Ref. Ares(2019)4140686 - 30/06/2019

Listed requirements for open APIs
Version 1.0

Deliverable D8.8

19/06/2019

This project has received funding
from the European Union's Horizon 2020
research and innovation programme under
grant agreement n°731289

ID & Title :	Listed requirements for open APIs		
Version :	V1.0	**Number of pages :**	27

Short Description
Listed requirements for open Application Programming Interfaces (going beyond the protocols to deliver flexibility services at the upper layer of the InterFlex API).

Revision history

Version	Date	Modifications' nature	Author
V0.1	26/04/2019	Document initialized and structure created	Sebastian Schwarz Amir Ahmadifar Jonas Baude
V0.2	24/05/2019	First draft ready for internal review by WP8	Sebastian Schwarz Amir Ahmadifar
V0.3	27/05/2019	Final draft ready for internal review by the TC	Sebastian Schwarz Amir Ahmadifar Marco Cupelli Pauline Ahlgren Jörgen Rosvall Helen Carlström
V1.0	19/06/2019	Final document including the feedback from the TC	Marco Cupelli Amir Ahmadifar Sebastian Schwarz

Accessibility

☒Public	☐ Consortium + EC	☐ Restricted to a specific group + EC	☐ Confidential + EC

Owner/Main responsible

Name(s)	Function	Company	Visa
Marco Cupelli		RWTH	

Author(s)/contributor(s): company name(s)
Amir Ahmadifar, Jonas Baude, Marco Cupelli, Sebastian Schwarz: RWTH Aachen University Pauline Ahlgren, Helen Carlström, Sebastian Jansson, Jörgen Rosvall: E.ON Sverige Stanislav Hes: CEZ Distribuce

Reviewer(s): company name(s)

Company	
CEZ Distribuce, E.ON, RWTH	

Approver(s): company name(s)

Company	
Enedis	

Work Package ID	GWP8	**Task ID**	8.8

Disclaimer: This report reflects only the author's view and the Agency is not responsible for any use that may be made of the information it contains.

Executive summary

Major distribution system operators are working together with different market players and other stakeholders within the Horizon 2020–LCE-06-2016 project InterFlex to develop open Application Programming Interface (API) solutions for flexibility management purposes. The fundamental idea is to easily initiate the whole chain of flexibility activation from the customer premise level up to the transmission level. To this aim, standardized systems and interfaces have been developed, set up, and successfully tested within a cloud platform setup, referred to as the InterFlex flexibility cloud platform. This deliverable provides a brief overview about the solutions developed and places them in the context of existing open standards.

Based on the work done by the different demonstration sites, and in particular by work package 8, the present work primarily derives and defines the main challenges and requirements on an open API specification to be used to communicate with the InterFlex flexibility cloud platform from the upper layer. This enables the different users of flexibility to access the InterFlex flexibility cloud platform in a user-friendly and rather abstract but distinct way. The work includes a discussion of the identified challenges on open APIs, and on how distribution system operators could effectively integrate an open API implementation into their existing tools and frameworks while satisfying all the crucial functional and non-functional requirements the same time. Moreover, this work provides technical recommendations on the consideration and application of these requirements, which should serve as references for future InterFlex open API realisations.

The content of this deliverable is intended to be standalone. For further deeper information, we refer the interested reader to the preceding deliverables D3.1 [1], D3.3 [2], and D3.4 [3].

TABLE OF CONTENT

LIST OF FIGURES

LIST OF TABLES

LIST OF ACRONYMS

API	Application Programming Interface
BESS	Battery Energy Storage System
BMS	Battery Management System
BRP	Balance Responsible Party
DSO	Distribution System Operator
DSR	Demand Side Response
EMS	Energy Management System
ESB	Enterprise Service Bus
EU	European Union
GUI	Graphical User Interface
HTTP	Hypertext Transfer Protocol
ICT	Information and Communications Technology
ISO	International Organization for Standardization
LES	Local Energy System
MG	Microgrid
OASIS	Organization for the Advancement of Structured Information Standards
OData	Open Data Protocol
PV	Photovoltaics
RES	Renewable Energy Source
REST	Representational State Transfer
SCADA	Supervisory Control and Data Acquisition
SGAM	Smart Grid Architecture Model
TSO	Transmission System Operator
TRL	Technology Readiness Level
UML	Unified Modelling Language
WP	Work Package

1 INTRODUCTION

The European Union (EU) project interactions between automated energy systems and flexibilities brought by energy market players (InterFlex) is a response to the Horizon 2020 Call for proposals, LCE-02-2016 (“Demonstration of smart grid, storage and system integration technologies with increasing share of renewable: distribution system”).

This Call addresses the challenges of the distribution system operators in modernizing their systems and business models in order to be able to support the integration of distributed Renewable Energy Sources (RESs) into the energy mix. Within this context, the LCE-02-2016 Call promotes the development of technologies with a high TRL (Technology Readiness Level) into a higher one.

InterFlex explores pathways to adapt and modernize the electric distribution system in line with the objectives of the 2020 and 2030 climate-energy packages of the European Commission. Six demonstration projects are conducted in five EU member states (Czech Republic, France, Germany, the Netherlands and Sweden) in order to provide deep insights into the market and development potential of the orientations that were given by the call for proposals, i.e., demand-response, smart grid, storage and energy system integration.

With Enedis as the global coordinator and CEZ Distribuce as the technical director, InterFlex relies on a set of innovative use cases. Six industrial-scale demonstration sites are being set up in the participating European countries. Figure 1-1 shows a map identifying the demo sites around Europe.

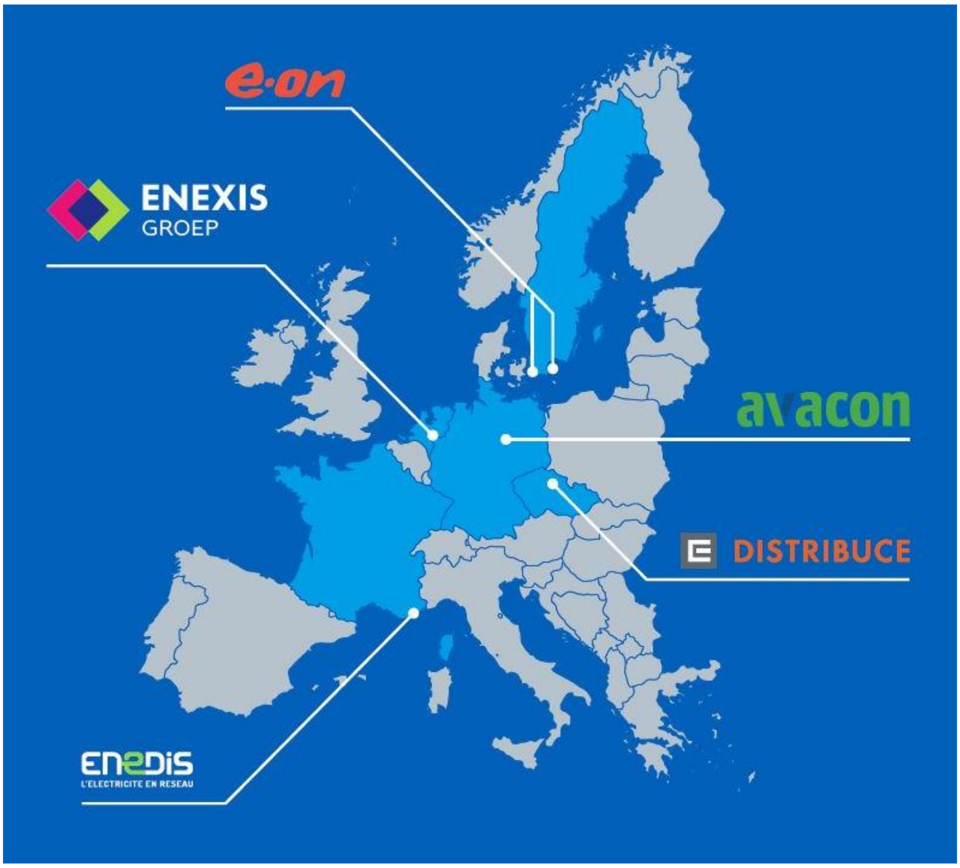

Figure 1-1 The map identifies the demo sites in the context of this project.

Through these demonstration showcases, the InterFlex project will assess how the integration of the new solutions can lead to a local energy optimization. Technically speaking, the success of these demonstrations requires that some of the new solutions, which are today at TRLs 5-7, are further developed reaching TRLs 7-9 to be deployed in real-life conditions.

1.1 Scope

The scope of this report is to list both the functional and non-functional requirements on an open Application Programming Interface (API) specification for flexibility control that addresses the specific needs and characteristics of the upper layer part of the InterFlex API as already defined in earlier work in InterFlex deliverable D3.4 [3]. The open API specification forms the communication interface to the so-called InterFlex API, which is a core component of the project's proposed flexibility cloud platform. The InterFlex API aims at abstracting characteristic properties of operational flexibility services in order to provide a unified interface to different stakeholders such as Distribution System Operators (DSOs), Transmission System Operators (TSOs) and/or Balance Responsible Parties (BRPs).

1.2 Objectives

The main objective of this work is to formally derive and define the requirements on an open API specification to be used to communicate with the InterFlex flexibility cloud platform. In other words, the open API specification is designed to provide access for the upper layer stakeholders such as DSOs, TSOs, BRPs or related market players to the InterFlex flexibility cloud platform in a user-friendly and rather abstract but distinct way. Crucial aspects such as Information and Communications Technology (ICT) security risks, the orchestration of functionality and data, or data privacy limitations are considered and briefly discussed in the following chapters. Based on that, we provide technical recommendations, which should serve as references for future InterFlex open API realisations.

1.3 Motivation

The enabling of flexibilities in distribution networks requires additional ICT systems and components. It is worthwhile to use standards and norms in order to ensure the highest possible level of interoperability of the various core components in the smart grid infrastructure and thus to increase the integration capability. The activation of flexibilities via the so-called upper bound, as presented in [1, 2], will provide numerous new (market) functionalities, services and applications, which are provided and used by (new) stakeholders [1, 2]. It is noteworthy to refer the reader to InterFlex deliverable D3.1 [1] for additional information about the clustering of the ICT architecture on the communication layer of Smart Grid Architecture Model (SGAM) diagrams and the respective decomposition on lower and upper bounds in the InterFlex project.

Figure 1-2 provides an overview of the already deployed InterFlex API and the overall flexibility cloud platform that primarily aims at interconnecting flexibility aggregators and DSOs. The InterFlex API is the key component of the flexibility platform that provides the interface between the cloud platform, its internal services and various stakeholders. In particular, the latter can be added to the platform in order to provide or enhance basic services. We distinguish between internal services such as data logging and user authentication that are deeply integrated into the platform and external services that can be used to integrate third-party data or services such as weather forecast or demand forecast. On these grounds, the flexibility platform allows for interconnecting DSOs to multiple aggregators that can control a variety of (heterogeneous) flexibilities. The flexibility representation is based on the characterization of operational flexibility as defined by the BRIDGE initiative [4]. It should be noted that the InterFlex API aims at abstracting the characteristic properties of the underlying flexibilities based on [4] and offering a unified, cross-carrier flexibility activation interface.

As depicted in Figure 1-2, DSOs – as the main users of flexibility - will use the upper layer of the InterFlex API, which is referred to as the *open API* in this report, to connect to the flexibility cloud platform. However, it is important to stress that not only DSOs may connect to the flexibility cloud platform but also other upper layer stakeholders such as the TSO or BRP might be the users of this open API, compare [5] and the InterFlex Work Package (WP) 3 flexibility request sequence diagram in Figure 1-3. Contrariwise, flexibility aggregators – as the providers of flexibility - connect to the lower layer of the InterFlex API, referred to as the *interoperable API*, which is specified in the form of Unified Modelling Language (UML) diagrams in deliverable D3.4 [3] in more detail. In general, both APIs work closely together in order to connect the stakeholders at the upper layer with the stakeholders accessing the platform from the lower layer. Nevertheless and to distinguish between both APIs in this report, we refer to two slightly different API definitions as follows:

- **Interoperable API:** "An Interoperable API is a specification for machine-to-machine interfaces that specifies the functionality, behaviour, and underlying data types but it does not specify the underlying (device and vendor-dependent) implementation".

- **Open API:** "An open API is a specification for machine-to-machine interfaces that describes, produces, consumes, and visualizes web services. The interface should provide open access, open data and open standards, but this does not mean that the IT system is open to everyone".

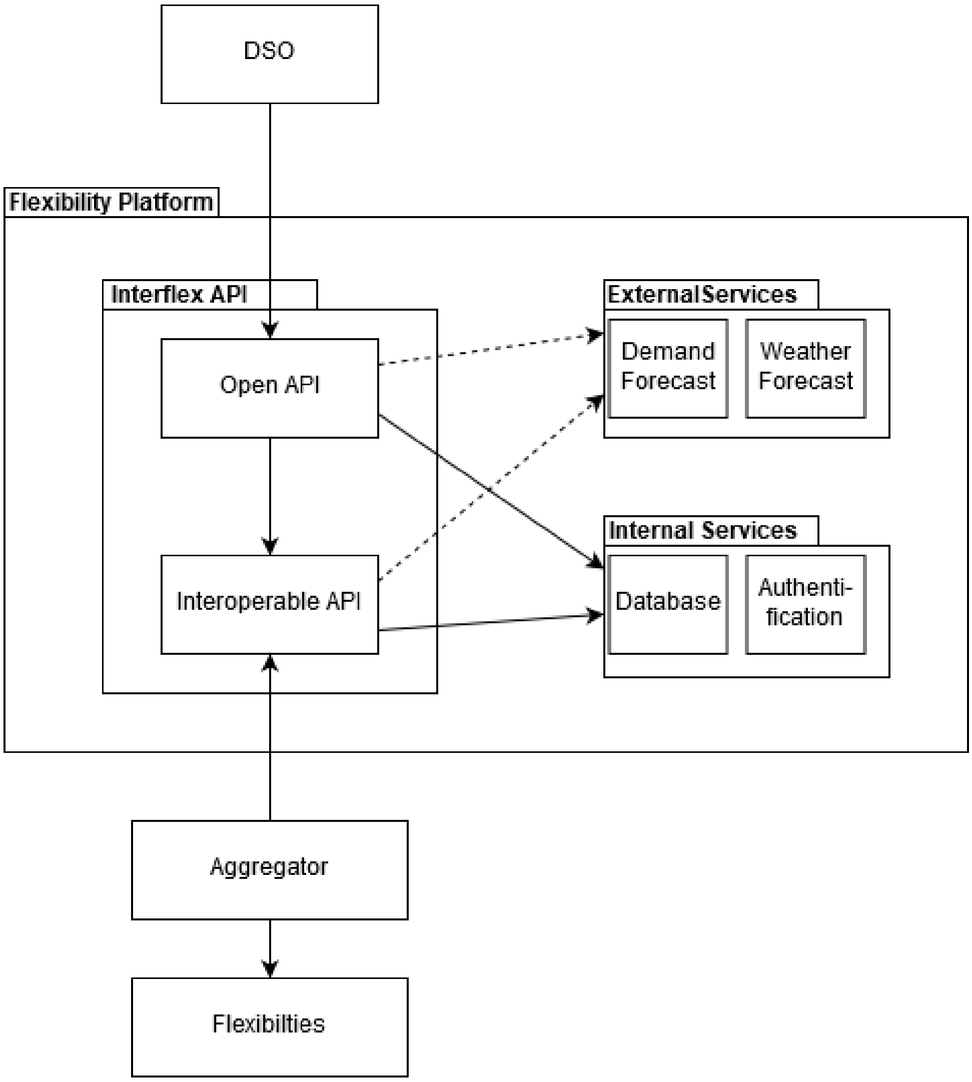

Figure 1-2 The InterFlex flexibility platform.

In this context, the provided definition for an open API grounds on the OpenAPI Specification, originally known as the Swagger Specification, which is a collaborative project of the community-driven OpenAPI Initiative within the Linux Foundation® as follows:
"The OpenAPI Specification (OAS) defines a standard, programming language-agnostic interface description for Representational State Transfer *(REST) APIs, which allows both humans and computers to discover and understand the capabilities of a service without requiring access to source code, additional documentation, or inspection of network traffic. When properly defined via OpenAPI, a consumer can understand and interact with the*

remote service with a minimal amount of implementation logic. Similar to what interface descriptions have done for lower-level programming, the OpenAPI Specification removes guesswork in calling a service”. [6]

According to this directive, the focus of the present work is on an open API specification adaptation for the InterFlex flexibility cloud platform, i.e., on the upper layer connection of the InterFlex API. We discuss the open API challenges on flexibility management purposes and analyse how DSOs could effectively integrate an open API implementation into their existing tools and frameworks while satisfying all the crucial functional and non-functional requirements on such an ICT interface at the same time.

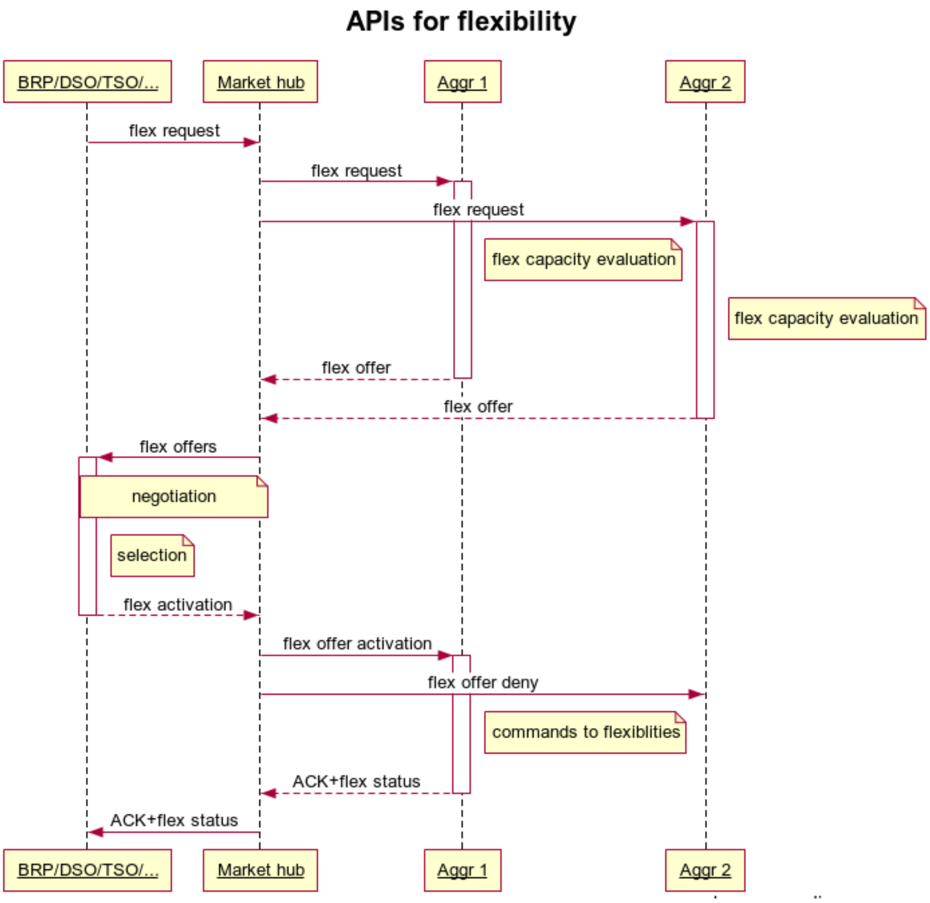

Figure 1-3 The InterFlex WP3 flexibility request sequence diagram.

1.4 Deliverable Organization

The rest of this document consists of three chapters. The following chapter provides a brief summary on two well-known open standards for open APIs and introduces state-of-the-art open API solutions, which are currently used in different InterFlex demonstration sites. In this context, we focus on the work done for the Swedish demonstrator by WP8. Based on the current state-of-the-art open API solutions in InterFlex, chapter 3 then identifies and lists the challenges and requirements on open API solutions that need to be taken into consideration in future open API realisations. Finally, chapter 4 concludes and summarizes this document and provides technical recommendations for the usage of an open API within upcoming revisions of the InterFlex API.

2 REFERENCE STANDARDS AND IMPLEMENTATIONS FOR OPEN APIs

This chapter presents and discusses the API implementations that have been developed during the InterFlex project and that are currently used by the different InterFlex demonstrators to connect DSOs to a flexibility cloud platform. Moreover, it should provide the reader of this document an overview on existing solutions for open API standards and on how they can be effectively used for flexibility management purposes.

2.1 Open Standards

Standards for ICT systems and components can be seen as agreements on how to do things in a well-defined way [7]. They specify a set of precise rules that serve as guidelines for both the developers and users of a software product or a software architecture in order to foster diverse software functional and non-functional requirements. Among others and according to ISO/IEC 25010 [8], these requirements should comprise the following criteria:

- Functional correctness and stability
- Performance efficiency
- Usability
- Accessibility
- Compatibility, interoperability, and portability
- Reliability
- Availability
- Maturity
- Security
- Maintainability

There are different classifications for such ICT standards. One can distinguish between the *personal* respectively *corporate* standards, where a group of individuals or a consortium of companies defines, documents, and sometimes patents its own standard for a certain solution, and the so-called *open* standards. As denoted by its name, an open standard is usually open in the terms of the free accessibility, transparency and reusability of its contents. It usually goes through an iterative process of design by a committee or non-profit association, where experts gradually make suggestions for technical improvement. These advantages usually make an open standard the preferred choice on the usage within new software products to be developed. [7]

One well-known open standard for an open API that takes advantage of the Representational State Transfer (REST) software architectural style for distributed hypermedia systems[1] [9], is the Open Data Protocol (OData) standard, which is introduced below and is also used

[1]REST is a widely used programming paradigm that defines a set of constraints for web services, i.e., machine-to-machine communication procedures. The main goal of REST is to provide interoperability between ICT systems on the Internet using the Hypertext Transfer Protocol (HTTP). REST Web services – also referred to as RESTful web services - allow ICT systems to access and manipulate textual representations of web resources (i.e., any kind of data) by using stateless operations. [9-11]

within InterFlex WP8. Furthermore, the concept of the FIWARE open source platform is described below, as it might be used in future releases of the InterFlex API, too.

2.1.1 OData

The OData application-level protocol is an ISO/IEC 20802 [12] approved standard that specifies a set of guidelines for producing and consuming RESTful APIs. Originally, OData was a Microsoft corporate standard, but it was submitted to the Organization for the Advancement of Structured Information Standards (OASIS) in 2012 and eventually became an open standard in 2014. OData defines a data model for REST-based resources to be accessed by heterogeneous web clients using HTTP requests. Its primary goal is to simplify the development of RESTful web services by predefining different REST implementation details such as request and response headers, status codes, HTTP methods, URL conventions, media types, payload formats, query options, and others. By providing a uniform way to describe both the data model and the data itself, OData is intended to improve the interoperability between systems as well as the overall software life cycle. For this purpose, the OData protocol provides the following features: [7, 12, 13]

- Metadata include a machine-readable description of the underlying data model
- Data may consist of sets of data entities and the relationships between them, but without assuming a relational data model
- Immediate filtering and transformations of data by queries
- Simple editing (creating, updating and deleting) of data
- Custom logics and semantics

2.1.2 FIWARE

FIWARE is an open source initiative defining open standards for context data management, where the term *context* refers to a variety of information from different domains, sources and devices such as from sensor networks or user interactions through mobile apps. The idea behind FIWARE is to efficiently capture, analyse, and further process these information through the FIWARE NGSI [14] open API standard. This means that the FIWARE API enables software developers to produce, gather, publish and consume context information at a large scale. The key element of FIWARE is the Core Context Broker, called *Orion*, which handles the different context information by implementing standard REST APIs. The strength of FIWARE is to model and gain access to context information in a way that is fully independent from the source of that information. This includes the usage of a sophisticated data model for storing context information, a context data interface for exchanging information by means of query, subscription, and update as well as a context availability interface for exchanging information on how to obtain the information. [15, 16]

Figure 2-1 provides a detailed overview on the FIWARE platform summarizing its software components and their functions.

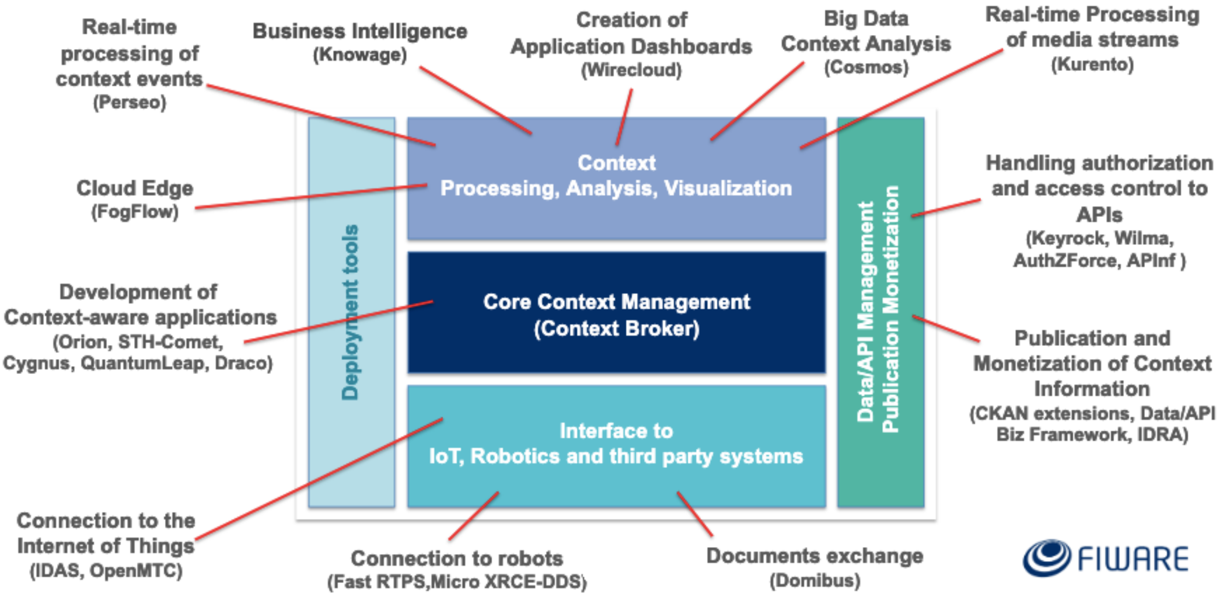

Figure 2-1 The FIWARE platform's components and functions. Source: [17]

2.2 InterFlex State-of-the-art Implementations

In this section, we present existing open API realisations that are currently used in the InterFlex project. However, as the focus of this report is on the work done in WP8, we primarily refer to the ICT systems and components used for the Swedish Local Energy System (LES) demonstrator in the village of Simris.

2.2.1 Swedish Demonstration Site

2.2.1.1 The Simris LES – Overview

The idea behind the Simris LES is to demonstrate that an electrical system can host a penetration of up to 100% of power obtained from local RESs by using field-proven and market available innovative technologies. For this purpose, the village of Simris in Southern Sweden can be connected and disconnected from the main electricity grid in a seamless way while being sourced by times solely by renewably energy coming from a local wind turbine of 500 kW, a PV farm of 442 kW, and residential rooftop PV installations of the local households. Thus, the Simris LES can be seen as a grid-connected Microgrid (MG) with the capability to operate islanded. In addition to the existing grid and RESs, the main assets and components forming the Simris LES, as visualised in Figure 2-2, are as follows:

- a main Battery Energy Storage System (BESS) of 333 kWh / 800 kW, which operates as the grid forming unit and which is in charge of the instantaneous balancing of the LES
- a secondary redox-flow BESS of 1.6 MWh / 200 kW
- a bio-diesel backup generator of 480 kW rated power

- a Demand Side Response (DSR) platform, which steers smart multi-carrier energy technologies (e.g., heat pumps, hot tap water boilers, and electric vehicle charging stations)
- an intelligent Energy Management System (EMS) communicating with all power production units so that the LES always delivers electricity within the conventional power quality limits

For further reading on the Simris LES, the reader is directly referred to earlier work in [18].

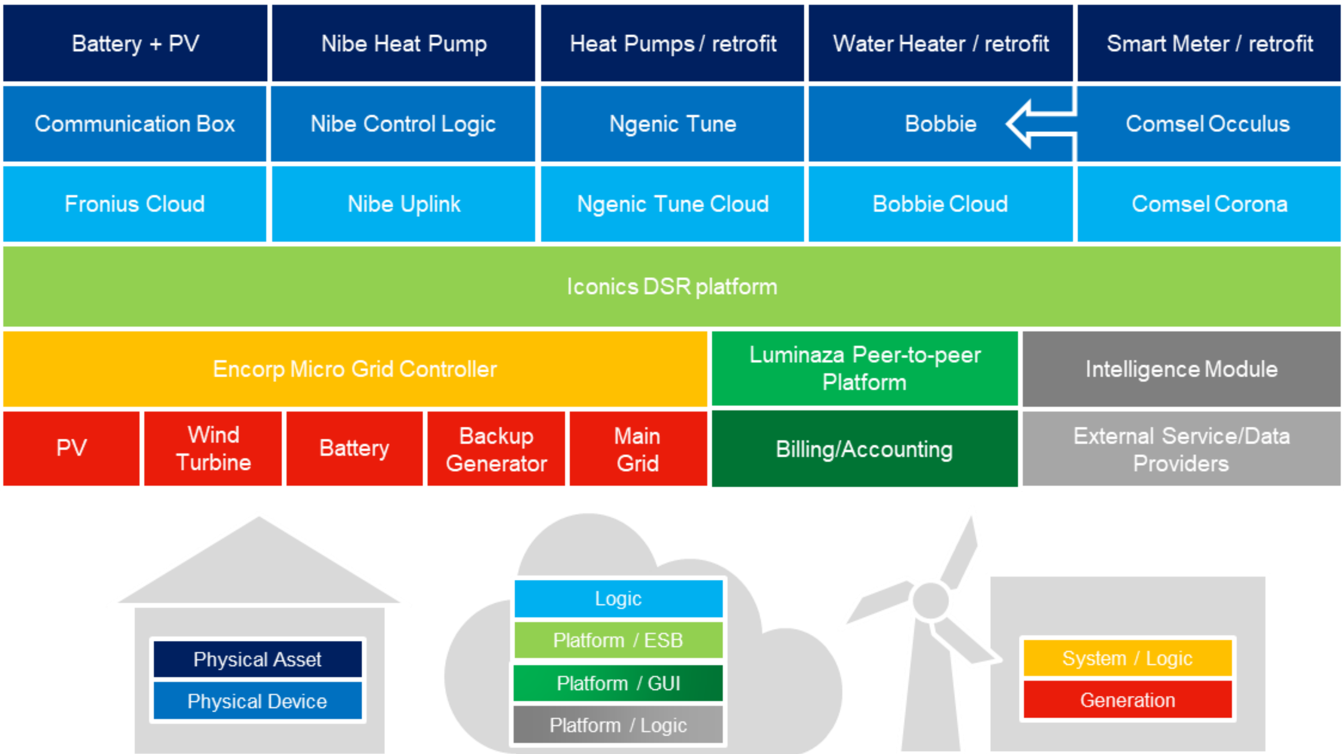

Figure 2-2 The assets and components of the Simris LES.

2.2.1.2 The Simris LES – State-of-the-art Open API Implementation

With reference to Figure 1-2, a flexibility cloud platform for the Simris LES was developed, set up, and successfully tested in InterFlex WP8. The current design of this flexibility cloud platform is shown in Figure 2-3. As one can see from this figure, the flexibility cloud platform contains an open API implementation, which allows E.ON – the responsible DSO for the Simris LES – to apply basic flexibility control strategies to the Simris LES. The open API rests on the OData standard enabling an upper layer stakeholder to communicate with the Simris DSR platform. The Simris DSR platform is an intermediate Enterprise Service Bus (ESB) device inside the flexibility platform separating upper layer and lower layer flexibility services. In other words, the DSR platform acts as a data carrier and is the orchestrating system between the different open API and interoperable API functionalities. It is important to mention that the Simris flexibility cloud platform is currently extended by the integration of the FIWARE standard in order to provide the user a sophisticated data model inside the flexibility cloud platform, compare InterFlex deliverable D3.5 [19].

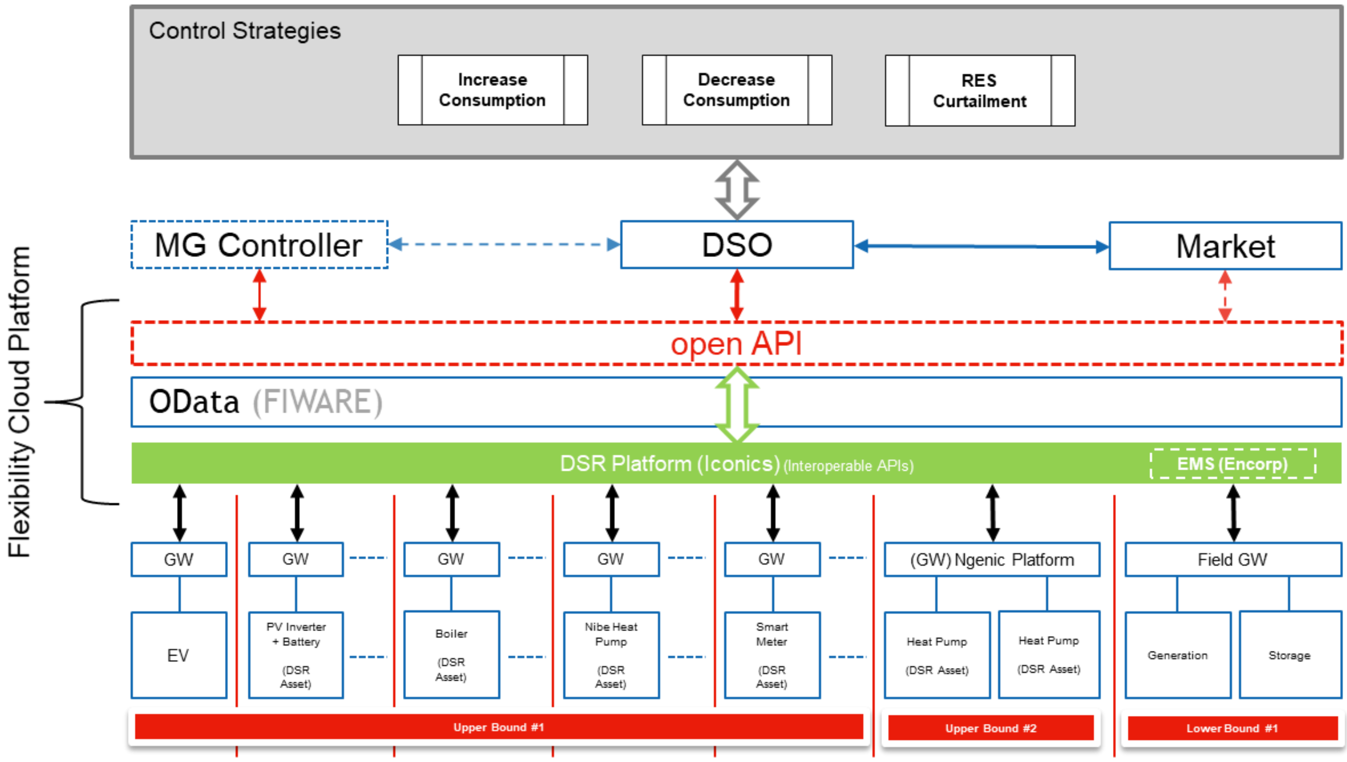

Figure 2-3 The Simris LES flexibility cloud platform.

In the platform's latest software version, the DSO can request (either manually or automatically through the MG controller) the Simris LES via the open API to apply the following flexibility control strategies:

- Increase Consumption
- Decrease Consumption
- RES Curtailment

The different requests are received and processed by the Simris DSR platform using internal logic for the LES asset control. For example, the control strategy may influence the residential heating units' or residential battery systems' electrical power set-points depending on the specific request. To this aim, the Simris DSR platform sends out the corresponding steering signals to the different LES assets using the interoperable APIs lower level connections as defined by the InterFlex API.

2.2.2 Other Demonstration Sites

Besides the Swedish demonstration site, open API implementations are also used at other pilot sites in the scope of InterFlex. This is due to the fact that open API solutions usually become interesting when third-party stakeholders should gain access to the different flexibility measures and services maintained by a DSO.

On these grounds, ENEDIS (WP9) is currently experimenting with an open API for the sharing of local load and generation data on the municipality level. The idea is to help local authorities to get a better understanding of local energy challenges and its interrelations by providing the authorities a simplified overview on local energy needs and properties through simplified open API functions.

Furthermore, ENEXIS (WP7) investigates in the applicability of an open API solution in connection with the USEF+ open framework standard [20] in order to exploit locally installed electric vehicle supply equipment for congestion management and voltage control. Thus, an open API can become helpful for interactions between the DSO, customers, commercial aggregators and/or technical aggregators.

3 CHALLENGES AND REQUIREMENTS ON OPEN APIs – FEEDBACK FROM DEMONSTRATORS

In this chapter, we list and define the key requirements on an open API specification to be used to communicate with a flexibility cloud platform. To this aim, feedbacks from the InterFlex demonstration sites have been collected during several workshops as well as through a questionnaire sent to each demonstrator leader throughout March 2019. Based on the workshop discussions and the answers to the questionnaire from the different demonstrators, subsequently we first identify the main challenges that are linked to the usage of an open API solution. In a second step and based on the identified challenges, we then derive the requirements on such open API realisations.

3.1 Challenges

The different demonstrators identified the following challenges on the usage of an open API solution for flexibility management purposes:

Challenge 1 – Security risks:
ICT systems always carry a certain (cyber-)security risk. Since the power grid is considered as a critical infrastructure, an open API solution must be resistant to cyberattacks.

Challenge 2 - Data privacy:
An open API solution must ensure the data privacy safeguard of all stakeholders involved. This becomes even more critical if communication happens in the customers' sphere of influence.

Challenge 3 – System complexity:
Systems that take advantage of open API use cases typically possess a high level of complexity that needs to be assessed and addressed. The reason for this usually grounds on both a huge number of ICT components installed and a huge number of different stakeholders present in the system.

Challenge 4 – Reliability and availability:
ICT systems and components are always subject to reliability and availability issues. In case of failure of an open API (sub-)system, the reliable operation of the power grid and other grid-related components must still be ensured.

Challenge 5 – Emergency cases:
In case of emergency, e.g., in case the reliable operation of the power grid is jeopardized, a DSO always needs the full opportunity to manually keep control over the entire system. Thus, an open API solution must consider and effectively handle such interventions.

Challenge 6 – System efficiency:
An open API solution should increase the overall efficiency of the flexibility management system up to a certain extent. For this purpose, it must bring a certain value to the DSO and/or other stakeholders.

Challenge 7 – Poor data quality:
One must assume different temporal resolutions, inconsistencies or erroneous entries for an open API's underlying working data set, since the data may stem from different sources and services. This means that an open API solution must be capable of dealing with (piecewise) poor data quality in an adequate way.

Challenge 8 – Easy integration:
An open API solution should always support an easy integration into/interaction with existing tools. This promotes the effectiveness of the solution while reducing the development time and the financial costs.

3.2 Requirements

We derive the key requirements on an open API solution for flexibility management purposes from the challenges presented in the previous section as follows:

Requirement 1 – Security risks:
Resistance towards cyberattacks is never completely avoidable. However, an open API implementation must at least be fully compliant with the national regulation and the requirements on (cyber-)security for critical infrastructures. Additionally, separate or parallel topologies isolated from the overarching Supervisory Control and Data Acquisition (SCADA) system functionality are considered indispensable. This delimitation should also apply for the EMS control in LESs. Furthermore, it is important to stress that one should always implement ICT security mechanisms such as data encryption or multi-step authentication procedures if available.

Requirement 2 - Data privacy:
In order to ensure the data privacy of the different stakeholders and customers involved, an open API solution must always satisfy the legal framework on the data privacy safeguard. Moreover, advanced measures such as the anonymization of data, information hiding and related techniques are highly appreciated.

Requirement 3 – System complexity:
In order to properly deal with a high system complexity, an open API solution must be subject to the following features:

- maturity
- scalability
- adaptability
- compatibility
- modularity

Since open standards target these features per se, compare section 2.1, the usage of an open API would already be a good basis to fulfil this requirement. Nevertheless, a comprehensive documentation of all ICT components and the overall software architecture as well as a methodical separation respectively clustering of subsystems is as important, too.

Requirement 4 – Reliability and availability:
The fully reliable and continuous operation of the power grid and its grid-related components can only be achieved by setting up redundant systems. Therefore, this requirement also

holds true for an open API solution and its connected ICT components. Further, the usage of field-proven devices is considered crucial.

Requirement 5 – Emergency cases:
During emergency cases, the DSO still needs to keep full control over the entire system. For example, an open API solution could hence incorporate a prioritization scheme, in which all DSO emergency requests are always prioritized compared to the ones of other stakeholders. Under normal operation conditions, however, the requests of all stakeholders should be treated equally.

Requirement 6 – System efficiency:
In order to increase the overall efficiency of the flexibility management system, an open API solution must provide a significant improvement to the overall system operation. For example, one can achieve this improvement through the orchestration of functionality and data, which would allow the system to apply, e.g., a data-driven instead of a rule-based control strategy approach. Furthermore, the open API solution should support a broad variety of different flexibility services in order to satisfy the needs of the heterogeneous stakeholders involved. An overview on such desirable flexibility services as specified by abstract but simple open API function calls can be found in appendix A1.

Requirement 7 – Bad quality of data:
The avoidance of bad data quality is a prerequisite on this requirement. Thus, to improve the data availability and data quality, the usage of high-performance system architectures and high frequent data streams matching the available bandwidth of the ICT connections is essential. Nevertheless, an open API solution must be capable to also deal with a bad quality of data and in fact requires routines to identify and probably recover from such bad data inputs.

Requirement 8 – Easy integration:
The integration of a new open API solution usually requires interoperability with other APIs and/or ICT systems. Thus, interoperability is a key requirement for open API realisations but is often already taken into consideration by the underlying open standard specification. Moreover, a new open API solution should be able to adapt to multiple standardized APIs, but should also have the ability to manage customized APIs as they are frequently used by DSOs today. Furthermore, it is desirable that a new open API solution comes along with the ability to seamlessly replace legacy technologies.

4 CONCLUSIONS AND INTERFLEX RECOMMENDATIONS

The present document provides a brief overview on state-of-the-art open API solutions for flexibility management purposes and their usage throughout the different InterFlex demonstration sites. With reference to the solutions developed by the Swedish demonstrator in WP8 and the specific needs and properties of the other InterFlex demonstration sites, this deliverable lists and defines the challenges and requirements on future open API realisations. This includes a brief discussion of the identified challenges on open APIs, and on how DSOs could effectively integrate an open API implementation into their existing tools and frameworks while satisfying all the crucial functional and non-functional requirements on such ICT interfaces at the same time.

Based on the analyses conducted, it turned out that the usage of an open API solution for flexibility management purposes brings several advantages. First, the usage of an open API solution guarantees free accessibility, transparency, and reusability of contents. Second, an open API solution usually fosters important software features such as scalability, adaptability, compatibility, modularity, and maintainability. Third, through the usage of an open API , upper layer stakeholders such as the DSO, TSO or BRP can easily initiate the whole chain of flexibility activation (i.e., from the customer premise level up to the transmission level) without requiring a detailed knowledge about the underlying ICT systems and components installed. Fourth, an open API realisation increases the overall system efficiency of the flexibility management system by significantly improving the core functionalities of the system, for instance, by enabling the system to apply advanced data-driven control strategies. Fifth, the abovementioned benefits additionally tend to decrease the overall system and operation costs, which makes the usage of an open API also viable from a financial perspective.

Nevertheless, there are also critical requirements, which might hinder the usage of an open API solution for flexibility management purposes in practice. As the major bottleneck, the requirement on ICT security was identified, which represents a legal obstacle on the usage of open APIs by DSOs. In particular, with regard to the bidirectional flow of information and the presence of critical grid infrastructure, ICT security requirements become difficult to meet. Moreover, the compliance with data privacy requirements can be a big challenge for DSOs. For this reason, we would like to emphasize that further investigations are necessitated in this area.

However, an answer towards the fulfilment of the ICT security and data privacy requirements could be in the usage of well-established open standards for energy management systems such as FIWARE. Since its design is intended to meet these aspects inherently, FIWARE is seen as a good starting point to remove the existing barriers. This is why InterFlex WP3 currently integrates the FIWARE open standard into the InterFlex API and hence highly recommends its application in the near future. We further suggest that upcoming H2020 projects should adapt from the InterFlex API implementations in order to make use of flexibility services in an open and interoperable way while satisfying the identified requirements in the best possible way.

5 BIBLIOGRAPHY

[1] InterFlex, *Deliverable 3.1: Demo and use case view on required interfaces/functionalities*. December 2018.

[2] InterFlex, *Deliverable 3.3: Feedback on demonstrations and use case interoperability*. December 2018.

[3] InterFlex, *Deliverable 3.4: Interoperable APIs Specification*. April 2019.

[4] Bridge Horizon 2020, *Characterization of flexibility services V1.1: Data Management WG*. March 2018.

[5] Smart Grids Task Force, Expert Group 3, *Regulatory Recommendations for the Deployment of Flexibility: SGTF-EG3 Report*. January 2015.

[6] The Linux Foundation®, *OpenAPI Initiative*. [Online] Available: https://www.openapis.org/. Accessed on: May 10 2019.

[7] L. Richardson and M. Amundsen, *RESTful Web APIs*. Beijing: O'Reilly, 2013.

[8] International Organization for Standardization (ISO), *ISO/IEC 25010:2011(E): Systems and software engineering - Systems and software Quality Requirements and Evaluation (SQuaRE) - System and software quality models*. Vernier, Switzerland, 2011.

[9] R. T. Fielding, *Architectural styles and the design of network-based software architectures*. Dissertation. University of California, Irvine, 2000.

[10] D. Murrmann, *REST: Grundlagen und Gestaltung von REST-Diensten*. Frankfurt am Main: entwickler.press, 2016.

[11] W3C Web Services Architecture Working Group, *Web Services Architecture: W3C Working Group Note 11 February 2004*. [Online] Available: https://www.w3.org/TR/2004/NOTE-ws-arch-20040211/wsa.pdf. Accessed on: May 10 2019.

[12] International Organization for Standardization (ISO), *ISO/IEC 20802-1:2016(E): Information technology - Open data protocol (OData) v4.0*. Part 1: Core. Vernier, Switzerland, 2011.

[13] OData, *OData - The Protocol for REST APIs*. [Online] Available: https://www.odata.org/. Accessed on: May 10 2019.

[14] FIWARE Foundation, *FIWARE-NGSI v2 Specification*. [Online] Available: http://fiware.github.io/specifications/ngsiv2/stable/. Accessed on: May 13 2019.

[15] FIWARE Foundation, *FIWARE: The Open Source Platform For Our Smart Digital Future*. [Online] Available: https://www.fiware.org/. Accessed on: May 13 2019.

[16] FIWARE Foundation, *FIWARE Webinars: FIWARE Overview*. [Online] Available: https://www.fiware.org/fiware-webinars/. Accessed on: May 13 2019.

[17] FIWARE Foundation, *The FIWARE Catalogue on GitHub*. [Online] Available: https://github.com/FIWARE/catalogue. Accessed on: May 13 2019.

[18] H. Wilms *et al.*, “Microgrid Field Trials in Sweden: Expanding the Electric Infrastructure in the Village of Simris,” *IEEE Electrific. Mag.*, vol. 6, no. 4, pp. 48–62, 2018.

[19] InterFlex, *Deliverable 3.5: Open Reference Implementation*. June 2019.

[20] Universal Smart Energy Framework (USEF) Foundation, *Universal Smart Energy Framework: A solid foundation for smart energy futures*. [Online] Available: https://www.usef.energy/. Accessed on: May 21 2019.

A1. APPENDIX – DESIRED OPEN API FUNCTIONS

The following table provides feedbacks from the different InterFlex demonstration sites on desired future open API functions for flexibility management purposes. We additionally evaluated the importance, the technical availability and the feasibility in practice for each of the open API functions.

Table 1 Overview on desired open API functions for flexibility management purposes.

Open API function	Brief function description	Importance of the function	(Technical) availability of the function in practice	Feasibility of the function in practice
get_weather_data	Get the current and/or future weather data (e.g. temperature, solar irradiation, wind speed, etc.) for a specified location	Medium	Already existing with minor limitations	Medium
get_consumption_data	Get the current and/or future load consumption data for a specified area (in kW)	Very important	State-of-the-art	Easy
get_generation_data	Get the current and/or future generation data for a specified area (in kW)	Very important	Already existing with minor limitations	Easy
get_flexibility	Get the current and future flexibility availability for a specified area taking into account all flexible assets present (in kWh)	Important	Already existing with major limitations	Very difficult
get_grid_value	Get certain physical values for a specified grid segment or component (e.g. voltage, current, active/reactive power, frequency, harmonics, etc.)	Very important	Already existing with major limitations	Medium
get_grid_status	Get the current status (e.g. breaker positions,	Very important	Already existing with major limitations	Medium

	congestions, maintenance mode, etc.) for a specified grid segment or component			
get_asset_status	Get the current status of a certain asset (e.g. a wind turbine) or a group of assets (e.g. all stationary battery units)	Very important	Already existing with minor limitations	Medium
get_operation_cost	Get the operation cost for operating the system under the current/future conditions	Important	To be developed	Difficult
set_consumption	Set the total load consumption for a specified grid area (in kW). The desired value must be physically realizable.	Nice to have	Already existing with minor limitations	Medium
set_generation	Set the total generation for a specified grid area (in kW). The desired value must be physically realizable.	Nice to have	Already existing with minor limitations	Medium
set_active_power	Set the current active power consumption or generation for a specified grid component (in kW). The desired value must be physically realizable.	Nice to have	Already existing with minor limitations	Difficult
set_reactive_power	Set the current reactive power consumption or generation for a specified grid component (in kVAr). The desired value must be physically realizable.	Nice to have	Already existing with minor limitations	Difficult

set_max_consumption	Set the maximum load consumption for a specified area (in kW). The desired value must be physically realizable.	Important	Already existing with major limitations	Difficult
set_max_generation	Set the maximum generation for a specified area (in kW). The desired value must be physically realizable.	Important	Already existing with major limitations	Difficult
set_max_active_power	Set the maximum active power consumption or generation for a specified grid component (in kW). The desired value must be physically realizable.	Nice to have	Already existing with minor limitations	Very difficult
set_max_reactive_power	Set the maximum reactive power consumption or generation for a specified grid component (in kVAr). The desired value must be physically realizable.	Nice to have	Already existing with minor limitations	Very difficult
set_flexibility	Set the future flexibility availability for a specified area (in kWh). The desired value must be physically realizable.	Nice to have	Already existing with major limitations	Difficult
set_operation_mode	Set the operation mode for a specified grid area or grid component. For instance, this might be the option to switch between the grid-connected and islanded mode for LESs.	Very important	Already existing with minor limitations	Medium

set_use_case	Apply a predefined use case/service to a specified grid area. Examples are peak shaving, cost optimization, increase of DER hosting capacity or other services.	Very important	Already existing with minor limitations	Medium