D6.1 Semantic Middleware Architecture Specification

Document Information

<table>
<thead>
<tr>
<th>Program</th>
<th>FP7-ICT-2013-11</th>
</tr>
</thead>
<tbody>
<tr>
<td>Project acronym</td>
<td>OS4ES</td>
</tr>
<tr>
<td>Grant agreement number</td>
<td>619302</td>
</tr>
<tr>
<td>Number of the Deliverable</td>
<td>D6.1</td>
</tr>
<tr>
<td>WP/Task related</td>
<td>WP6 / T6.1</td>
</tr>
<tr>
<td>Type (distribution level)</td>
<td>PU</td>
</tr>
<tr>
<td>Start date of project</td>
<td>01/07/2014</td>
</tr>
<tr>
<td>Date of delivery</td>
<td>31/07/2015</td>
</tr>
<tr>
<td>Status and Version</td>
<td>Final Version 1.0</td>
</tr>
<tr>
<td>Number of pages</td>
<td>52 pages</td>
</tr>
<tr>
<td>Document Responsible</td>
<td>TECNALIA, HYPERTECH</td>
</tr>
<tr>
<td>Author(s)</td>
<td>Antonis Papanikolaou, Christos Malavazos, Eleana Chatzoplaki, Stjepan Sucic, Mikel Fernandez, Maitena Ilardia</td>
</tr>
<tr>
<td>Reviewers</td>
<td>Christoph Brunner – IT4Power</td>
</tr>
</tbody>
</table>
# Revision History

<table>
<thead>
<tr>
<th>Version</th>
<th>Date</th>
<th>Author/Reviewer</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.01</td>
<td>24.02.2015</td>
<td>Tecnalia</td>
<td>ToC generation</td>
</tr>
<tr>
<td>1.0</td>
<td>16.06.2015</td>
<td>Hypertech</td>
<td>ToC finalization</td>
</tr>
<tr>
<td>2.0</td>
<td>21.07.2015</td>
<td>Hypertech</td>
<td>Chapter 4 updating</td>
</tr>
<tr>
<td>3.0</td>
<td>10.08.2015</td>
<td>Hypertech</td>
<td>Draft document generation</td>
</tr>
<tr>
<td>4.0</td>
<td>20.08.2015</td>
<td>IT4Power</td>
<td>Document review</td>
</tr>
<tr>
<td>Final</td>
<td>02.09.2015</td>
<td>Hypertech</td>
<td>Document revision</td>
</tr>
</tbody>
</table>
Executive Summary

The Semantic Middleware Specification describes the message transferring component of the OS4ES system that will be implemented and developed in the Work Package 6 of the OS4ES project. This deliverable includes only a textual description, while the other deliverables of WP6 will constitute the deliverables of the implementation phases of the developed software prototypes. The deliverable takes the D.1.2 “OS4ES system architecture, component requirements and communication infrastructure”, where the general system architecture is described and its various components requirements are analysed, as an input for the design of the Semantic Middleware. The Semantic Middleware is the central communication platform of the OS4ES system and handles the interaction between the different components. Possible components connected to the platform are the DER Systems, Registry and SmartGrid applications. This document describes how the communication between the components is performed and how it is secured. Finally, this deliverable provides an important input for the other work packages which are designing and implementing other components, since they will all interact with the Semantic Middleware.
List of Tables

Table 1 - Acronyms list ............................................................................................................ 10
Table 2 – MMS objects and services in use within this SCSM .................................................. 36
Table 3 – Required IEC 61850-7-2 client/server services ..................................................... 37
Table 4 – Mapping of ACSI Associate service to MMS ............................................................ 41
Table 5 – Mapping of ACSI Release service to MMS ............................................................... 42
Table 6 – Usage of XMPP stanzas ............................................................................................ 50
List of Figures

Figure 1 - General OS4ES Platform Architecture ........................................................................ 9
Figure 2 - Middleware General Conceptual Architecture .............................................................. 11
Figure 3 – IEC 61850 data class model .......................................................................................... 15
Figure 4 – IEC 61850 Client and Server normal operation ............................................................ 17
Figure 5 – IEC 61850 Client or Server Association Aborting ....................................................... 18
Figure 6 – Reference Middleware Conceptual Architecture .......................................................... 23
Figure 7 – Reference Implementation Component Diagram .......................................................... 24
Figure 8 – XMPP Stream Management Extension ....................................................................... 29
Figure 9 – UML Sequence Diagram of Message Routing between Components having an IEC 61850 Client or Server .......................................................................................... 33
Figure 10 – UML Sequence Diagram of Message Routing between a SmartGrid Application and a component with an IEC 61850 Server ............................................................................. 34
Figure 11 – Two OS4ES-XMPP Clients deployment methods ....................................................... 44
Figure 12 – XMPP Deployment within an XMPP Domain ............................................................... 45
Figure 13 – Concept of federation in XMPP ................................................................................ 46
Figure 14 – Two domain federation ............................................................................................. 46
Figure 15 – End-to-end security profiles ..................................................................................... 48
Figure 16 – Initial handshake for AL/A+/AE+ Profiles ................................................................ 49
Figure 17 – Request / Response messages for AL/A+ profiles ...................................................... 49
Figure 18 – Request / Response messages for AE+ profile ............................................................ 49
1 Introduction

1.1 Document scope and objectives

The purpose of this document is to describe the design and the specifications of the Middleware of the OS4ES platform. Moreover it details the specifications given in a D.1.2 “OS4ES system architecture, component requirements and communication infrastructure”, where the overall reference architecture was described.

This document also describes the internal design of the Semantic Middleware and its internal architecture making sure that it fits to the overall OS4ES project general architecture. Looking into the Semantic Middleware architecture further in detail, the different components are presented and described. At start, the General Architecture of the Middleware OS4ES component will be described, without taking into account the technologies & standards that will be used in its implementation phase. In later chapters, the technologies that are selected for the various components of the OS4ES platform are presented and collocated at the Semantic Middleware architecture. The final architecture with the technologies and standard integration is then analysed in more detail, as well as the available deployment methods and implementation paths.
1.2 OS4ES Overall Architecture and Middleware Area

The high level architecture of the OS4ES platform was defined in Deliverable 1.2 and is illustrated in Figure 1.

As shown in the figure, the architecture contains four basic blocks of capabilities:
1. Registry, represented by the leftmost red square
2. OS4ES functions, represented by the rightmost green rectangle
3. Conversion layer, represented by the outer dark purple concentric ring
4. Control layer, represented by the inner light purple concentric ring

These four modules/blocks comprise the scope of the OS4ES system. The Smart Grid Applications are assumed to host the business intelligence of the OS4ES user, hence will reside in his IT domain. DER systems do not belong to the scope of the OS4ES system, but will interface with it directly to facilitate the creation and provision of the envisioned services.
The role of the Semantic Middleware focuses on the Control and the Conversion layer of the whole OS4ES platform architecture. The middleware Software component will sit between the different peripheral components of the platform and provide messaging services, such as concurrency, transaction management, threading and messaging, in order to facilitate their communication processes (Control layer). Modern integration infrastructure as API management software will provide greater governance, risk management and accountability. Message conversion between different data models and communication protocols will provide interoperability between otherwise non-interoperable OS4ES users (Conversion Layer).

1.3 Notations, abbreviations and acronyms

<table>
<thead>
<tr>
<th>Term</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>HTTP</td>
<td>Hypertext Transfer Protocol is an application protocol for distributed, collaborative, hypermedia information systems</td>
</tr>
<tr>
<td>QoS</td>
<td>Quality of Service is the overall performance of a telephony or computer network, particularly the performance seen by the users of the network. To quantitatively measure quality of service, several related aspects of the network service are often considered, such as error rates, bandwidth, throughput, transmission delay, availability, jitter, etc.</td>
</tr>
<tr>
<td>XML</td>
<td>eXtensible Markup Language is a markup language that defines a set of rules for encoding documents in a format which is both human-readable and machine-readable.</td>
</tr>
<tr>
<td>XMPP</td>
<td>Extensible Messaging and Presence Protocol is a communications protocol for message-oriented middleware based on XML (Extensible Markup Language)</td>
</tr>
<tr>
<td>MMS</td>
<td>Manufacturing Message Specification is an international standard (ISO 9506) dealing with message system for transferring real-time process data and control information between network devices and computer applications</td>
</tr>
<tr>
<td>SCSM</td>
<td>Specific Communication Service Mapping is a part of the entire IEC 61850 communication framework</td>
</tr>
</tbody>
</table>
2 Middleware General Architecture

The Semantic Middleware general architecture is described in this chapter, without taking into account any technology and standard integration. The conceptual architecture design was driven by the objective to allow the adaptation to any communication protocol, middleware and the various OS4ES components data models and communication standards.

2.1 Middleware Conceptual Architecture Diagram

![Middleware General Conceptual Architecture Diagram]

Figure 2 - Middleware General Conceptual Architecture
In the above figure (Figure 2) the Semantic Middleware high level architecture is illustrated. The whole design was driven by the fact that the in contradiction to the heterogeneity of the various components and actors of the system, their interoperability and mediate interaction should be ensured. The whole architecture is server based; any communication between the clients is done by exchanging messages over a central server farm. More specifically, the Semantic Middleware can be divided into two significant parts: the Core and the Peripheral part.

### 2.2 Control Layer – Core Part

The Core part of the Semantic Middleware is represented in the architecture as the Middleware Server Farm and plays the role of the Control Layer of the high level architecture of the OS4ES platform. Physically, the Core part components can be located in one or more places, being though co-located with network switched or routers in order to be able to communicate.

Any OS4ES system component having network access to the Core part is able to exchange messages between other connected components through the Middleware Server Farm. An authentication process will check the ability of an external component to connect to the Middleware Server Farm through a Middleware Client and an authorization process will guarantee for the corresponding component rights to exchange messages with certain components. The message validation process will also ensure the communication compatibility between the senders and the receiver.

The essence of this layer though, is the message routing of any passing message. The safe, correct and precise message delivery is the most important functionality of this layer, supporting the whole OS4ES platform operation. Different interaction modes between the system components can be supported by the layer, like synchronous invocations, asynchronous message passing as well as shared coordination.

Traffic management and indirection mechanisms will be also supported by the Core part of the Semantic Middleware in order to obtain performance at acceptable levels.

Backup functionality on primary servers will also secure the Middleware Server Farm in cases of primary-server failure.

Server clustering where multiple servers support the same domain, so that any of the set of cluster servers handles the domain can be also supported by the Semantic Middleware. This has a number of advantages:

- In the event of server failure (short or long outage) there is another server in place to take over. Clients and servers connected to the failed server can simply reconnect to one of the other cluster nodes.
• It can help with network failure and network partition. This can be conscious partitioning (e.g., operating a pair of cluster nodes, with one internal and one in the DMZ with external access). By placing cluster nodes at separate network locations, it can also deal with temporary network outages, minimizing client/server failures by always connecting clients to local servers.

2.3 Conversion Layer – Peripheral Part

The Conversion Layer of the OS4ES architecture is located on every Client of the Semantic Middleware, which is why it is considered to be the peripheral part of the Middleware, and is the responsible layer for facilitating all the possible message conversions. Three types of Middleware Clients are considered in the general architecture of the Semantic Middleware: the registry, the DER and the Smart Grid application Middleware Client.

Any message conversion process may require two steps. The first step includes the sender’s (DER, SG Application or Registry) data model translation to the Middleware data model and will be realized locally at each of the Middleware Clients dedicated to the sender. The second step will again be executed locally on the Middleware Client of the message receiver and will concern the reverse procedure of transforming the Middleware data model to the corresponding receiver’s data model. The data model mapping process will be conducted by software components located in each Middleware Client, which will be specialized in the corresponding message transformation and de-transformation.
3 Technology & Standards selections

In this chapter, the final standard and technology choices will be described. More emphasis will be given to the main technologies/standards characteristics that affect the Middleware functionality.

3.1 IEC 61850 Standard

The DER System and Registry final protocol selection is the IEC 61850 Standard family as far as data modeling and communications are concerned. More specifically, all the DER system connected to the OS4ES platform are considered to be an IEC 61850 Server and the Registry components will be introduced to the system an IEC 61850 Client and Server. For simplicity reasons, as being the dominating standard of the whole platform, the IEC 61850 Standard data model main principles are adopted at the Semantic Middleware data model.

3.1.1 Standard Overview

The IEC 61850 international standard, drafted by substation automation domain experts from 22 countries, seeks to tackle the aforementioned situation. This standard takes advantage of a comprehensive object-oriented data model and the Ethernet technology, bringing in great reduction of the configuration and maintenance cost. Unlike its predecessor, the Utility Communication Architecture protocol 2.0 (UCA 2.0) [12], the IEC 61850 standard is designed to be capable for domains besides substation automation. To make the new protocol less domain dependent, the standardization committee endeavored to emphasize the data semantics, carving out most of the communication details. This effort, however, could result in difficulties in understanding the standard.

3.1.2 IEC 61850 Semantic data modeling

Data semantics provided by IEC 61850 are closely related to the functionalities of devices in utility subsystems such as DERs. IEC 61850 data models are based on object-oriented modelling of process data semantics required for the power system automation.

Figure 3 shows relationships between IEC 61850 data model classes. Top parent class is the Server which is hosted by physical device, i.e., the controller part of the device which is also called the Intelligent Electronic Device (IED). The Server consists of one or more Logical Devices (LDs), i.e., virtual representations of devices intended for supervision, protection or control of automated systems. LDs are created by combining several Logical Nodes (LNs) which represent various device functionality interfaces. LNs are crucial parts of IEC 61850
data semantics. First letter in LN notation designates functionality group (e.g. D - distributed energy resource, W - wind power plant, M - measurement), while the rest of LN notation reflects function name (e.g. DPVM – photo voltaic module, WROT – wind power plant rotor, MMXU – measurement unit). Data Objects (DOs) are groups of Data Attributes (DAs). DAs are IEC 61850 system endpoints and are also granular LN elements. Both DOs and DAs can be recursively nested. A typical DO instance consists of at least three DAs: val (the data value), q (the data quality) and t (the data timestamp).

Common data classes enable creating information models for domain-specific namespaces used by IEC 61850 application services. From application-level perspective, DAs are categorized according to specific functional use (e.g. control, configuration, measurement). Hence, DAs are designated by Functional Constraint (FC) indicating use category. Also at application-level, references to DOs and its subtypes can be organized into Data Set (DS) which are used by IEC 61850 client application in order to optimize communication bandwidth usage.

![Information modeling scope](image)

**Figure 3 – IEC 61850 data class model**

### 3.1.3 IEC 61850 Data-exchange services

The ACSI is a novel paradigm, introduced by IEC 61850, for describing data exchange procedures in utility subsystems such as substations. ACSI model classes define abstract information services used for vertical and horizontal communication among IEC 61850 devices. ACSI is not a protocol but a method to tie IEC 61850 abstract services to application layer protocols such as MMS or DPWS. Within OS4ES only vertical communication services will be utilized. Horizontal communication services which are commonly used in substation automation domain, i.e. GOOSE are out of scope of this project.
The ACSI model classes enabling vertical communication are as follows:

- **Association model**: Service for managing bi-directional connection-oriented information exchange between client and server.
- **Server/LD/LN/DO/DS models**: Services for browsing and editing information model and reading/writing data values.
- **Setting-Group-Control-Block model**: Mechanism for switching between several predefined values for one or more DOs representing parameters values only.
- **Control model**: Service enabling sending command, i.e., changing state of server process.
- **Log-Control-Block model**: Service for storing historical DO values at server level.
- **Report-Control-Block models**: Services enabling event-driven information exchange. It defines mechanism for reporting DO value changes to the clients based on publish/subscribe paradigm.

These ACSI model classes can be used as standardized information interfaces for devices which are implemented as IEC 61850 servers. Thus, any IEC 61850 enabled client software can take full advantage of their remote control.

### 3.1.4 Managing and engineering IEC 61850 systems

The engineering process for IEC 61850 systems is based on exchange of XML documents which are formatted according to the *System Configuration description Language* (SCL). There are several SCL document types depending if they describe a device or the integrated system itself. The engineering process based on SCL document exchange is relatively static and most commonly used for substation automation systems where communication system and electric network topology are predefined. Smart Grid is envisioned as dynamic and constantly changing environment with intermittent power resources and power grid actors which can be producers and consumers at the same time (i.e., prosumers). These issues are not manageable with current IEC 61850 engineering capabilities and require an alternative approach that is capable of handling Smart Grids integration principles.

### 3.1.5 ACSI middleware requirements

The fundamental ACSI middleware requirements include the following:

- Distributed application design;
- Real-time performance capability;
- Device-level integration;
- Event-driven data exchange;
- Standards-based and platform-independent.

However, novel Smart Grid control models such as VPPs and microgrids introduce a new set of application design requirements which surpass the aforementioned list.
These have been identified as follows:
- Service-oriented application design;
- Performance scalability;
- Zero-configuration integration;
- Dynamic system management;
- Short time-to-market.

3.1.6 Middleware Concerning IEC 61850 Protocol Processes and Functionalities

The main IEC 61850 Communication processes and functionalities that will be also served through the Middleware Component are analyzed in this chapter.

**Two Party-Application-Association (TPAA)**

A two-party application association type shall provide a bi-directional connection-oriented information exchange. The application associations shall be reliable and the information flow shall be controlled end to end. Reliable means that the connection on which the application association relies provides measures to notify reasons for non-deliverance of information in due time. End-to-end flow control means that sources of information do not send more information than the destination can buffer.

The services for associate, data exchange, and association release of the two-party application association class is depicted in the following figure.

![IEC 61850 Client and Server normal operation](image)
The abort service for the two-party application association class is depicted in Figure 5.

![Figure 5 – IEC 61850 Client or Server Association Aborting](image)

### 3.2 IEC 61970(CIM) Standard

The role of the CIM Communication Standard in the OS4ES System focuses on data model served by the Smart Grid Applications developed to represent any external actor to the system, Aggregators, DSOs, ESCOs, etc.

#### 3.2.1 Communication Standard Description

The common information model (CIM) is an abstract model that represents all the major objects in an electric utility enterprise typically involved in utility operations. By providing a standard way of representing power system resources as object classes and attributes, along with their relationships, the CIM facilitates the integration of Energy Management System (EMS) applications developed independently by different vendors, between entire EMS systems developed independently, or between an EMS system and other systems concerned with different aspects of power system operations, such as generation or distribution management.

International standard IEC 61970 defines application programming interfaces (APIs) for Energy Management Systems (EMS) commonly used in utility control centers.

Most important document in IEC 61970 series is IEC 61970-301 which defines the Common Information Model (CIM), i.e., power system semantics required by various power utility applications. CIM is based on using Unified Modeling Language (UML) to describe real world objects and information entities used for utility’s operational systems such as EMS or Supervisory Control and Data Acquisition (SCADA) systems. CIM specifies semantics and structure for data which represent power system resources (e.g. substation, transformer, switch), their attributes (e.g. value of the active power) and their relationships (e.g. transformer has a two windings). CIM data classes, attributes and their relationships are divided into several packages which define logical views on functional aspects of specific utility operation. Applications using CIM usually combine several packages.
IEC 61970 offers comprehensive data model intended for independently developed applications with soft real-time requirements. First IEC 61970 editions were closely related to transmission system, but requirements for distribution automation lead to development of data model extensions for distribution system.

3.3 XMPP Communication Protocol

The final technology or standard choice concerns the Semantic Middleware Communication Protocol, the rationale of the selection has been detailed in OS4ES deliverable 3.1. The selected protocol is XMPP that allows utilization of the already existing IEC 61850 technology by deploying different mapping for the application level services based on MMS protocol. Furthermore, it allows utilization of cloud-based and expendable architecture of XMPP systems making it a perfect candidate for the Smart Grid expansion.

3.3.1 Overall description

The Extensible Messaging and Presence Protocol (XMPP) is an open eXtensible Markup Language (XML) protocol specified by the Internet Engineering Task Force (IETF) for near-real-time messaging, presence, and request–response services. The protocol applications are based on extensions to the basic XMPP protocol, and are:

- The creation of multi-user chat rooms, similar to Internet Relay Chat systems
- Application-layer publish/subscribe systems, where the clients are the interacting parties and the server is the router of the messages

XMPP was originally developed for instant messaging applications in the project Jabber (http://xmpp.org/xmpp-protocols/xmpp-core/), and it has characterized by the large number of protocol extension it received. E.g. XMPP has been extended to support voice and video communications. The main implementation of the protocol aims at scalability, having the goal of delivering data from a large number of connected devices to many user applications. The functionalities of XMPP include XML streaming, encryption using Transport Layer Security (TLS), authentication, Unicode support, and information about publishers' and subscribers' presence in the network.

XMPP was introduced to get past the limitations of the traditional architecture of the internet, which is based on cumbersome HTTP-like protocols and request/response paradigm. XMPP introduces stateful communication that can keep track of sessions between the server and the clients. XMPP is based on the client/server paradigm. Two applications, identified by a unique identifier called Jabber Identification (JID), can communicate by connecting to a XMPP server and exchanging XMPP messages, which are routed through the servers down to the message's destination.
XMPP is a communication protocol enabling two entities (XMPP clients) to exchange pieces of XML data, called stanzas. The XMPP clients are not directly connected together. They are instead connected to one or several intermediary entities (XMPP servers) making the routing of the stanzas.

Several servers can connect together in order to enable inter-domain communications between XMPP clients. The principle is that each client initiates a connection to its XMPP server (with TCP by default) and creates a logical channel called a stream. Servers connect together in a similar way so that a stanza sent by an XMPP client can flow through the server(s) up to the recipient XMPP client.

### 3.3.2 XMPP Security Layer

TLS can be used between XMPP Clients and the XMPP server. The specific threats countered at the transport layers level, as long as the XMPP server is a trusted hop, include:

- Unauthorized access to information through message level authentication and encryption of the messages
- Unauthorized modification (tampering) or theft of information through message level authentication and encryption of the messages

An additional simple authentication mechanism allows the XMPP server to guarantee the identity of the connected XMPP client. XMPP servers can be isolated (e.g., on a company intranet), and secure authentication (SASL) and encryption (TLS) have been built into the core XMPP specifications.
4 OS4ES Middleware Reference Implementation

Based on the analysis and outcomes of previous and concurrent deliverables of the project, the following technology selections were made regarding the OS4ES platform components:

- OS4ES aims to be interoperable with any legacy DER system. The main components of the OS4ES middleware (the core, conversion layer and registry) will natively use an extension of the IEC 61850 data model. This provides natural support for DER systems compatible with the IEC 61850 protocol. In order to communicate with other DER systems as well, the OS4ES middleware will provide the necessary conversion functionality. For the reference implementation this conversion functionality will at least be able to handle DER systems communicating through IEC 60870-5-104 transmission protocol.

- The Registry Server Farm of the OS4ES platform will be able to interact with the remaining system through an IEC 61850 Server and Client configured to cover different needs of the OS4ES system and will make use of the same extended IEC 61850 data model of the DER components.

- The SmartGrid applications will interact with the middleware using an Application Programming Interface (API) that may be based on CIM (IEC 61970). The middleware specifications remain abstract on this topic, until this API has been co-decided with the SG application specification definition.

- The selected communication protocol that will facilitate all the essential communication needs of the system components will be the Extensible Messaging and Presence Protocol (XMPP).

The Semantic Middleware of the OS4ES platform is the platform component that will have to perform the entire realizable message transferring between the separate components of the system and to ensure reliable and secure message delivery.

4.1 Reference Conceptual Architecture

The Semantic Middleware will inherit, for interoperability reasons, the same extended IEC 61850 data model used in the converted IEC 61850 Servers communicating with the DER components and the Registry of the OS4ES system. Thus, the mapping elements used for the data model translation between the Middleware data model and the data model of the DER components and the Registry are redundant and can be eliminated from the overall architecture. What is more, in order to serve all the communication requirements of the IEC
61850 standard, an IEC 61850 Client will be hosted in every Middleware Client intended for the Smart Grid applications and will be responsible for triggering any needed communication with the other OS4ES components for data acquisition or energy requests sending reasons. A CIM to IEC 61850 Mapping element will be needed for the translation of the incoming from the Smart Grid applications CIM messages to IEC 61850 requests targeting corresponding IEC 61850 Servers.

The Middleware Conceptual Architecture is provided in the following figure (Figure 6) after importing to the schema all the selected communication technologies and protocols, as well as the changes described above.

As shown in the Reference Conceptual Architecture the Core element of the Middleware system, the Middleware Server Farm, is replaced with an OS4ES-XMPP Server Farm which is an XMMP Server Farm with extended functionality for the Middleware needs. It may consist of one or multiple XMPP domains (different XMPP Server deployments will be analysed in section 5.1). All the separate components interfacing with the OS4ES System are enclosed inside OS4ES-XMPP Clients (previous Middleware Clients), that are extended XMPP Clients and interact with the Core System through the TCP/IP network protocol used as a bit-pipe for end-to-end connectivity that will be provided by the telecommunication network. DER System Components are interfacing with the Middleware though an IEC 61850 Server. For DER systems that do not natively use this protocol, the middleware will provide conversion functionalities as shown in Figure 6 for the case of a DER system using IEC 60870-5-104. A similar approach will be followed in order to accommodate DER systems using alternative protocols. In the most special case of the Smart Grid applications, the enclosing “extended” XMPP Client is also equipped with an IEC Client and a CIM to IEC 61850 Mapping element as part of an Application Programming Interface (API) exposed to the Smart Grid application.

The IEC 61850 Specific Communication Service Mapping (SCSM) over the XMPP for the communication of the various IEC Servers and Clients is realized with the IEC 61850-8-2 Standard Draft. The detailed integration of the IEC 61850-8-2 standard to the Semantic Middleware of the OS4ES platform is described in chapter 4.3.

Finally the CIM to IEC 61850 Mapping element of the Smart Grid exposed API will be described in chapter 4.4 and will be based on the IEC 62361 part 102 CIM 61850 Harmonization Mapping Specification.
Figure 6 – Reference Middleware Conceptual Architecture
4.2 Reference Component Diagram

A more analytic Component Diagram of the Middleware with the software layers of each Component is presented in the figure below (Figure 7).
4.2.1 OS4ES-XMPP Server Farm

The OS4ES-XMPP Server Farm constitutes the Control Layer of the OS4ES System, as the responsible layer for handling all the system exchanged messages. The OS4ES-XMPP Server Farm can consist of one or more XMPP Server elements. The typical XMPP Server Farm design includes a core router/server to which other parts of the Server Farm connect in order to form together one “Server”.

The main functionality of the OS4ES-XMPP Server Farm is the routing functionality. More specifically, it should be able to accept several components connections and transfer messages (stanzas) between them in a transparent way; the message sender need not care about the whereabouts and the communication passage to the recipient. This functionality is inherited and mainly offered by the XMPP communication technology, where the XMPP Server acts as a service bus with no business logic.

A set of extra capabilities should be offered on the top of the main functionality of the Server Farm, as the Conversion and Control Layer requirements indicate. For every capability a separate Software Component is designed to be implemented. The functionality and specifications of each Component is described below.

**Message Validation Component**

A Message Validation process should be able to be conducted for every message that is delivered to the OS4ES-XMPP Server Farm in order to avoid verify message correctness. This validation process will include the check process of the right form of the message based on the type of the message, as well as its expected structure, checks that will ensure the compatibility of the message payload. Finally the message recipient address (Jabber ID) is to be confirmed, concerning the validity of the address, the presence status of recipient, as well as the recipient capabilities compliance with the message request. In XMPP communication protocol the routing process is an application-agnostic process, so no routing rules are expected inside an XMPP message expect to (recipient) and from (sender) information.

**Message Multicast Component**

The Message Multicast Component is responsible for managing messages that are exchanged between a set of users/clients. The Multi-User Chat protocol (specified in [XEP-0045](https://xmpp.org/extensions/xep-0045.html)) will be utilized in this component. This specification defines an XMPP protocol extension for multi-user text chat, whereby multiple XMPP users can join a virtual chat room for the exchange of messages between multiple participants, similar to Internet Relay Chat
(IRC). In addition to standard chat room features such as room topics and invitations, the protocol defines a strong room control model, including the ability to kick and ban users, to name room moderators and administrators, to require membership or passwords in order to join the room, etc.

**Security Layer**

A discrete software layer is intended for managing and ensuring the Security and Credibility of the whole Semantic Middleware Components. Two Components are delivering different security aspects, the Authentication/Authorization and the Certificate/Encryption Component.

**Authentication/Authorization Component**

The Authentication/Authorization Component makes use of the Authentication service, also defined in RFC3920, provided by the XMPP communication technology for secure application development. In this case, the authentication service ensures that entities attempting to communicate over the network are first authenticated by a server, which acts as a kind of gatekeeper for network access. The authentication flow that will be provided by this Component will be able to assess not only the access rights of the sender of the message to OS4ES the whole platform, but “his” permissions to the requested service or API of the recipient.

What is more the Authentication process will be conducted in two steps. The first step will be realized as an end-to-middle Authentication process between an XMPP Client and an XMPP Server or between XMPP servers. The second step, though, will be conducted outside the XMPP communication protocol as an end-to-end authentication process between IEC 61850 Client and Server instances.

**Certificate/Encryption Component**

A channel encryption service, provided by the XMPP communication protocol and also defined in RFC3920, is exploited in the Certificate/Encryption Component. This service provides encryption of the connection between a client and a server, or between two servers. Through this component, the integrity between all instances is protected, as well as the confidentiality between the IEC 61850 client and server instances.
API Management Component

Though the Semantic Middleware Core Component, the OS4ES system nodes will be able to provide various APIs (Application Programming Interfaces) to the other system nodes. This interface targets the Smart Grid applications which are software components that need to interface with an API, rather than physical DER system controllers that communicate through the IEC 61850 protocol (or equivalent). These APIs will be represented to the OS4ES platform through IEC 61850 Servers that will be hosted within the dedicated node XMPP Client. There are two implementation alternatives for such architecture.

- The API is based on an IEC Server in order to expose its services. This API can be used by IEC 61850 clients to send their requests;
- The API is compatible with the CIM (IEC 61970) data model and can react to respective requests. A “CIM to IEC 61850 Mapping” element will reversely translate the IEC 61850 requests, retrieved from the exposed IEC 61850 Server provided by an extended XMPP Client, and forward it to the OS4ES node API. Any answer will again be transformed to an IEC 61850 reply from the “CIM to IEC 61850 Mapping” element and returned to the XMPP Server Farm.

The API Management Component will provide certain functionalities concerning this API management, as automating and controlling the connections between the APIs and the components that use them, monitoring API performance and providing analytics concerning the APIs health and usage, and protecting the APIs from misuse by wrapping it in security procedures and policies. API Management Component should support all APIs running a single process, as well as APIs distributed over multiple processes or even machines (and subsequently different XMPP Clients).

QoS Component

This component will measure and monitor Quality of Service metrics (e.g. end-to-end communication latency, dropped message rates, network throughput) to evaluate the performance of the OS4ES communications and ensure that they take place within the requirements outlined in Deliverable 2.1 of the project. Optionaly it may employ functionalities to improve communication performance over a distributed XMPP network, such as a load balancing component to better distribute computational/communication loads.

*Extending XMPP reliability using advanced XMPP capabilities*

Although use of XMPP can seem very reliable, a basic XMPP system has characteristics that are not reliable in some situations. There are a number of ways in which components of an XMPP system can fail, that would lead to lack of reliability.
Components that should be considered are:

- **XMPP Server.** If an XMPP server fails, communication with the associated users will be prevented. Also, if the server fails while messages are in transit through the server, they will be lost.
- **Links.** XMPP messages are sent down a TCP stream (without acknowledgement). This means that if a link fails (either between client and server or between two servers), messages may be lost.
- **XMPP Client.** There could be a failure in an XMPP client involved in communication, or the hardware on which it is running.
- **User.** There could be a failure by a participating user, for example not noticing a critical message.

The importance of each of these types of failure will depend on the target environment, risks, and criticality of messages.

**Server Failure & Clustering**

XMPP Clustering is not standardized, so clustering support is vendor specific. Support by some products is much better than others, and this capability should be examined with care when selecting an XMPP server for a high reliability deployment.

When an XMPP server crashes, messages will generally be lost. XMPP servers, unlike message switches, do not write messages to permanent storage before sending them on (much like an IP router). It is desirable that XMPP servers switch high volumes of messages very quickly, so this is an architectural decision (and not a design error). The consequence of this is that there are situations where messages can be lost on server failure, although in practice this will be very rare. Design changes could reduce risk of losing messages in transit (but not eliminate the risk), and these changes would have negative performance impact. Dealing with this (low risk) message loss is covered later, using end to end acknowledgements.

What will be used in the Semantic Middleware will be a peer to peer cluster architecture, with all cluster nodes directly connected to each other, with updates synchronized to all cluster nodes. A common configuration will have all cluster nodes connected by a fast LAN. Wide area clustering will be also supported, where cluster nodes are connected over "Internet" quality links. This is important to support reliability in a distributed survivable environment.
Link Failures & XEP-0198

XMPP messages, both between client and server and between two servers, flow down either one or two TCP streams with messages unacknowledged. This is an efficient approach, and appropriate for high speed low latency message switching provided by XMPP. A consequence of this design is that messages that have been "sent", may be lost if the network or destination fails and the sender will be unaware of the loss. In deployments where all clients connect to a single server over a LAN, such message loss is very rare and so this protocol unreliability is not of operational concern. However, where connections are made over the Internet or less reliable links, some message loss is likely.

In environments where any message loss is of operational concern, or where network reliability is such that link failures are to be expected, the standard protocols do not provide sufficient reliability. The solution is XEP-0198: Stream Management. XEPs (XMPP Extension Protocols) provide standardized extensions to the core XMPP protocols. The figure below shows how it works. Essentially, messages may request an acknowledgement, which will cause an acknowledgement to be sent. This is done on the normal XMPP streams, so no (synchronous) hand-shaking is introduced.

Where a message is not acknowledged, a server can automatically resend the message (over a new connection). This will often happen in conjunction with connection reset and re-establishment. This automatic resend is essential for reliability. For a client, and alternative strategy is to tell the user about un-acknowledged messages, and let the user take appropriate action, which may or may not be message re-send.

This basic resend strategy solves the reliability problem, but will also cause resend when the acknowledgements are lost, which will lead to message duplication. The solution to this is the "resume" option of XEP-0198, where the server records messages that have been successfully received. On resumption the sender can use this to determine which unacknowledged messages have actually been sent.

![Figure 8 – XMPP Stream Management Extension](image)
End to End Acknowledgements

If you need end to end reliability, you need end to end acknowledgement. There are two types of end to end acknowledgement that can be used with 1:1 XMPP messages:

- Delivery Report. This confirms that the message has reached the recipient’s client. It can be generated automatically, and will usually come back very quickly. This addresses failures in the recipient’s XMPP client and failures in intermediate servers and links.
- Read Receipt. This confirms that the message has been read by the recipient, and requires manual action by the recipient. This addresses recipient user-related failures.

Read receipts can be useful where it is critical that the recipient gets a specific message (e.g., an order to do something). They would be very awkward if used for large numbers of messages, so this needs to be an option reserved for messages where it is critical to be certain that the message has been read. End to end acknowledgements are defined in XEP-0184: Message Receipts, which will be integrated in order to ensure successful message delivery.

DNS Component

The Domain Name System (DNS) Component is responsible for handling all the route rules for all the XMPP Clients connections and keeping track of their presence as well as their contact lists. Multiple connected XMPP Servers information is also handled and utilized by this Component.

Middleware Database

The XMPP Server Farm is equipped with a Database for storing any required operational information of the Farm, as well as of the various Software Components. This information can include routing tables, contact lists, acceptable XML schemas, etc.

4.2.2 OS4ES-XMPP Clients

The set of all the OS4ES-XMPP Clients – including the internal mapping/conversion functionalities - of the Semantic Middleware constitute the Conversion Layer of the OS4ES platform. This layer ensures the smooth communication of the OS4ES components (Registry, DERs, Smart Grid Applications) with the rest of the OS4ES system.
As soon as an OS4ES-XMPP client is registered to the OS4ES-XMPP Server Farm, it is able to receive and send any XMPP message. For the OS4ES components that have already at their disposal an IEC 61850 Server, for instance a DER system, and both IEC 61850 Clients and Servers, as in the case of the Registry, the XMPP Client role is limited to the appropriate wrapping of an IEC 61850 Request before forwarding it to the XMPP Server and the unwrapping of an XMPP Message to an IEC 61850 Reply, based on XER encoding rules as indicated from the IEC 61850-8-2 standard (see chapter 4.3). In case of DER systems utilizing alternative protocols, a software component embedded in the OS4ES-XMPP Client will take care of the protocol conversion (e.g. IEC 60870-5-104 as shown in Figure 7), so that a common interface (IEC 61850 based) will be exposed to the XMPP servers. This wrapping and unwrapping mechanism will be conducted by the SCSM-8-2 (MMS to XML Messaging) software component of the XMPP Clients, addressing the DER systems and the Registry.

In case of the Smart Grid Applications, the hosting OS4ES-XMPP Clients are “extended” XMPP Clients in the sense that they provide along with the SCSM-8-2 (MMS to XML Messaging) software component, an IEC 61850 Client and CIM Interface API1 software components. The role of the CIM Interface API is in fact the gateway of the Smart Grid Application to the whole OS4ES platform. A Smart Grid Application should integrate the CIM Interface API provided by the “extended” XMPP Client in order to forward its request. The forwarded requests are then handled by the API where message translation and protocol conversion processes from the CIM to the IEC 61850 data model take place within the “CIM to IEC 61850 Mapping” mechanism included inside the CIM Interface API. The just created IEC 61850 compliant request is then delivered to the embedded IEC Client software Component of the “extended” XMPP Client which is responsible for serving it correctly to the wrapping mechanism of the SCSM-8-2 (MMS to XML Messaging) software component and finally forwarding it to the OS4ES-XMPP Server Farm. The reply on this initial request will follow the exact reverse procedure until it reaches the appropriate Smart Grid Application.

The whole message transferring mechanism end-to-end is described in more detail in chapter 4.2.3.

CIM Interface API (SG Application)
The CIM Interface API Component exists at the OS4ES-XMPP Clients intended to represent Smart Grid applications. This interface component constitutes, in fact, the entrance point of the Smart Grid application to the whole OS4ES system. A set of services will be exposed from the CIM Interface API and will be consumed by the Smart Grid applications. The format of

1 In the middleware reference implementation, this interface will be targeted toward the data model used by the Smart Grid applications. If they do not use the IEC 61970 CIM but a proprietary data model for instance, this converter will be adapted accordingly.
this service will be generic, intended to cover all the operational Use Cases that the OS4ES System will support and will be compatible with the CIM communication data model. Any received CIM request will be transformed and translated to the IEC 61850 corresponding request and subsequently will be delivered to the hosting OS4ES-XMPP client, in order to be handled by the rest of the OS4ES system.

4.2.3 Message Transferring Process

As mentioned previously, the Message Routing Process is the most important process that the Semantic Middleware component should execute. What is worth mentioning again is that the main part of this process, which is conducted in the Core/Control layer of the Middleware, is message agnostic, as well as recipient and sender agnostic, i.e. the same exact procedure is followed in all the cases of message transferring, except from some small variations depending on the type of sender and recipient.

More specifically, in case the established communication is between the components that possess an IEC 61850 Server or an IEC 61850 Client, i.e. a DER System or Registry, the message transferring process is illustrated in the following UML Sequence Diagram (Figure 9).

The IEC 61850 Client initiates the communication\(^2\) by sending an IEC 61850 Request to an IEC 61850 Server based on the JabberID of its hosting OS4ES-XMPP Client. The request is forwarded to the OS4ES-XMPP Client which is subsequently wrapped to an XML Message, according to the XER Encoding rules. The current XER request is then routed through the OS4ES-XMPP Server to the OS4ES-XMPP Client with the message recipient JabberID. The second OS4ES-XMPP Client, which represents also the wanted IEC 61850 Server inside the Middleware, un-wraps the XER Message and translates it to the initial IEC 61850 Request. The IEC 61850 Server generates the corresponding IEC 61850 Reply and hands it back to its OS4ES-XMPP Client, who wraps it again in XER format. The message reply recipient ID coincides with the JabberID of the OS4ES-XMPP Client hosting the IEC 61850 Client. The XER Reply is finally routed through the OS4ES-XMPP Server to the OS4ES-XMPP Client that initiated the communication. The OS4ES-XMPP Client hosting the IEC 61850 Client un-wraps the XER Reply to an IEC 61850 Reply and proceeds with its delivery to the IEC 61850 Client, which finalizes the communication process.

\(^2\) In some cases the communication may be initiated by an IEC 61850 server with a client as the recipient, but this fact does not alter the fundamentals of the communication sequence diagram/pattern shown in Figure 9.
The whole communication process slightly differentiates when the triggering source is a Smart Grid Application. The following UML Sequence Diagram (Figure 10) illustrates the whole procedure.
Figure 10 – UML Sequence Diagram of Message Routing between a SmartGrid Application and a component with an IEC 61850 Server

The communication initiates now from a SmartGrid application that makes use of an exposed service of the CIM Interface API in order to launch a Request to the OS4ES System. This message request should be CIM compliant, as the supporting data model of the SmartGrid Application Components. Translation of the message to the IEC 61850 data model is realized from the CIM to IEC 61850Mapping software component of the CIM Interface API and the IEC 61850 compliant message is then delivered to the embedded IEC 61850 Client of the SmartGrid Application hosting “extended” XMPP Client. A IEC 61850 Request commences and the same procedure, that was previously described, is followed until the IEC 61850 Reply reaches back the IEC 61850 Client. The reply message is forwarded finally to the
CIM interface API which in turn will translate it to the CIM data model and closes the communication process by delivering the CIM Reply to the initial request message.

4.3 IEC 61850-8-2 Standard Integration

The IEC 61850-8-2 Standard is a part of the IEC 61850 family and describes a Specific Communication Service Mapping (SCSM) over the Extensible Messaging and Presence Protocol (XMPP), providing detailed information on how to create and exchange concrete communication messages that implement abstract services and models specified in IEC 61850-7-4, IEC 61850-7-3, and IEC 61850-7-2. This mapping can be utilized between all kinds of utility Distributed Energy Resource devices and their related management systems, in particular over public networks.

Based on the guidelines of this standard, the seamless communication between the IEC Clients and Servers of the various components of the OS4ES platform will be ensured through the Semantic Middleware.

4.3.1 Mapping Objects and Services

For the client/server services, the principle is to map the objects and services of the ACSI (Abstract Communication Service Interface defined in IEC 61850-7-2) to MMS (Manufacturing Message Specification, ISO 9506) and XMPP.

Used MMS objects

Manufacturing Message Specification (MMS – ISO 9560) specifies a number of MMS objects that could be used in the SCSM. However, there is a certain subset of objects that are required in order to map IEC 61850-7-2, IEC 61850-7-3 and IEC 61850-7-4 (Table 2), that will also be used during the IEC 61850-8-2 integration.

<table>
<thead>
<tr>
<th>MMS OBJECT</th>
<th>IEC 61850 OBJECT</th>
<th>MMS SERVICES IN USE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Application Process VMD</td>
<td>Server</td>
<td>Initiate</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Conclude</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Abort</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Reject</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Cancel</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Identify¹</td>
</tr>
<tr>
<td>Named Variable Objects</td>
<td>Logical Nodes and Data</td>
<td>Read</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Write</td>
</tr>
<tr>
<td></td>
<td></td>
<td>InformationReport</td>
</tr>
<tr>
<td></td>
<td></td>
<td>GetVariableAccessAttribute</td>
</tr>
<tr>
<td></td>
<td></td>
<td>GetNameList</td>
</tr>
<tr>
<td>Named Variable List Objects</td>
<td>Data Sets</td>
<td>GetNamedVariableListAttributes</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>-----------</td>
<td>---------------------------------</td>
</tr>
<tr>
<td>Journal Objects</td>
<td>Logs</td>
<td>ReadJournal</td>
</tr>
<tr>
<td>Domain Objects</td>
<td>Logical Devices</td>
<td>GetNameList</td>
</tr>
<tr>
<td>Files</td>
<td>Files</td>
<td>FileOpen</td>
</tr>
</tbody>
</table>

*Required by ISO 9506 for conformance.

Table 2 – MMS objects and services in use within this SCSM

IEC Client/Server supported services

The client/server services that should be covered from the Semantic Middleware in order to be able to conform to this IEC 61850-8-2 standard are shown in Table 3.

<table>
<thead>
<tr>
<th>IEC 61850-7-2 model</th>
<th>IEC 61850-7-2 service</th>
</tr>
</thead>
<tbody>
<tr>
<td>Server</td>
<td>GetServerDirectory</td>
</tr>
<tr>
<td>Association</td>
<td>Associate</td>
</tr>
<tr>
<td></td>
<td>Abort</td>
</tr>
<tr>
<td></td>
<td>Release</td>
</tr>
<tr>
<td>Logical Device</td>
<td>GetLogicalDeviceDirectory</td>
</tr>
<tr>
<td>Logical Node</td>
<td>GetLogicalNodeDirectory</td>
</tr>
<tr>
<td></td>
<td>GetAllDataValues</td>
</tr>
<tr>
<td>Data</td>
<td>GetDataValues</td>
</tr>
<tr>
<td></td>
<td>SetDataValues</td>
</tr>
<tr>
<td></td>
<td>GetDataDirectory</td>
</tr>
<tr>
<td></td>
<td>GetDataDefinition</td>
</tr>
<tr>
<td>Data Set</td>
<td>GetDataSetValues</td>
</tr>
<tr>
<td></td>
<td>SetDataSetValues</td>
</tr>
<tr>
<td></td>
<td>CreateDataSet</td>
</tr>
<tr>
<td></td>
<td>DeleteDataSet</td>
</tr>
<tr>
<td></td>
<td>GetDataSetDirectory</td>
</tr>
<tr>
<td>Setting Group Control Block³</td>
<td>SelectActiveSG</td>
</tr>
<tr>
<td></td>
<td>SelectEditSG</td>
</tr>
<tr>
<td></td>
<td>SetEditSGValue</td>
</tr>
<tr>
<td></td>
<td>ConfirmEditSGValues</td>
</tr>
</tbody>
</table>

³ This model is not expected to be used for the OS4ES system.
### 4.3.2 Service Message Forming

According to the OSI Reference Model (ISO/IEC 7498-1), that is also followed by the XMPP communication protocol, the ISO Application Communication profile (A-Profile) of a communication system is the set of specifications and agreements of the application, presentation and session layer of a communication functionality. This A-Profile will participate in the Message Forming of the supported services.

More specifically, the application layer of the A-Profile will make use of the corresponding concepts of MMS of the IEC 61850-7-2 services, as defined in the SCSM 8-1, and the MMS end-to-end security which is also a shared mechanism with the SCSM 8-1. There are three

---

<table>
<thead>
<tr>
<th>Service</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GetEditSGValue</td>
<td></td>
</tr>
<tr>
<td>GetSGCBValues</td>
<td></td>
</tr>
<tr>
<td>Report Control Block</td>
<td>Report</td>
</tr>
<tr>
<td>GetBRCBValues</td>
<td></td>
</tr>
<tr>
<td>SetBRCBValues</td>
<td></td>
</tr>
<tr>
<td>GetURCBValues</td>
<td></td>
</tr>
<tr>
<td>SetURCBValues</td>
<td></td>
</tr>
<tr>
<td>LOG Control Block</td>
<td>GetLCBValues</td>
</tr>
<tr>
<td>SetLCBValues</td>
<td></td>
</tr>
<tr>
<td>GetLogStatusValues</td>
<td></td>
</tr>
<tr>
<td>QueryLogByTime</td>
<td></td>
</tr>
<tr>
<td>QueryLogAfter</td>
<td></td>
</tr>
<tr>
<td>GOOSE(^4)</td>
<td>GetGoCBValues</td>
</tr>
<tr>
<td>SetGoCBValues</td>
<td></td>
</tr>
<tr>
<td>Sampled Values (multicast/unicast)(^5)</td>
<td>GetMSVCBValues</td>
</tr>
<tr>
<td>SetMSVCBValues</td>
<td></td>
</tr>
<tr>
<td>GetUSVCBValues</td>
<td></td>
</tr>
<tr>
<td>SetUSVCBValues</td>
<td></td>
</tr>
<tr>
<td>Control</td>
<td>Select</td>
</tr>
<tr>
<td>SelectWithValue</td>
<td></td>
</tr>
<tr>
<td>Cancel</td>
<td></td>
</tr>
<tr>
<td>Operate</td>
<td></td>
</tr>
<tr>
<td>CommandTermination</td>
<td></td>
</tr>
<tr>
<td>TimeActivatedOperate</td>
<td></td>
</tr>
<tr>
<td>FILE transfer</td>
<td>GetFile</td>
</tr>
<tr>
<td>SetFile</td>
<td></td>
</tr>
<tr>
<td>DeleteFile</td>
<td></td>
</tr>
<tr>
<td>GetFileAttributeValue</td>
<td></td>
</tr>
</tbody>
</table>

| Table 3 – Required IEC 61850-7-2 client/server services |

---

\(^4\) This model is not expected to be used for the OS4ES system.
\(^5\) This model is not expected to be used for the OS4ES system.
possible security profiles which are negotiated during the establishment of a two party association (see more details in A.2):
  - AL: end-to-end authentication for all messages
  - A+: end-to-end authentication and session integrity
  - AE+: A+ profile with additionally encryption

For the presentation layer of the A-Profile, the used encoding of the message will be based on the Canonical XML Encoding Rules defined by the ITU X.693 XER, rather than the binary encoding used by SCSM 8.1.

The session layer will manage the two party associations by two communication applications as described at chapter 4.3.5. What is also defined by the IEC 61850-8-1 is that the clause ServiceAccessPoint is always expected inside the session layer, as the one to be responsible for supporting the service. The 61850-8-2 implementation can support one or several ServiceAccessPoints dedicated to the XMPP communication. Each access point shall be defined by one or more JIDs.

The forming of the message is complete as soon as the Transport Profile (T-Profile) of the OSI Reference Model of a communication functionality is included, which concerns the set of specifications and agreements relating to the lower 4 communication layers and are explicitly defined by the XMPP communication protocol.

### 4.3.3 Service Mapping

**Mapping of solicited services**

Most of the client/server services are solicited services defined by three elements: the Request, the positive Response and the negative Response.

<table>
<thead>
<tr>
<th>Parameter name</th>
<th>Request</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>&lt;request parameters&gt;</td>
</tr>
<tr>
<td>Response+</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;response parameters&gt;</td>
</tr>
<tr>
<td>Response-</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;service errors&gt;</td>
</tr>
</tbody>
</table>

All solicited services are implemented by IQ stanzas. The request message is implemented by an IQ stanza of type “set” or “get” and the response message is implemented by a ‘result’ IQ
stanza (see details in Table 6). The direct child element of the IQ stanza, AssociationContext, represents the association established between the client and the server and it contains the payload generated by the presentation layer which includes the XML encoded MMS message (userData) embedded into end-to-end security information (SecurityRequest-PDU). The following example shows the IQ stanzas for the request and the positive response of a solicited service (using the A+ security profile):

- **Request:**
  ```xml
  <iq id="10"
   from="61850app@XMPPServer/ressourceId"
   to="61850server@XMPPServer"
   type="get">
   <AssociationContext id="32:78">
     <SecurityRequest-PDU>
       <signedData>
         <tokenVals>
         ...
         </tokenVals>
       </userData>
       <confirmed-RequestPDU>
         ...
       </confirmed-RequestPDU>
     </userData>
     <authenticator>
     ...
     </authenticator>
   </signedData>
   </SecurityRequest-PDU>
   </AssociationContext>
  </iq>
  ```

- **Response+:**
  ```xml
  <iq id="10"
   from="61850server@XMPPServer"
   to="61850app@XMPPServer/ressourceId"
   type="result">
   <AssociationContext id="32:78">
     <SecurityRequest-PDU>
       <signedData>
         <tokenVals>
         ...
         </tokenVals>
       </userData>
       <confirmed-ResponsePDU>
         ...
       </confirmed-ResponsePDU>
     </userData>
   </signedData>
   </SecurityRequest-PDU>
   </AssociationContext>
  ```
Mapping of unsolicited services

Unsolicited services are implemented as message stanzas of type “normal” (see Table 6 in page 50). The direct child element of the stanza, AssociationContext, represents the association established between the client and the server and it includes the XML encoded MMS message (userData) embedded into end-to-end security information (SecurityRequest-PDU). The following example shows a message stanza for an unsolicited service (with security profile A+):

```xml
<message
from="server@XMPPServer"
to="61850app@XMPPServer/RessourceId"
type="normal">
  <AssociationContext id="32:78">
    <SecurityRequest-PDU>
      <signedData>
        <tokenVals>
          ...
        </tokenVals>
        <userData>
          <unconfirmed-PDU>
            ...
          </unconfirmed-PDU>
        </userData>
        <authenticator>
          ...
        </authenticator>
      </signedData>
      </SecurityRequest-PDU>
  </AssociationContext>
</message>
```

4.3.4 Usage of XMPP presence

Two-party associations shall be monitored by using the presence information of the IEC 61850 entities.
During the Associate request, the IEC 61850 client shall subscribe to the presence of the IEC 61850 server (if not already done before). The server may also subscribe to the client presence. An entity detecting the unavailable presence status of its associated peer shall generate a local Abort.

### 4.3.5 Association Services

The mapping of the most important Association services is described in this chapter.

**Associate**

During the Associate service, both the calling and called entities shall generate an AssociationID that is locally unique within the pairing of JIDs. The first part of the AssociationID is calculated and transmitted by the calling entity. The called entity then calculates the second part and returns the final AssociationID to the calling entity. This AssociationID shall represent the MMS context and shall be part of all exchanged messages between the two associated entities.

The end-to-end security handshake also takes place during the Associate service (see A.2). The ACSI associate request service shall map directly to the MMS initiate request service as specified in Table 4. The ACSI associate response service shall map directly to the MMS initiate response service. The ACSI associate response – service shall map to the MMS initiate service result (−). The ACSI ServiceErrors shall map to Error Class/Error Code as detailed in IEC 61850-8-1. ACSI ServiceError values that do not appear in the table are not mapped.

<table>
<thead>
<tr>
<th>Associate parameters</th>
<th>MMS service or parameter</th>
<th>Constraint</th>
</tr>
</thead>
<tbody>
<tr>
<td>Request</td>
<td>initiate-Request service</td>
<td></td>
</tr>
<tr>
<td>ServerAccessPointReference</td>
<td>Presentation Addresses a</td>
<td></td>
</tr>
<tr>
<td>Authentication parameter</td>
<td>ACSI AuthenticationValue b</td>
<td>Mandatory</td>
</tr>
<tr>
<td>AssociationId</td>
<td>AssociationID c is transmitted within the session layer</td>
<td></td>
</tr>
<tr>
<td>Result</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Response–</td>
<td>Initiate-ErrorPDU service</td>
<td></td>
</tr>
<tr>
<td>ServiceError</td>
<td>ServiceError (ErrorClass, Error) See IEC 61850-8-1</td>
<td></td>
</tr>
</tbody>
</table>

| a | Presentation Address corresponds to the JID of the receiving entity |
| b | How the AuthenticationValue is transported is defined by IEC 62351-4 ed 2.0 |
| c | AssociationID is calculated by the two associated entities as defined above |

Table 4 – Mapping of ACSI Associate service to MMS
Abort

The ACSI abort request service shall map directly to the MMS abort request service. The ACSI Abort Indication service shall map directly to the MMS abort indication. The reason code values shall be as defined in ISO 9506 (all parts).

Release

The ACSI release request service shall map directly to the MMS conclude request service as specified in Table 5. The ACSI release response (+) shall map directly to the MMS conclude response. The ACSI release response (−) shall map directly to the MMS conclude error. The ACSI ServiceErrors shall map to Error Class/Error Code as detailed in IEC 61850-8-1. ACSI ServiceError values that do not appear in the table are not mapped.

<table>
<thead>
<tr>
<th>Release parameters</th>
<th>MMS service or parameter</th>
<th>Constraint</th>
</tr>
</thead>
<tbody>
<tr>
<td>Request</td>
<td>Conclude-Request service</td>
<td></td>
</tr>
<tr>
<td>AssociationId</td>
<td></td>
<td>AssociationID is transmitted within the session layer ^a</td>
</tr>
<tr>
<td>Response+</td>
<td>Conclude-Response service</td>
<td></td>
</tr>
<tr>
<td>AssociationId</td>
<td></td>
<td>AssociationID is transmitted within the session layer ^a</td>
</tr>
<tr>
<td>Result</td>
<td>Conclude-ErrorPDU service</td>
<td></td>
</tr>
<tr>
<td>Response–</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ServiceError</td>
<td>ServiceError (ErrorClass, Error)</td>
<td>See IEC 61850-8-1</td>
</tr>
</tbody>
</table>

^a The AssociationID calculated during the Associate service is transmitted with all messages

Table 5 – Mapping of ACSI Release service to MMS

4.4 CIM - 61850 Harmonization Mapping Integration

Despite that IEC 61850 and IEC 61970 both represent power system domain vocabulary; there are some crucial issues which complicate their seamless integration. These issues are results of independent standardization efforts of various IEC’s working groups. This problem has been only partly addressed by introducing reference architecture for IEC Technical Committee 57 (TC57). Harmonization problem has been identified in and has been a part of several research projects ever since.

IEC 61850 data model is a result of elaborate domain analysis. Despite introducing classes and inheritance, the final model is not entirely object-oriented. This is caused by specific device-oriented semantics required by centralized applications used in substation automation. IEC 61850 standard extensions, such as IEC 61850-7-420 introduced
requirements to use IEC 61850 for decentralized automation systems such as VPPs. One of the model merging approaches is based on development of IEC 61850 UML model. Because of different mandatory attributes, attribute levels and specializations it has been shown that without changing both standards it is impossible to realize bi-directional mapping of data by using aforementioned approach. EPRI’s harmonization efforts have resulted in introduction of required customizations of IEC 61850. Only recently, ontology integration has been presented as an applicable approach, albeit with significant limitations.

The CIM to IEC 61850 Harmonization Mapping may be required in order to implement the CIM Interface API component of the XMPP Clients that represent in the OS4ES system the Smart Grid applications. The actual implementation will heavily depend on the data model chosen for the implementation of the Smart Grid applications in order to ensure smooth execution of the lab/field trials. Furthermore, depending on the data model profile instantiated by the Smart Grid applications, an investigation of the optimal approach toward the harmonization will take place. At this point in time, the needed information to make these decisions is not available, hence the outcomes of this decision making process will be documented along with the implementation details in deliverable 6.3 which will document the middleware implementation.
5 Reference Implementation Deployments

There are several ways to deploy the Semantic Middleware components inside the XMPP communication protocol. For instance, OS4ES-XMPP Clients can be hosted in the same or different physical nodes with their corresponding IEC Servers or Clients and the OS4ES-XMPP Server Farm can extend to more than one XMPP domain. The various deployment methods are presented in this chapter in detail.

5.1 Deployment of Middleware OS4ES-XMPP Clients

The preferable method for the deployment of the OS4ES-XMPP Clients of the Semantic Middleware is that it will be hosted in the same physical node (computer) as the OS4ES component (e.g. DER system controller). However, this is not a deployment constraint. The XMPP Client can be hosted at another physical device with the only constraint that it should be able to communicate with the external OS4ES component physical node (DER System, Registry and SmartGrid application) through the network.

Either way, a configuration phase will be necessary during the OS4ES-XMPP Client installation to the physical device that will include the establishment of the communication channel between the OS4ES component and the OS4ES-XMPP client. This configuration phase will slightly differ in the two cases (single machine vs. two machine deployment). However the differences are minor: deployment in a single machine will use the localhost for connection of the two software components; deployment in two machines requires that they are connected to the same sub-network, in this case a simple TCP connection will suffice. The two different deployments methods are illustrated in Figure 11.

![Figure 11 – Two OS4ES-XMPP Clients deployment methods](image)

Copyright OS4ES project Page 44 of 51
5.2 Deployment of XMPP with one or more XMPP domains

The Semantic Middleware can be implemented making use of one or multiple domains in a seamless way for the OS4ES external components. The final implementation choice will be made based on the reliability and the scalability needs of the system.

The existence of one XMPP domain will ensure the simplicity of the Middleware Core Part implementation. The same deployment topology with an instance of a facility domain using an one domain XMPP communication environment, as displayed in Figure 12, will be followed in this case.

![Figure 12 – XMPP Deployment within an XMPP Domain](image)

Multiple XMPP domain usage is possible with the use of federation. Federation is a capability given to XMPP Clients located in different domains, and therefore connected to different XMPP servers, to communicate with each other. The XMPP server resolves the domain, indicated in the destination field in a stanza, starts a XML stream to the XMPP server of the target domain, and uses that connection for forwarding the message as shown in Figure 13.
The usage of federation in a multi-domain communication is illustrated in Figure 14.

Figure 13 – Concept of federation in XMPP

Figure 14 – Two domain federation
6 Conclusions

We consider that the chosen Semantic Middleware architecture will sustain the whole OS4ES platform operation. By supporting both IEC 61850 communication standard and CIM communication data model, the Semantic Middleware will be able to provide compatibility to many Smart Grid applications and DER Systems and also increase the number of potential developers which are able to write and maintain components for the OS4ES system.

During the implementation phase of the Semantic Middleware and after the first prototype will be released, the architecture design decisions and technologies and standards selections will be evaluated and readapted if necessary.

Finally, several improving potentials of the operation mechanism of the Semantic Middleware will be taken into account. Additionally, depending on the experiences during the development and first evaluation phase as well as the evolution of the other discussed technologies, further optimizations will be considered.
Annex A.

A.2 Mapping of MMS secure session concept for SCSM 8.2

For the SCSM8-2, the secure session concepts specified by the IEC 62351-4 ed.2 apply to the applicative payload transported over the XMPP protocol. Figure 23 below shows which parts of this payload are concerned by the various security profiles.

The IEC 62351-4 ed.2 defines the abstract structure of the application layer MMS messages including the following specific information for end-to-end security:

- **ClearToken (1):** contains session key management (Diffie Hellman), peer IDs and A-Profile type selection via OID. This structure is dedicated to the initial handshake for all profiles.
- **ConfidentialityParams:** determines confidentiality protection by a possible combination of T-Profile, A-Profile, or VPN and the buffer size for encrypted packets (for AE+). This structure is also dedicated to the initial handshake for all profiles.
- **Authenticator:** contains the digital signature or HMAC.
- **ClearToken (2):** contains peer IDs and A-Profile type selection, to be used all along the secured session.
It may be noted that the association context managed by the session layer is outside of the secured payload.

The figures below show the packet assemblies resulting of the mapping for the SCSM 8-2 of the abstract definitions of MMS and security specific structures. The Figure 24 corresponds to the handshake made during the Associate for all the security profiles. The other request/response packets are shown in Figure 25 for the AL/A+ profiles and Figure 26 for the AE+ profile.

Figure 16 – Initial handshake for AL/A+/AE+ Profiles

Figure 17 – Request / Response messages for AL/A+ profiles

Figure 18 – Request / Response messages for AE+ profile
### B.2 Mapping of MMS services over XMPP stanzas

For each MMS service used in the present SCSM the Table 12 below shows how it is transported over XMPP stanzas.

<table>
<thead>
<tr>
<th>MMSpdu kind</th>
<th>MMSpdu subType</th>
<th>Stanza</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>confirmed-RequestPDU</td>
<td>getNameList</td>
<td>iq</td>
<td>get</td>
</tr>
<tr>
<td>confirmed-RequestPDU</td>
<td>read</td>
<td>iq</td>
<td>get</td>
</tr>
<tr>
<td>confirmed-RequestPDU</td>
<td>write</td>
<td>iq</td>
<td>set</td>
</tr>
<tr>
<td>confirmed-RequestPDU</td>
<td>getVariableAccessAttributes</td>
<td>iq</td>
<td>get</td>
</tr>
<tr>
<td>confirmed-RequestPDU</td>
<td>defineScatteredAccess</td>
<td>iq</td>
<td>get</td>
</tr>
<tr>
<td>confirmed-RequestPDU</td>
<td>getScatteredAccessAttributes</td>
<td>iq</td>
<td>get</td>
</tr>
<tr>
<td>confirmed-RequestPDU</td>
<td>defineNamedVariableList</td>
<td>iq</td>
<td>set</td>
</tr>
<tr>
<td>confirmed-RequestPDU</td>
<td>deleteNamedVariableListAttributes</td>
<td>iq</td>
<td>get</td>
</tr>
<tr>
<td>confirmed-RequestPDU</td>
<td>readJournal</td>
<td>iq</td>
<td>get</td>
</tr>
<tr>
<td>confirmed-RequestPDU</td>
<td>fileOpen</td>
<td>iq</td>
<td>get</td>
</tr>
<tr>
<td>confirmed-RequestPDU</td>
<td>fileRead</td>
<td>iq</td>
<td>get</td>
</tr>
<tr>
<td>confirmed-RequestPDU</td>
<td>fileClose</td>
<td>iq</td>
<td>get</td>
</tr>
<tr>
<td>confirmed-RequestPDU</td>
<td>fileDelete</td>
<td>iq</td>
<td>set</td>
</tr>
<tr>
<td>confirmed-ResponsePDU</td>
<td>getNameList</td>
<td>iq</td>
<td>result</td>
</tr>
<tr>
<td>confirmed-ResponsePDU</td>
<td>read</td>
<td>iq</td>
<td>result</td>
</tr>
<tr>
<td>confirmed-ResponsePDU</td>
<td>write</td>
<td>iq</td>
<td>result</td>
</tr>
<tr>
<td>confirmed-ResponsePDU</td>
<td>getVariableAccessAttributes</td>
<td>iq</td>
<td>result</td>
</tr>
<tr>
<td>confirmed-ResponsePDU</td>
<td>defineScatteredAccess</td>
<td>iq</td>
<td>result</td>
</tr>
<tr>
<td>confirmed-ResponsePDU</td>
<td>getScatteredAccessAttributes</td>
<td>iq</td>
<td>result</td>
</tr>
<tr>
<td>confirmed-ResponsePDU</td>
<td>defineNamedVariableList</td>
<td>iq</td>
<td>result</td>
</tr>
<tr>
<td>confirmed-ResponsePDU</td>
<td>deleteNamedVariableListAttributes</td>
<td>iq</td>
<td>result</td>
</tr>
<tr>
<td>confirmed-ResponsePDU</td>
<td>getNamedVariableListAttributes</td>
<td>iq</td>
<td>result</td>
</tr>
<tr>
<td>confirmed-ResponsePDU</td>
<td>readJournal</td>
<td>iq</td>
<td>result</td>
</tr>
<tr>
<td>confirmed-ResponsePDU</td>
<td>fileOpen</td>
<td>iq</td>
<td>result</td>
</tr>
<tr>
<td>confirmed-ResponsePDU</td>
<td>fileRead</td>
<td>iq</td>
<td>result</td>
</tr>
<tr>
<td>confirmed-ResponsePDU</td>
<td>fileClose</td>
<td>iq</td>
<td>result</td>
</tr>
<tr>
<td>confirmed-ResponsePDU</td>
<td>fileDelete</td>
<td>iq</td>
<td>result</td>
</tr>
<tr>
<td>confirmed-ErrorPDU</td>
<td>informationReport</td>
<td>message</td>
<td>normal</td>
</tr>
<tr>
<td>rejectPDU</td>
<td></td>
<td>iq</td>
<td>result</td>
</tr>
<tr>
<td>cancel-RequestPDU</td>
<td></td>
<td>iq</td>
<td>get</td>
</tr>
<tr>
<td>cancel-ResponsePDU</td>
<td></td>
<td>iq</td>
<td>result</td>
</tr>
<tr>
<td>cancel-ErrorPDU</td>
<td></td>
<td>iq</td>
<td>result</td>
</tr>
<tr>
<td>initiate-RequestPDU</td>
<td></td>
<td>iq</td>
<td>set</td>
</tr>
<tr>
<td>initiate-ResponsePDU</td>
<td></td>
<td>iq</td>
<td>result</td>
</tr>
<tr>
<td>initiate-ErrorPDU</td>
<td></td>
<td>iq</td>
<td>result</td>
</tr>
<tr>
<td>conclude-RequestPDU</td>
<td></td>
<td>iq</td>
<td>set</td>
</tr>
<tr>
<td>conclude-ResponsePDU</td>
<td></td>
<td>iq</td>
<td>result</td>
</tr>
<tr>
<td>conclude-ErrorPDU</td>
<td></td>
<td>iq</td>
<td>result</td>
</tr>
</tbody>
</table>

*Table 6 – Usage of XMPP stanzas*
References

[1] Deliverable D.1.2 “OS4ES system architecture, component requirements and communication infrastructure”.
[2] IEC 61850 Standard Part 7-2: Basic communication structure for substation and feeder equipment – Abstract communication service interface (ACSI)