in the Platone Open Framework

3 Languages, technologies, and external tools

The architecture of the DSO Technical Platform consists of different open-source tools and new components developed in the context of the different Platone demo use case requirements. The following table provides an overview of the core components of the platform and the integrated services including an insight into the used technologies, deployment methods and programming languages. The orange rows originate from the first and second release of the platform while the additional services and components of the third release are highlighted in green.

Table 1 Languages, technologies, and external tools

Layer/Component	Technologies/Framework	Deployment	Languages
Infrastructure	Kubernetes (K8s, K3s) Helm Docker	bare-metal	
Databus	RabbitMQ, Kafka Broker	Helm Chart, Operator	
Databus Bridge	Mosquitto	K8S Deployment	
Timeseries Database	InfluxDB, TimescaleDB	Helm Chart	
Database Adapter	Telegraf	Docker image Kubernetes deployment	Golang
Visualization Service	Grafana, React, React Flow	Helm Chart	JavaScript
Identity Management	Keycloak	Helm Chart	
Authentication Proxy	Nginx	K8S deployment	
BESS Optimizer Service	Microservice	K8S deployment	Python
PMU Power Calculation Service	Microservice	K8S deployment	Golang
State Estimation Service	Microservice	K8S deployment	Matlab, Octave

4 Packaging and Deployment

The third iteration of the DSO Technical Platform follows the same packaging and deployment provided in the second release.

Thus, for the experienced users that already know how to use Kubernetes and have access to a full-fledged Kubernetes cluster (either on premise, hybrid, or on public cloud infrastructure), the DSO Technical Platform can be deployed in that environment.

In other cases, with users that need further instructions for development setups or for edge cloud deployments, a set of instructions was provided in Deliverable D2.6 [2] for setting up a minimal Kubernetes cluster based on k3s [12], a lightweight Kubernetes distribution.

All mentioned configuration files as well as the entire documentation are available at [13].

5 Conclusion

The developments done on the third release of the Platone DSO Technical Platform exploited the micro-service oriented architecture enabling a smooth evolution towards the final implementation of flexibility control and grid observability tools needed on the demo sites.

As done for the previous versions, the services were defined and developed collaborating with the respective work package teams while the integration into the DSOTP and the additional components were conducted by WP2 partners, exclusively by RWTH, to ensure a reliable and secure operation of the services in the field trial environments of the demos.

The evolution of the services, Databus, and MQTT Bridge confirm the flexible approach adopted in the Platone DSO Technical Platform assuring stability, extensibility and re-usability of the solution possibly also for future projects.

In the next step and towards the remaining months of the project period, the state-estimation tool will be dockerized and deployed for the Greek demo while the corresponding dashboard will be implemented and deployed, too.

6 List of Tables

7 List of Figures

8 List of References

[1] European Commission, “2050 long-term strategy”, [Online]. Available: https://ec.europa.eu/clima/policies/strategies/2050_en

[2] Platone “D2.6 Platone DSO Technical Platform (v1)” [Online].

Available: https://www.platone-h2020.eu/data/deliverables/864300_M18_D2.6.pdf

[3] Platone “D2.7 Platone DSO Technical Platform (v2)” [Online].

Available: https://www.platone-h2020.eu/data/deliverables/864300_M38_D2.7.pdf

[4] Platone “D2.1 Platone Platform requirements and reference architecture (v1)” [Online]. Available: https://www.platone-h2020.eu/data/deliverables/864300_M12_D2.1.pdf

[5] Platone “D2.2 Platone Platform requirements and reference architecture (v2)” [Online]. Available: https://www.platone-h2020.eu/data/deliverables/864300_M30_D2.2.pdf

[6] “SOGNO energy: Service Oriented Grid for the Network of the Future” [Online]. Available: https://www.sogno-energy.eu/

[7] “Kubernetes open source” [Online]. Available: https://kubernetes.io/

[8] “Docker: Empowering App Development for Developers” [Online]. Available: https://www.docker.com/

[9] “NetworkX Python package”. Available: https://networkx.org

[10] “React-Flow”. Available: https://reactflow.dev/

[11] “Telegraf/basic-stats plugin”. Available: https://github.com/influxdata/telegraf/tree/master/plugins/aggregators/basicstats

[12] “K3s” [Online]. Available: https://k3s.io/

[13] “GitLab of the RWTH Aachen University” [Online]. Available: https://git.rwth-aachen.de/acs/public/deliverables/platone

9 List of Abbreviations

Abbreviation	Term
API	Application Programming Interface
BAL	Blockchain Access Layer
BESS	Battery Energy Storage System
DSO	Distribution System Operator
DSOTP	DSO Technical Platform
MQTT	Message Queuing Telemetry Transport
PMU	Phasor Measurement Unit
REST	REpresentational State Transfer
TSO	Transmission System Operator
UI	User Interface