Interoperability of fast charging station with battery booster

Josef Schindler[1], Venesa Watson[2] und Karl Waedt[3]

Abstract: Before a high proportion of Electric Vehicles (EV) will be deployed, sufficient charging infrastructure for these EVs must be provided. In Germany, this deployment is hindered by the lack of a universal standard for the geometry of the charging plugs, the different charging voltage levels, the structure of data that is provided by the charging station and so forth. Related standards such as ISO/IEC 15118 and ISO/IEC 61850, respectively addresses front-end communication between EVs and common charging stations and defines communication messages for grid automation. Whilst ISO/IEC 15118 does address interoperability between charging infrastructures from different manufacturers, its restriction to front-end communication limits the extent of advanced interoperability. In this paper, we investigate the interoperability on battery boosted charging stations based on OPC Unified Architecture (OPC UA). OPC UA, powered by the OPC Foundation, is the primary interoperability standard for Industry 4.0 (I4.0). OPC UA enables data exchange regardless of the manufacturer of individual components. OPC UA can be implemented in embedded devices and controllers without considering additional “black boxes”, “windows boxes” or gateways in front of the equipment. This level of interoperability provides significant cost-saving for the utilities during all lifecycle phases. It also ensures transparent data acquisition for customers. In this paper, an example of fast charging station with battery booster is used to demonstrate the interoperability feature of OPC UA compared to that of ISO/IEC 15118. Also, their differing structure makes it necessary to consider these charging stations separately from common ones. Therefore, various data representation formats are utilized to show information sharing within two model user groups.

Keywords: interoperability**;** fast charging station; OPC UA; Address Space

1 Introduction

Additional power grid expansion for heavy loaded nodes (with fluctuating power demand) can be avoided by usage of buffering. Fast charging stations with a peak power up to 350 kW [HL17] represent such heavily used grid nodes. The buffer system is represented by a lithium-ion battery, a so-called booster. These batteries consume energy at a medium power until an Electric Vehicle (EV) is plugged in or the upper limiting State of Charge (SoC) is reached. When a loading process of EV starts, power grid and booster

---

[1] Framatome GmbH, Erlangen, Germany/FAU University of Erlangen-Nuremberg, Faculty of Engineering, Martensstraße 5a, Erlangen, 91058, josef.schindler@framatome.com/josef.s.schindler@fau.de

[2] Framatome GmbH, Erlangen, Germany/University of Siegen, Faculty of Science and Engineering, Adolf-Reichwein-Straße 2, Siegen, 57068, venesa.watson@framatome.com/venesa.watson@uni-siegen.de

[3] Framatome GmbH, Erlangen, Germany, karl.waedt@framatome.com

doi:10.18420/inf2019_ws33

deliver jointly a very high power, assuming the booster has enough capacity. The overall process can be found in Fig. 1. After the charging process, the booster starts to refresh again. Considering the total capacity of the battery, it can charge several cars in a sequence before going out of energy.

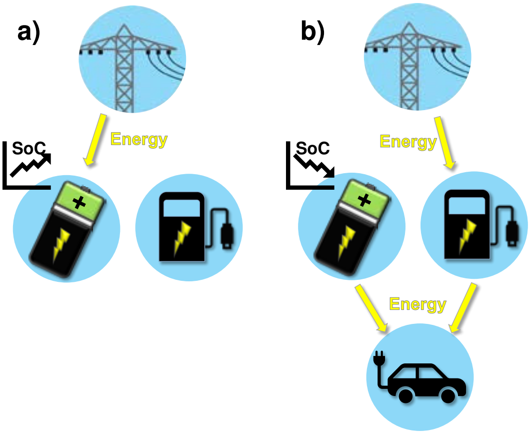

Fig. 1. a) grid charging battery at medium power; b) EV charging process at high power

As batteries behave very sensitive to some crucial states, the internal logic must observe its states e.g. SoC, current, inside temperature. If an error occurs, a (remote) maintenance provider wants to know the reason and therefore needs detailed information about the pre-named variables and access to historical data. Within this context, the State of Health (SoH) of a battery is defined as the ratio between recent maximum capacity and maximum capacity at the beginning of the battery's lifecycle. So, it can be useful to predict service intervals for the maintenance provider or to set manual current limits (to keep thermal caused damage low and the SoH high). Other user groups, like potential customizers, require additional and diverse information about the storage and its states.

Given that charging stations are deployed and managed by different industry partners, interoperability becomes necessary to facilitate exchanges between the charging stations and between the various EVs in operation. This will significantly

- reduce cost and boost convenience – for e.g. EVs will be able to use any charging station regardless of the manufacturer;
- improve service delivery – knowledge-sharing means that a larger data pool for intelligent analysis can be provided; and
- enhance safety monitoring – grid overload can be prevented if information is shared to advice EV users of ideal charging stations based on their voltage requirements and so forth.

ISO/IEC 15118 “Road vehicles -- Vehicle to grid communication interface” is described as the international standard for charging EVs that supports interoperability between charging stations. This allows EVs to be conveniently charged at any available charging point. This standard also defines a security profile for securing this connection [VG02]. Security is a key requirement for protecting the electric grid and ISO/IEC 15118 implements different cryptographic elements in this regard to protect confidentiality, integrity and availability [VG02]. However, ISO/IEC 15118 has a limited scope that considers exchanging messages and initiating a charging session between the charging station and the EV. As highlighted earlier, different user groups could stand to benefit from the interoperable communication features. With ISO/IEC 15118, the EV users benefit from the interoperability for charging conveniently, and electricity distributors benefit from ensuring a stable grid. Additional capabilities can be unlocked by deploying a more advanced interoperability standard, such as IEC 62541 OPC UA; which like ISO/IEC 15118, also provides secure communication between systems. With OPC UA, interoperability is available to both front-end (charging stations and EV connection) and back-end (charging stations and operators) interfaces. The advantages presented by OPC UA are explored through the presentation of use cases for these users.

Before the advantages of OPC UA within this context are shown with standardized Address Space Model & user views (section 3), event notification (section 4) and discovery service (section 5), the hardware setup is presented in section 2. Section 6 gives a short summary and preview on planned work.

2 Description of the Hardware Setup

Each charging station in this example scenario is composed by:

- The charging station itself (e.g. power electronics and the charging interface or plug for EVs);
- An electrical grid connection for power supply;
- The battery used for boosting. In this scenario, we use EVs’ batteries that had already been used in cars. Since their performance decreases during the lifetime, they couldn’t be used any longer in a mobile application;
- A communication interface/gateway, called Batcon. It is directly connected to the battery, simulates the EV’s hardware and thus enables the usage of the battery for immobile application. Furthermore, it has a well-defined and known communication interface unlike the EV’s battery itself; and
- A Server that collects all data from the battery and charging station. This Server supports OPC UA’s Address Space Model and provides other OPC UA functionality and data for external usage.

The plan of such hardware bundle or charging station entity (shorter: entity) is highlighted in Fig. 2 a). Each of these entities is linked via ethernet/internet to the testing computer. The whole scenario containing two entities and the testing computer is then shown in Fig. 2 b). In Fig. 2 a) there are several communication links and a non-specified energy transmission link. The energy exchange occurs at a DC level. Since, data from power electronics is not considered here, they are not shown in this figure (but they are located between the grid and the charging entity).

Data transfer between Batcon and battery doesn't matter here, so it is not specified. For all other communication amongst charging entity components, a Modbus Transmission Control Protocol (TCP) link is implemented. Here, the Programmable Logic Controller (PLC) works as Master/Client. The battery and the charger are the Slaves/Servers, respectively. To an outside communication partner, the PLC works as Server unit.

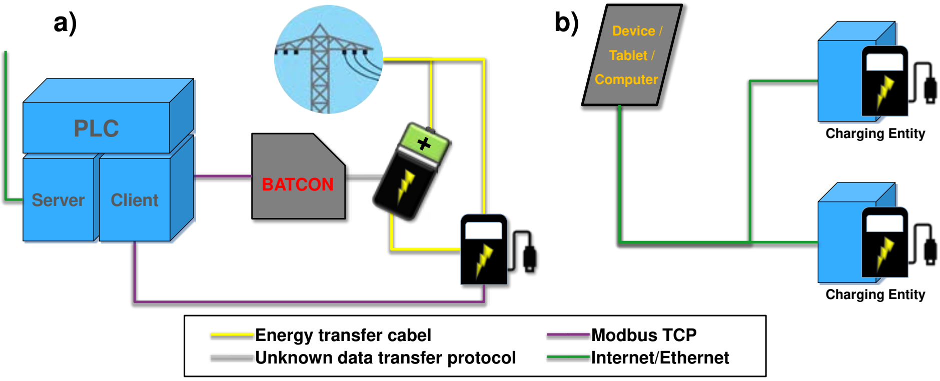

Fig. 2. a) charging entity with all data and energy exchange line (to the power grid); b) setup overview

3 Description of the Data Structure

3.1 OPC UA Address Space Model.

Information sharing in an OPC UA architecture occurs between OPC UA Servers and OPC UA Clients. Data is made available to Clients through Services hosted by the Servers. These Services are operations that determine the actions that the Clients can carry-out on the data, such as Read, Write, Query, and Subscription/MonitoredItem [IC03]. The OPC UA standard defines an Address Space Model that contains the data within the Server and is represented as a set of Nodes (Objects). When a Node is defined, Attributes (data elements that describe Nodes) and References (explicit relationship between Nodes) are instantiated. OPC UA defines a list of eight non-extensible NodeClasses that are described in terms of Attributes and References [IC03][CA14].

Fig. 3 provides an example OPC UA NodeClass representation of an object (here: boiler) using the OPC UA graphical notation. The access control mechanisms are then applied to this Address Space to determine the access rights (data/information) and permissions (actions/operations) of the user. By applying a similar approach to the charging stations a representative perspective of the different users can be obtained. First, it is necessary to define the data that is available. Second, the data that should be visible to each user is determined. This bases on a need-to-know/least-privilege concept, to ensure that each user will have just the right amount of access to complete their tasks. From this, the actions of each user can be determined – for instance, at least 2 or more user groups may have access to the same data, however, one user group would only need read-only access to this data, whilst another user group would need to modify this data. This is discussed in the following subsection for the context of this paper.

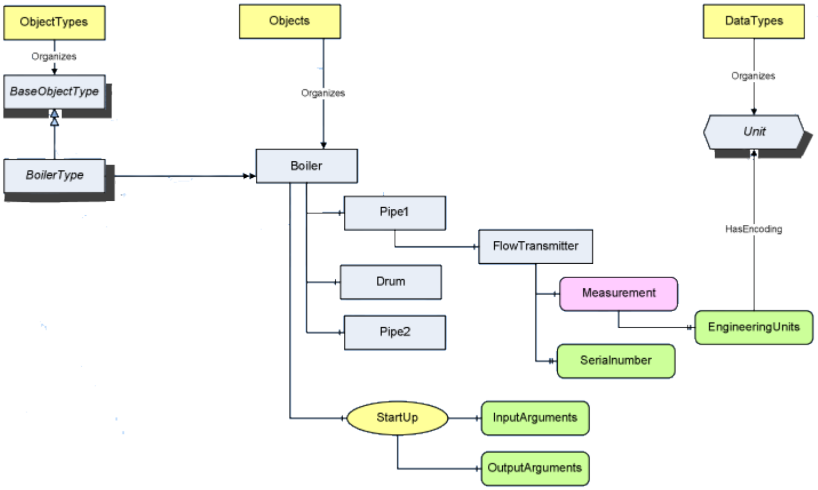

Fig. 3. OPC UA NodeClass Example [OP00]

3.2 Charging Station Data Structure

In XML-code 1, we give an overview of available data in the Server section of the PLC. Since any detail can be relevant for troubleshooting, the default-/overview equals the maintenance view.

Of course, there is some more internal data from the booster, like the voltage of each single battery-cell (batteries consist of hundreds of small cells which are wired up serial for higher external voltage and parallel for greater current). But it is not necessary to be specified here and stored at the Server. Any important details on that (e.g. overvoltage in a single cell, temperature caused errors) can be expressed directly in the error-array.

XML-code 1: Data structure of a boostcharger-entity (default/maintenance view):

```
 <?xml version="1.0" encoding="utf-8" ?>
 <xs:schema
  targetNamespace="http://opcfoundation.org/BinarySchema/"
elementFormDefault="qualified"
xmlns="http://opcfoundation.org/BinarySchema/"
xmlns:xs="http://www.w3.org/2001/XMLSchema"
 >
 <opc:StructuredType Name="BoostchargerEntity">
<opc:Documentation>
     Composed by charging station and battery booster to enable fast charging
</opc:Documentation>


<opc:StructuredType Name="StaticBoostchargerEntityData">
     <opc:Field Name="Provider" TypeName="opc:String" />
     <opc:Field Name="EntityID" TypeName="opc:String" />
     <opc:StructuredType Name="GeoLocation">
        <opc:Field Name="YLatitude" TypeName="opc:Float" />
        <opc:Field Name="XLongitude" TypeName="opc:Float" />
        <opc:Field Name="Floor" TypeName="opc:Byte" SwitchField="InBuilding" />
        <opc:Field Name="InBuilding" TypeName="opc:Bit" />
     </opc:StructuredType>
     <opc:Field Name="EmbeddedInterfaces" TypeName="opc:String" Length="Size" Documentation="Possible plugs for charging">
     <opc:StructuredType Name="ChargingVoltageLevels">
        <opc:Documentation>
           Possible voltage levels for charging processes
        </opc:Documentation>
        <opc:Field Name="LevelsDCArray" TypeName="opc:Int32" Length="SizeDC" />
        <opc:Field Name="LevelsACArray" TypeName="opc:Int32" Length="SizeAC" />
     </opc:StructuredType>
     <opc:Field Name="RatedPower" TypeName="opc:Float" Documentation="Consists of rated battery and grid power, respectively" />
     <opc:StructuredType Name="BoosterSpecification">
        <opc:Field Name="RatedPower" TypeName="opc:Float" />
        <opc:Field Name="RatedCapacity" TypeName="opc:Float" />
     </opc:StructuredType>
```

```
</opc:StructuredType>

<opc:StructuredType Name="DynamicBoostchargerEntityData">
     <opc:StructuredType Name="BoosterStates">
        <opc:StructuredType Name="AvailabilityBooster">
           <opc:Field Name="BlnAvailable" TypeName="opc:Bit" />
           <opc:Field Name="DateAvailable" TypeName="opc:DateTime" SwitchField="BlnAvailable" SwitchValue="False" Operand="Equal" />
        </opc:StructuredType>
        <opc:StructuredType Name="ErrorBattery">
           <opc:Field Name="BlnError" TypeName="opc:Bit" />
           <opc:Field Name="ErrorIDs" TypeName="opc:UInt16" SwitchField="BlnError" Terminator="FFFF" />
        </opc:StructuredType>
        <opc:Field Name="CurrentPower" TypeName="opc:Float" />
        <opc:Field Name="CurrentCapacity" TypeName="opc:Float" />
        <opc:Field Name="StateOfCharge" TypeName="opc:Float" />
        <opc:Field Name="StateOfHealth" TypeName="opc:Float" />
     </opc:StructuredType>
     <opc:StructuredType Name="ChargerStates">
        <opc:StructuredType Name="AvailabilityCharger">
           <opc:Field Name="BlnAvailable" TypeName="opc:Bit" />
           <opc:Field Name="DateAvailable" TypeName="opc:DateTime" SwitchField="BlnAvailable" SwitchValue="False" Operand="Equal" />
        </opc:StructuredType>
        <opc:StructuredType Name="ErrorCharger">
           <opc:Field Name="BlnError" TypeName="opc:Bit" />
           <opc:Field Name="ErrorIDs" TypeName="opc:UInt16" SwitchField="BlnError" Terminator="FFFF" />
        </opc:StructuredType>
        <opc:Field Name="CurrentPower" TypeName="opc:Float" />
        <opc:StructuredType Name="ReservationCharger">
           <opc:Field Name="BlnReservationActive" TypeName="opc:Bit" />
           <opc:Field Name="DateReservationEnd" TypeName="opc:DateTime" SwitchField="BlnReservationActive" />
        </opc:StructuredType>
     </opc:StructuredType>
</opc:StructuredType>
```

```
<opc:StructuredType Name="HistoricalBoostchargerEntityData" >
    <opc:Field Name="Size" TypeName="opc:UInt32" />
    <opc:Field Name="ChargedVehicleID" TypeName="opc:String"
    Length="Size"/>
    <opc:Field Name="StartTime" TypeName="opc:DateTime"
    Length="Size"/>
    <opc:Field Name="EndTime" TypeName="opc:DateTime" Length="Size"/>
    <opc:Field Name="EnergyTransfered" TypeName="opc:Double"
    Length="Size"/>
    <opc:Field Name="MaximumPower" TypeName="opc:Double"
    Length="Size"/>
    <opc:Field Name="Successful" TypeName="opc:Bit" Length="Size"/>
</opc:StructuredType>
 </opc:StructuredType>
```

From that overview a smaller view for end-users can be derived. To illustrate the shrinking process an example is used: When a customizer wants to charge his EV at the station, some conditions must be fulfilled:

1. The hardware interfaces or plugs need to be compatible. The most common system in Europe is the “Type 2”-connector and the high power using Combined Charging System (CCS). There are other often used systems like one from Tesla or the CHAdeMO [HL17].
2. The station needs to be available, meaning that only one EV can be charged at the same time
3. There mustn’t be any fatal error at the station (e.g. some wires are broken)

There are also some constraints that do not allow the fast charging mode for the booster:

1. The SoC or capacity needs to be sufficient for the charging cycle and the EV’s energy demand
2. There must not be any fatal error at the battery (e.g. the battery’s internal temperature is too low or too high)

Sure, each of these details could be provided to the end-user. But the customer, who wants to charge his EV, doesn’t need the exact error description or the reason, why the station is not available. He only needs a status information available or not (compare the upper conditions 1.-3.). And if he wants to use the fast charging boost, he needs to know whether that mode is available (compare the lower constraints 1. & 2.).

Hiring such consideration for each variable in XML-code 1, a short view for a customizer that searches for an available charging station via application or internet can be derived. The user’s device can then decide whether to list this charging station (if the station is available and EV’s specifications are matching the charging station one’s) or not.

For a better understanding, Fig. 4 shows the graphical notation of the PotentialEndUserView as shrinking result from XML-code 1.

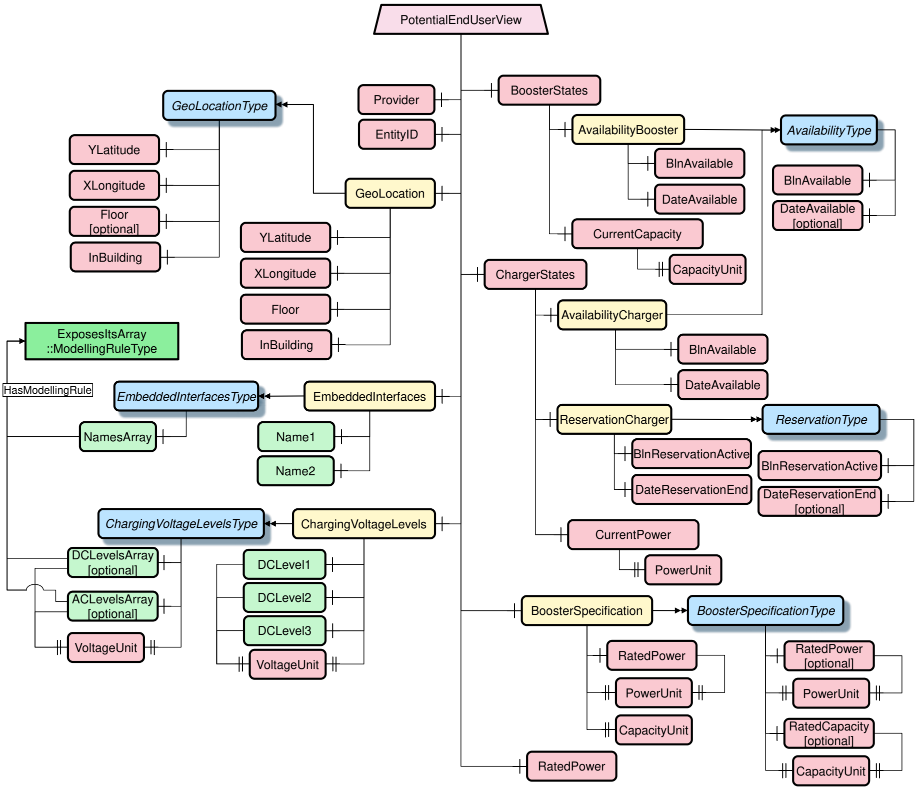

Fig. 4. Graphical notation of the end-user view

All the important types are defined inside this structure (blue), their instances are colored yellow. Since it would exceed the space, the view does not contain:

- Type definitions of simple variables (that inherit their structure from standard types as “BaseObjectType”, “PropertyType”, “BaseDataVariableType…)
- Possible notification structures/references
- Superior data structure information – it is already given in XML-code 1
- “Organizer”, such as ObjectTypes and Objects

4 Usage of subscription/events

Besides the Address Space Model, OPC UA provides several services that are useful in the scenario's context. This section shows how Events and Subscriptions can help for a better user experience and faster fault handling in the pre-defined scenario.

"Events represent specific transient occurrences. System configuration changes and system errors are examples of Events. Event Notifications report the occurrence of an Event. Events […] are not directly visible in the OPC UA Address Space. Objects and Views can be used to subscribe to Events. The EventNotifier Attribute of those Nodes identifies if the Node allows subscribing to Events. Clients subscribe to such Nodes to receive Notifications of Event occurrences." [IC03]

An example for such Event can be found easily for a maintenance staff-Client. The staff needs to know the appearance of errors in a charging entity immediately. Therefore, the Client needs to subscribe to the Event "ErrorAppearance", which is illustrated in Fig. 5 and is active, if an "Error" is "Active". The notification service can save effort, work and thus money, because the maintenance staff does not need to check the error storage of every charging entity manually.

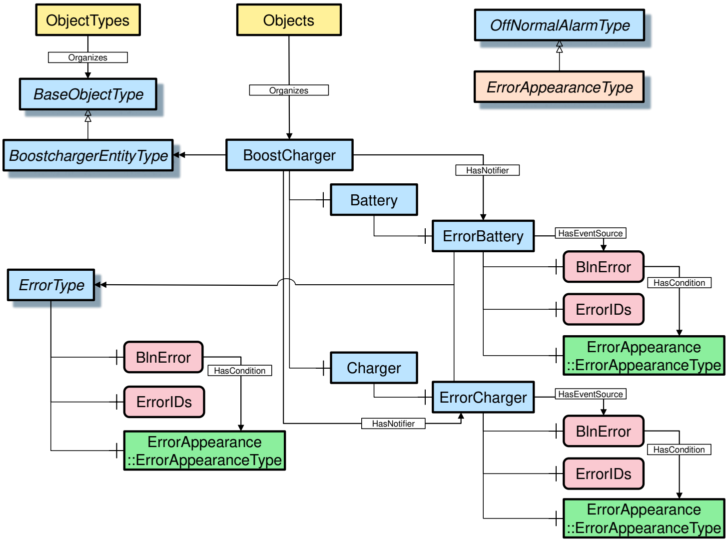

Fig. 5. Example of error detection with event notification (only necessary nodes shown)

For potential user, there are examples for usage of event notifiers too. The owner of an EV wants to drive from city A to B. He needs one stop for charging, because the distance between the cities exceeds the range of the car. The EV owner wants to save time and therefore the charging station must provide fast charging and be reachable as possible. His navigation system calculates an optimized route, but since no fast charging station is available at the moment, the route is planned with a normal charging station. At the same time the navigation system can set up Event Notifiers at all charging stations on the route. These shall tell, when they get available. If so, the EV can derive, whether it is still a possible alternative to the planned charging station. If so, the EV can set a reservation.

5 Functionality of Discovery Service

Essentially, the Address Space is how Servers represent their functionality to Clients in a standard way. The more sophisticated the Address Space, the better the device is at representing its data and functionality [Rj18]. However, in order to make use of these functionalities and data, an OPC UA Client must first be aware of OPC UA Servers available to it. A Client needs to somehow obtain the address of a Server in order to connect to it. This is facilitated through the OPC UA Discovery, which is a part of the OPC UA defined architecture. The OPC UA Discovery not only provides Clients with Server addresses, but supports protocols, security policies and other capabilities [OP01].

Once, a Server becomes visible to a Client, the pre-configured access control mechanisms are activated to distribute the Client's privileges. In the context of a charging station as an OPC UA Server, Clients at the end-user (remote or directly plugged-in) and the maintenance staff access this charging station and are provided with different views (compare chapter 3.2).

For example, upon connection to the charging station, the user is presented with information pertaining to battery status information (e.g. "CurrentPower") and the Client can then derive other values from it (e.g. "percentage charge completed", "estimated time until completely charged"). On the end of the maintenance staff, low-level communication (not accessible or of interest to the end user) is communicated to the Server (charging station) for their action (See Fig. 6).

For instance, the maintenance staff could subscribe to the charging station for events/alerts battery failures. They may then be equipped with the necessary privileges to stop availability of a charging station for maintenance or repair work.

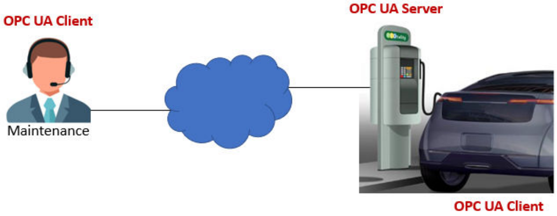

Fig. 6. Example of communication configuration for Electric Vehicle (EV) charging station with OPC UA concepts

6 Conclusion

This paper gives an example for the usage of OPC UA standard by representing data for different users and explaining some OPC UA Services. With this first work, it is maybe possible to determine a standardized data structure for fast charging entities with battery boosters. Please note that this paper does not claim to set a new standard. It should only provide a first idea or serve as a template for extension of existing standards (ISO 15118…).

Anyway, further work is necessary and will be published soon. Some unsolved issues are that e.g. payment hasn't been addressed yet. To underline the benefits of OPC UA, further services provided can be demonstrated. Here, authentication needs to be mentioned, since it is crucial for implementation several user groups. Authentication and other security relevant topics can then be compared to IEC 62351, the existing standard for security and concomitant data transfer.

7 Acknowledgements

Some of the addressed topics are being elaborated as part of Framatome GmbH's participation in the DECENT R&D (2018-2020) with Technical University of Munich (Germany), FORTISS (Germany), IBDM (Germany), FENECON (Germany), VTT Technical Research Centre of Finland Ltd (Finland), Empower IM Oy (Finland), Wirepas (Finland) and Fourdeg (Finland).

References

[CA14] CAS: OPC UA Information Model Deployment, http://www.cas.internetdsl.pl/commserver/P_DowloadCenter/P_Publications/20140301EN_DeploymentInformationModel.pdf, accessed: 26/04/2019

[HL17] Dale Hall & Nic Lutsey: Emerging best practices for electric vehicle charging infrastructure, https://www.theicct.org/sites/default/files/publications/EV-charging-best-practices_ICCT-white-paper_04102017_vF.pdf, accessed: 20/05/2019

[IC03] IEC: IEC 62541-3: 2015 OPC unified architecture – Part 3: Address Space Model

[OP00] OPC Foundation: OPC UA Information Modeling, https://www.automaatioseura.fi/site/assets/files/1442/opc_6_hunkar_informationmodel-4.pdf, accessed: 26/04/2019

[OP01] OPC Foundation: UA Companion Specifications, https://opcfoundation.org/about/opc-technologies/opc-ua/ua-companion-specifications/, accessed: 26/04/2019

[Rj18] Rinaldi, J.: What's Really In Your OPC UA Server - OPC UA Servers are much more sophisticated than Servers of other technologies, http://www.automatedbuildings.com/news/jan16/articles/realtime/151218114909realtime.html, accessed: 26/04/2019

[VG02] V2G: The Basics of Plug & Charge, https://v2g-clarity.com/knowledgebase/basics-of-plug-and-charge/, accessed: 26/04/2019