PNNL- 34256

# Universal Utility Data Exchange (UUDEX):

Information Exchange Models for Solar DER

May 2023

S.R. Mix

Prepared for the U.S. Department of Energy
under Contract DE-AC05-76RL01830

PNNL- 34256

# Universal Utility Data Exchange (UUDEX):

Information Exchange Models for Solar DER

May 2023

S.R. Mix

Prepared for
the U.S. Department of Energy
under Contract DE-AC05-76RL01830

Pacific Northwest National Laboratory
Richland, Washington 99354

## Acknowledgement

The author acknowledges the support of the DOE Office of Energy Efficiency and Renewable Energy (EERE), Solar Energy Technologies Office (SETO), Securing Solar for the Grid (S2G) program for the development of this report.

## Summary

This report describes how the Universal Utility Data Exchange (UUDEX) can encapsulate and exchange information about distributed energy resources (DER) within UUDEX-formatted JSON documents. It proposes to use the existing extensible markup language (XML) information exchange format described in *IEEE Std 2030.5, IEEE Standard for Smart Energy Profile Application Protocol* but not other information exchange provisions of IEEE 2030.5, rather, relying on equivalent mechanisms in UUDEX. This will allow adoption of the UUDEX exchanges with minimal changes to the structure of the information exchanged and will not require a modification to the UUDEX documentation if IEEE 2030.5 makes changes to the information models.

# Acronyms and Abbreviations

| | |
|---|---|
| API | application programming interface |
| BA | Balancing Authority |
| CESER | Cybersecurity, Energy Security, and Emergency Response, an office of the U.S. Department of Energy |
| CIM | common information model |
| DER | distributed energy resource(s) |
| DERMS | distribution energy resource management system |
| DMS | distribution management system |
| DNS | Domain Name System |
| DOE | U.S. Department of Energy |
| ECMA | European Computer Manufacturers Association |
| EMS | energy management system |
| EVSE | electric vehicle supply equipment – an electric vehicle charging station |
| GMS | generation management system |
| GOP | Generator Operator |
| IEC | International Technical Commission |
| IEEE | Institute of Electrical and Electronics Engineers |
| JSON | JavaScript Object Notation |
| NERC | North American Electric Reliability Corporation |
| PKI | public key infrastructure |
| PSAP | Public Service Answer Point |
| RC | Reliability Coordinator |
| RFC | (Internet) Request for Comment |
| SCADA | supervisory control and data acquisition |
| SEP2 | Smart Energy Profile Version 2 |
| SEP 2.0 | alternative for Smart Energy Profile Version 2 |
| TOP | Transmission Operator |
| URI | uniform resource identifier |
| UUDEX | Universal Utility Data Exchange |
| W3C | World Wide Web Consortium |
| XML | Extensible Markup Language |

## Contents

## Figures

## Tables

# 1.0 Introduction

This section of the report contains a brief introduction to the three major concepts discussed later in the report, namely formats for expressing structured information (XML and JSON) and how UUDEX works. It also provides an introduction to the IEEE 2030.5 standard and the web-based interface it uses to communicate with DER devices.

## 1.1 XML and JSON

This report references both Extensible Markup Language (XML) and JavaScript Object Notation (JSON) as formats for representing and transmitting data.

The World Wide Web Consortium (W3C) specifies XML in [B6].

Internet Request for Comment (RFC) 8259 (and European Computer Manufacturers Association [ECMA] standard ECMA-404) specifies JSON in [B3].

Both XML and JSON refer to collections of data expressed as sets of data as “documents”. Each XML or JSON document has a structure that defines the name of the document, and the start and end of the document.

XML organizes data in “elements”, with a start tag (the element name) and end tag (the element name preceded by a slash [/] character) surrounding the contents of the element. XML does not directly support arrays in elements, but the equivalent of an array of elements is expressed by repeating the element in the document. For example, a minimal XML document containing the element “document” that does not have any data (i.e., a “null” document) would be as shown in Figure 1.1:

```
<document></document>
```

Figure 1.1: Example Blank XML Document

JSON organizes data using name-value pairs, organized as objects surrounded by curly braces (“{“ and “}”), with the name of the object as a quoted string, followed by a colon (“:”) and the value of the object, generally expressed as a string (but numbers, and the terms *true* and *false* are also allowed). The value may also be an array of values or other objects as a comma-separated list enclosed by square brackets (“[“ and “]”). The same null document shown above in JSON would be as shown in Figure 1.2:

```
{"document":""}
```

Figure 1.2: Example Blank JSON Document

Both XML and JSON allow for inclusion of an arbitrary number of nested elements or objects (or “sub-documents”) within a single document, and computer files containing XML or JSON can include one or more documents. Each has a separate syntax defined in their specifications.

A complete introduction to XML and JSON is beyond the scope of this report – see the specifications in [B6] and [B3] for more information.

Schema definitions for both XML and JSON describe the format of specific documents, e.g., whether specific fields are numeric or alphanumeric, and which fields are mandatory. A discussion of this is also beyond the scope of this report.

## 1.2 UUDEX

The Universal Utility Data Exchange (UUDEX) (see Universal Utility Data Exchange (UUDEX) Functional Design Requirements - Rev 1 [B16], Universal Utility Data Exchange (UUDEX) - Workflow Design - Rev 1 [B21], Universal Utility Data Exchange (UUDEX) – Protocol Design - Rev 1 [B20], Universal Utility Data Exchange (UUDEX) Information Structures - Rev. 1 [B17], and Universal Utility Data Exchange (UUDEX) – Security and Administration - Rev.1 [B18]) is a U.S. Department of Energy (DOE) Office of Cybersecurity, Energy Security, and Emergency Response (CESER) funded project to provide a self-describing, model-based, information exchange mechanism, primarily for exchange of information between electric utility control centers such as those of transmission operators (TOP), generator operators (GOP), balancing authorities (BA), and reliability coordinators (RC)[1]. UUDEX can also submit reports (e.g., DOE-417 disturbance reports), exchange large files (e.g., power system model files), and to provide outage information from an outage management system at an electric utility to the public safety answer point (PSAP) system at a 9-1-1 call center to provide first responder organizations with the energization status of electric lines and other equipment.

The Institute of Electrical and Electronics Engineers (IEEE) is standardizing UUDEX as *IEEE P2030.103 Draft Standard for Universal Utility Data Exchange (UUDEX)* [B12]. The UUDEX technical reports listed previously are the source material for that standardization effort. The standardization effort is updating this material based on broader industry input. The IEEE standardization effort will use concepts from this report to describe DER-specific communications.

UUDEX uses a *publish-subscribe with persistence* mechanism for information exchange. UUDEX Publishers send information to an intermediary device (referred to as either the UUDEX Server or the UUDEX Instance); the UUDEX Instance stores a copy of the message until all UUDEX Subscribers have retrieved a copy of the message. (UUDEX Publishers and UUDEX Subscribers are sometimes referred to as UUDEX Clients.) This allows asynchronous (where the UUDEX Subscriber may delay retrieving messages until it is convenient) and near synchronous (where the UUDEX Subscriber is either constantly polling for a UUDEX Publisher to publish new messages or the UUDEX Subscriber issues a request and waits for a UUDEX Publisher to publish a message) exchange of information. The UUDEX Instance uses an underlying message bus to manage the subscription processing, and has been successfully demonstrated in a laboratory environment using both RabbitMQ [B22] and Apache Kafka [B1] interoperating with the same UUDEX Client application programming interface (API). The modular nature of the UUDEX design allows replacement of underlying message bus implementations as they become available.

The individual data exchange interactions are one-way (UUDEX Publisher to UUDEX Subscriber). Each interaction requires that a subscription “subject” (a term used by many message bus implementations for describing the information exchanges) – a UUDEX Subject in this context – be set up to convey the information between the UUDEX Publisher and the UUDEX Subscriber. There can be multiple UUDEX Publishers to a UUDEX Subject allowing

[1] The North American Electric Reliability Corporation (NERC) Functional Model [B19] describes the names, abbreviations, and roles for these electric sector participants.

aggregation of information by the UUDEX Subscriber, and multiple UUDEX Subscribers to a UUDEX Subject allowing broader distribution of the same information with minimal impact to the UUDEX Publisher as new UUDEX Subscribers subscribe to the same information. This means that UUDEX can support any combination of one-to-one, one-to-many, many-to-one, and many-to-many interactions. In the case of a distributed energy resource (DER) interacting with a higher-level control system, a combination of many-to-one (for telemetry) and one-to-one (for issuing command and control instructions) interactions are the most likely, although other interactions are possible, such as using a one-to-many interaction with a common UUDEX Subject for publishing frequency setpoints to all DERs from the BA control system.

If bi-directional interactions are required, for example in the case of a request/reply interaction, two UUDEX Subjects would be required: one for the UUDEX Requestor (requestor publishes and responder subscribes) and one for the UUDEX Responder (responder publishes and requestor subscribes), with a reference identifier specified to correlate a specific request with its specific reply. Note that since both the UUDEX Requestor and UUDEX Responder will be publishing and subscribing to the UUDEX messages, the terms “UUDEX Publisher” and “UUDEX Subscriber” are generally not used in the context of the request/reply capability.

Figure 1.3 (extracted from Section 5 of [B17]) shows an example relationship between publishers, subjects, subscriptions, and subscribers in a hypothetical UUDEX implementation. Section 5 of [B17] provides additional details, including a detailed explanation of the process flow in the figure.

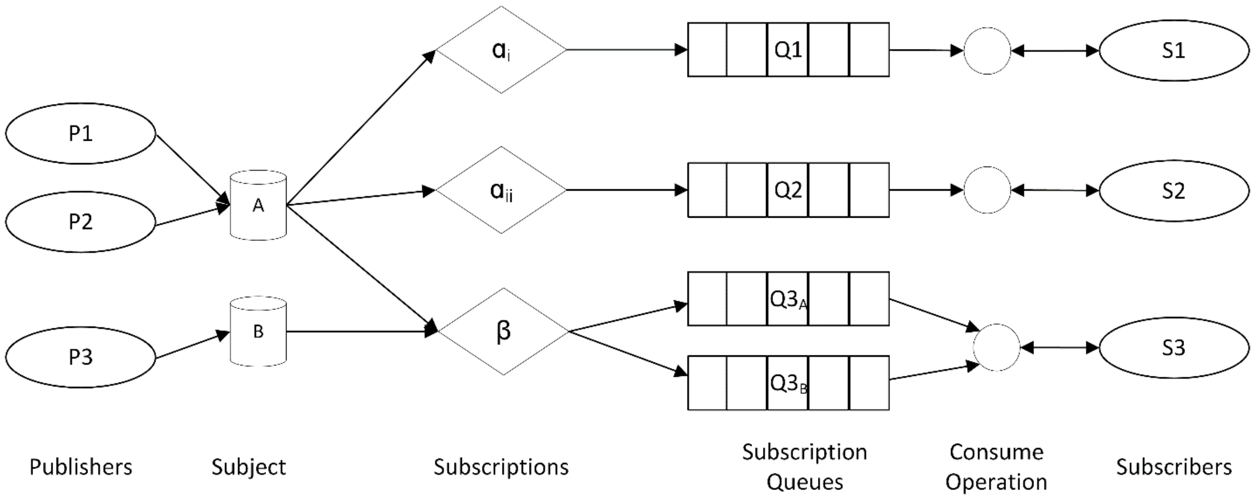

Figure 1.3: Subscription and Queue Processing

The one-way store-and-forward mechanism UUDEX uses does not allow for negotiation of parameters, so all information exchange parameters must be pre-negotiated. This approach is unlike many other client-server mechanisms such as that used by *IEEE Std 2030.5-2018 IEEE Standard for Smart Energy Profile Application Protocol* [B11]. The lack of negotiation requires that certain aspects of the information exchange (e.g., the name of the UUDEX Subject, compression and encoding algorithms) be agreed by both the UUDEX Publisher and UUDEX Subscriber *a priori*, but in most cases other than the name of the UUDEX Subject, the UUDEX standard specifies a minimum required set of parameters, and may specify a default that can be used without any negotiation. However, for information exchanges, the lack of negotiation is not a detriment for well-known or standardized exchanges since the exchanged information is mostly self-describing, and UUDEX Subscribers may ignore any exchanged data elements that they cannot understand or interpret. UUDEX also supports custom information exchanges (e.g., interfacing to a market information system), but they require added agreement between the UUDEX Publisher and UUDEX Subscriber for exchanged data and formats, since UUDEX does not specify data format details for non-standard interactions.

UUDEX provides information security by using standard X.509 digital certificates [B13] with specific meaning applied to certain fields of the digital certificate. The digital certificates are part of a public-key infrastructure (PKI) system that uses a closed certificate authority hierarchy (likely managed by the same organization that manages the UUDEX Instance) to manage which nodes or organizations can participate in the information exchanges with the UUDEX Instance. A given UUDEX Client may participate in multiple UUDEX Instances by using a different digital certificate for each UUDEX Instance, for example acting as a gateway to subscribe to DER information from a GOP using the GOP's UUDEX Instance and publish it to a BA using the BA's UUDEX Instance. The closed certificate hierarchy allows UUDEX nodes to authenticate and communication with each other without the need for external access or communication, for example to an Internet-based certificate authority. UUDEX uses digital certificates to establish authenticated and encrypted network sessions between UUDEX Publishers or UUDEX Subscribers and the UUDEX Instance. Allowable actions for a given UUDEX Publisher or UUDEX Subscriber and a given interaction request are based on the identity in the digital certificate. These interactions occur through UUDEX Subjects. The UUDEX Subject owner controls which certificate identities (either by individual name, or membership in a group or role) have access: which can publish to the UUDEX Subject, and which can subscribe to the UUDEX Subject. The owner (and therefore default access manager) of a UUDEX Subject may be any UUDEX Participant but is typically the UUDEX Publisher for UUDEX Subjects that primarily supply information, or the UUDEX Subscriber for UUDEX Subjects that primarily gather information.

UUDEX was initially intended for use in control center-centric communications such as those between a centralized generation management system at a GOP's control center and the control center systems at a BA, but the flexibility of the information exchange allows it to be used in other applications (much like the predecessor Inter-control Center Communications Protocol [ICCP] was initially designed for communication between control centers, but was extended for communications directly to generation plants [B9]). The primary expected use of the DER information structures described in this report is for relaying information from an individual DER resource or DER plant through a GOP's generation control system to the control system at a BA, TOP, or RC. The GOP could communicate with the individual DER using native IEEE 2030.5 and forward the data content of the IEEE 2030.5 response directly to the BA, TOP, or RC using a UUDEX formatted message containing the IEEE 2030.5 formatted DER information. This would allow the control systems at the BA, TOP, or RC to receive DER information without needing a complete implementation of IEEE 2030.5 or requiring that it establish communication directly to the DER allowing the GOP to receive and transmit the DER information without significant reformatting.

## 1.3 IEEE 2030.5

IEEE 2030.5 is the IEEE standard for exchanging information about DER devices and installations with DER control systems. It is based on earlier work by the ZigBee Alliance and HomePlug Powerline Alliance published in 2021 as the Smart Energy Profile 2.0 (SEP2 or SEP 2.0) Application Protocol Specification [B25]. It uses a RESTful interface [B7] built on existing internet protocols such as HTTP.

The RESTful interface used by IEEE 2030.5 relies on uniform resource identifiers (URI) to describe the desired information source (or destination). Figure 1.4 shows an example SEP 2.0 specification URI for requesting DER status information:

```
/sep2/edev/1/der/2/ders
```

Figure 1.4: Example URI to request DER Status Information – SEP 2.0 format

Table 1-1 describes the breakdown of the fields in Figure 1.4.

Table 1-1: URI Fields used in the Example

| URI Field | Meaning |
|---|---|
| `sep2` | Identifies the URI as a Smart Energy Profile 2.0 URI |
| `edev` | End device list |
| `1` | End device instance number |
| `der` | Distributed Energy Resource (DER) |
| `2` | DER instance number |
| `ders` | DER Status Information request |

Note – In the SEP 2.0 specification, URIs start with “/sep2”, but for IEEE 2030.5, URIs, the “/sep2” prefix has not been caried forward. Figure 1.5 shows the equivalent IEEE 2030.5 URI for Figure 1.4:

```
/edev/1/der/2/ders
```

Figure 1.5: Example URI to request DER Status Information – IEEE 2030.5 format

The remainder of this report uses the SEP 2.0 format for URIs, since the sources for the examples used in this report use them.

IEEE 2030.5 clients invoke the RESTful interface using URIs passed by `HTTP`[2] `PUT`, `GET`, `POST`, or `DELETE` commands as shown in Figure 1.6:

```
HTTP PUT /sep2/edev/1/der/2/ders
```

Figure 1.6: Example REST request for DER Status Information

In Figure 1.6, an IEEE 2030.5 client makes a request to an End Device (`edev`) with an instance number of one (1), and to the specific DER at that instance with a DER instance of two (2). The URI structure supports a given node (as identified by an IP address that receives the RESTful interface request) that can support one or more End Devices, with each End Device supporting one or more individual DERs.

More detail on the usage of the URIs and other values for the various fields and formats of the URIs is available in Annex A of the IEEE 2030.5-2018 standard.

[2] Note – the IEEE 2030.5 standard primarily references HTTP (the unsecured version of Hypertext Transfer Protocol), although does make mention of HTTPS (the secure version of HTTP). This report will reference HTTP, but notes that in most cases, HTTP and HTTPS are interchangeable when using RESTful calls.

# 2.0 Approach

The initial concept for this task was to develop native UUDEX information structures (in JSON) to communicate essential information about DERs. After reviewing existing DER communication protocols and mechanisms, the task investigated using existing information structures rather than developing custom models for UUDEX. As a result of the investigation, the task deliverable changed to using existing IEEE 2030.5 information structures.

Note that this report describes only an approach for information sharing and does not replace all the functionality of IEEE 2030.5 or any other standard. IEEE 2030.5 will still be required for other functions such as provisioning and configuration, and information exchanges using IEEE 2030.5 may still be appropriate when communicating directly with DER devices for gathering telemetry or issuing controls.

## 2.1 Alternatives

To understand the requirements for DER information exchange, existing protocols and standards were reviewed as part of this task. These alternatives included SunSpec Modbus [B22], IEEE 2030.5, and *IEEE Std 1858, IEEE Standard for Electric Power Systems Communications-Distributed Network Protocol (DNP3)* [B10]. After this review, it was determined that existing models were feature rich and complex, which would require a similarly feature rich and complex UUDEX information structure. Therefore, rather than generate a new JSON information exchange structure for UUDEX to communicate DER measurements and other values of interest, UUDEX will use information exchange structures using the XML format from the existing IEEE 2030.5. Note that this approach only uses the information exchange structures from IEEE 2030.5, not the entire protocol that also specifies the transport mechanism and cybersecurity controls (authentication, transport security, etc.), since UUDEX already includes equivalent mechanisms for these functions.

IEEE 2030.5 was selected over alternatives since IEEE 2030.5 uses an inherently self-describing XML-based information model. This model is consistent with the JSON models already used by UUDEX and can use the existing UUDEX support for the direct encapsulation of XML. This approach also reduces the need to update UUDEX-specific models that may over time become inconsistent with future revisions of IEEE 2030.5. By specifying that the information models are the same as those specified by IEEE 2030.5 (current version), as the IEEE standards review process updates IEEE 2030.5, the UUDEX models will automatically update without any added UUDEX standard development. If information models for a specific version of IEEE 2030.5 are used (for example, the earlier version IEEE 2030.5-2013), if necessary or appropriate, the UUDEX Client can specify a specific version of the IEEE 2030.5 standard in the appropriate UUDEX noun field (see Section 2.5).

An additional alternative approach that was considered and rejected was to use DER information models under development by the International Technical Commission (IEC) Technical Committee 57 (TC57) Working Group 14 (WG14) for inclusion in the IEC's Common Information Model (CIM) energy management system (in either standard IEC 61968 [B2], IEC 61970 [B4], or IEC 62325 [B7]). However, since the specifics of those models are still undergoing development, they would not be proper for use at this time.

## 2.2 Implementation

As noted in Section 1.2, UUDEX is an asynchronous information exchange mechanism and does not allow for negotiation to determine what values are available for use; rather, it requires that the UUDEX Publisher and UUDEX Subscriber agree *a priori* about which elements the UUDEX Publisher publishes and expects that the published elements are sufficiently self-describing so that the UUDEX Subscriber can interpret the published values. This self-describing nature of the XML also allows UUDEX to transmit individual elements in an arbitrary order (e.g., the order in which the information is gathered by the UUDEX Publisher from the individual DER devices), and to transmit subsets of elements (e.g., only those that have significantly changed since the last transmission).

The lack of negotiation also results in the inability to negotiate protocol parameters such as compression or encoding algorithms. The UUDEX specification provides several standard approaches for these (and other) parameters, and in some cases default values for the parameters. If standard parameters are not appropriate for a particular exchange, information exchange participants can negotiate alternatives and agree to them out-of-band. Once agreed to, the UUDEX participants can create custom message structures containing the additional parameters.

At the UUDEX Instance managed by the central control center, several UUDEX Subjects will need to be set up to exchange DER information. For example:

- One UUDEX Subject for all DER devices to report their real-time measurements and other information to the central control system (i.e., an energy management system [EMS], generation management system [GMS], distribution management system [DMS], distributed energy resource management system [DERMS], or supervisory control and data acquisition system [SCADA]). Each DER device will publish its information to this UUDEX Subject, and the central control system will subscribe from it to receive the information. Since each message will include a reference to the specific DER device and there is no need to protect the information about individual DERs from the central control system, the UUDEX instance will not need separate UUDEX Subjects for each DER.
- One UUDEX Subject for each DER device to subscribe to and receive control or other information from the central control system (e.g., GMS or DERMS). The central control system will publish information specific to the DER device (or collection of DER devices that are collocated with each other such as multiple inverters at a large solar farm), and each DER device will subscribe to the UUDEX Subject corresponding to themselves. Although the information contained in the message identifies the specific DER device, separate UUDEX Subjects for each DER minimizes spurious download traffic for messages that the DER would end up ignoring when processed on the DER.

  Since each issued command is specific to a particular DER, each DER needs a separate UUDEX Subject (even if the UUDEX Publisher sends the same logical command to multiple DERs). Cybersecurity provisions built into UUDEX decrease the possibility of a DER subscribing to the wrong UUDEX Subject. This also decreases the possibility for market sensitive information leakage about control actions for a group of unrelated or unaffiliated DERs.
- Potentially, one UUDEX Subject that the central control system can publish “broadcast” messages to, and to which all DER devices will subscribe. This allows the central control system to effectively communicate to all devices simultaneously, allowing the UUDEX message bus software to make a copy of the message is available to each subscribed DER

device. An example of this could be if the central control system at the BA needs to change the frequency setpoint schedule for all generators (DER and conventional) in the system.

In this scenario, the DER devices and the central control system all publish and subscribe to multiple UUDEX Subjects. The UUDEX Subject name as well as the contents of the published message supply information about processing individual messages. Different program threads in the UUDEX Client processing could handle publication and subscription for a specific set of UUDEX Subjects, or a single thread could handle publications for all UUDEX Subjects while another thread could handle subscriptions for all UUDEX Subjects. The UUDEX architecture and interface specification allows for multiple UUDEX Publisher or UUDEX Subscriber implementation variations, even within the same UUDEX Instance.

There is no practical limit to the number of individual UUDEX Subjects or subscriptions that a UUDEX Instance can support, allowing exchange of IEEE 2030.5 formatted data in this scenario alongside other UUDEX information exchanges supported by the UUDEX Instance.

## 2.3 Format

There are two approaches for inserting IEEE 2030.5 content in a UUDEX message. The first is to translate the IEEE 2030.5 XML data to JSON, resulting in native JSON formatted data. This approach does not require any added XML processing for the UUDEX Subscriber to interpret the data but requires re-translating the data back into XML if an IEEE 2030.5 client needs the data. The second is to keep the IEEE 2030.5 XML formatting and encapsulate the XML data as a stream of characters in a JSON string. This approach requires the UUDEX Subscriber to process the XML if the UUDEX Subscriber needs access to the data, but requires no additional processing before sending the data to an IEEE 2030.5 client.

The XML and JSON examples shown in this report are shown using both “pretty” (blank space and indenting for human readability) and “compact” (blank space removed for efficient communication) format. In several examples (especially when using compact format) the text is longer than a single line; in these cases, the lines are wrapped at the end of the printed line without regard to where the normal word breaks would normally occur in a text paragraph.

### 2.3.1 XML Examples

To supply realism to the examples used, this report extracted examples of IEEE 2030.5 XML data from published papers describing DER interactions using IEEE 2030.5. These include Johnson *et.al.* [B14], and online documentation for VOLTTRON [B24].

Figure 2.1 and Figure 2.2 (extracted from [B14]) show XML documents that includes status and capability reporting, while Figure 2.3 (also from [B14]) shows an XML document for setting parameters. These examples show the commands used to request the messages (`HTTP PUT` commands) followed by the IEEE 2030.5 XML response documents. They also refer to the DER device identified by the IEEE 2030.5 URI nomenclature “`/sep2/edev/1/der/1`” or “`/sep2/edev/96/ps`”.

```
HTTP PUT /sep2/edev/1/der/1/ders

<DERStatus xmlns="urn:ieee:std:2030.5:ns">
   <genConnectStatus>
      <dateTime>l 4563 45000</dateTime>
      <value>O</value>
   </genConnectStatus>
   <readingTime>l4 56345000</readingTime>
</DERStatus>
```

Figure 2.1: Example XML DER Status Reporting

```
HTTP PUT /sep2/edev/1/der/1/dercap

<DERCapability xmlns="urn:ieee:std:2030.5:ns">
   <modesSupported>3FFFFF</modesSupported>
   <rtgA>
      <multiplier>O</multiplier>
      <value>20</value>
   </rtgA>
   <rtgW>
      <multiplier>O</multiplier>
      <value>SOOO</value>
   </rtgW>
   <type>4</type>
</DERCapability>
```

Figure 2.2: Example XML DER Capability Reporting

```
HTTP PUT /sep2/edev/1/der/1/derg

<DERSettings xmlns="urn:ieee:std:2030.5:ns">
   <setGradW>O</setGradW>
   <setMaxA>
      <multiplier>O</multiplier>
      <value>20</value>
   </setMaxA>
   <setMaxW>
      <multiplier>O</multiplier>
      <value>SOOO</value>
   </setMaxW>
   <updatedTime>1483257600</updatedTime>
</DERSettings>
```

Figure 2.3: Example XML DER Settings Reporting

Figure 2.4 shows an IEEE 2030.5 XML example data from [B24] containing a Power Status resource that could be posted by an EVSE (electric vehicle supply equipment – an electric vehicle charging station) identified by the IEEE 2030.5 nomenclature “`/sep2/edev/96/ps`”. As before, the example shows the `HTTP PUT` request followed by the IEEE 2030.5 XML response document.

```
HTTP PUT /sep2/edev/96/ps

<PowerStatus xmlns="http://zigbee.org/sep"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" href="/sep2/edev/96/ps">
   <batteryStatus>4</batteryStatus>
   <changedTime>1487812095</changedTime>
   <currentPowerSource>1</currentPowerSource>
   <estimatedChargeRemaining>9300</estimatedChargeRemaining>
   <PEVInfo>
      <chargingPowerNow>
         <multiplier>3</multiplier>
         <value>-5</value>
      </chargingPowerNow>
      <energyRequestNow>
         <multiplier>3</multiplier>
         <value>22</value>
      </energyRequestNow>
      <maxForwardPower>
         <multiplier>3</multiplier>
         <value>7</value>
      </maxForwardPower>
      <minimumChargingDuration>11280</minimumChargingDuration>
      <targetStateOfCharge>10000</targetStateOfCharge>
      <timeChargeIsNeeded>9223372036854775807</timeChargeIsNeeded>
      <timeChargingStatusPEV>1487812095</timeChargingStatusPEV>
   </PEVInfo>
</PowerStatus>
```

Figure 2.4: Example XML VOLTTRON Power Status

Note that these examples are properly formatted IEEE 2030.5 information exchanges and may not be representative of information exchanges used in the Solar industry.

### 2.3.2 Conversion from XML to JSON

The first approach for using these information exchange structures would be to convert them to native JSON and directly embedding them into the UUDEX JSON structures[3]. Similarly, Figure 2.5 shows an example conversion from Figure 2.3 to JSON, and Figure 2.6 shows an example conversion from Figure 2.4 to JSON. Note that these examples only show the XML to JSON conversion, not the full UUDEX JSON documents that would result from the conversion.

[3] Note that the tool (https://jsonformatter.org/xml-formatter) used to automatically generate these examples resulted in JSON structures that are in “canonical” form order, that is the name-value pairs are ordered alphabetically by name, and the XML metadata strings that define the XML namespace that appear in the XML structure as attributes in the top of the XML tree structure in the type tag are prefixed by underscore characters (_) and located at the end of the JSON structure, rather than at the beginning as in the XML structure. Other translation or formatting tools may produce different results.

```
{
   "DERSettings":{
      "setGradW":"O",
      "setMaxA":{
         "multiplier":"O",
         "value":"20"
      },
      "setMaxW":{
         "multiplier":"O",
         "value":"SOOO"
      },
      "updatedTime":"1483257600",
      "_xmlns":"urn:ieee:std:2030.5:ns"
   }
}
```

Figure 2.5: Example DER Settings translated from XML to native JSON

```
{
   "PowerStatus":{
      "batteryStatus":"4",
      "changedTime":"1487812095",
      "currentPowerSource":"1",
      "estimatedChargeRemaining":"9300",
      "PEVInfo":{
         "chargingPowerNow":{
            "multiplier":"3",
            "value":"-5"
         },
         "energyRequestNow":{
            "multiplier":"3",
            "value":"22"
         },
         "maxForwardPower":{
            "multiplier":"3",
            "value":"7"
         },
         "minimumChargingDuration":"11280",
         "targetStateOfCharge":"10000",
         "timeChargeIsNeeded":"9223372036854775807",
         "timeChargingStatusPEV":"1487812095"
      },
      "_xmlns":"http://zigbee.org/sep",
      "_xmlns:xsi":"http://www.w3.org/2001/XMLSchema-instance",
      "_href":"/sep2/edev/96/ps"
   }
}
```

Figure 2.6: Example VOLTTRON Power Status translated from XML to native JSON

However, this translation will require added coding to perform the translation from XML to JSON for exchange, and from JSON to XML if the UUDEX Publisher or UUDEX Subscriber application or implementation already uses the IEEE 2030.5 XML structure, or if it needs to transform the information back into native IEEE 2030.5 for a different exchange. Additionally, translating XML formatted data into JSON formatted data does not produce an exact representation; specifically, XML can use namespace prefixes to help organize information in the XML structure, while JSON does not have such a construct. Mapping XML formatted data into JSON formatted data, at a minimum, would require a re-naming of the XML tag elements in JSON to account for the XML namespace designation if used (e.g., merging the XML class and XML element name into a single JSON element name), resulting in an imperfect mapping that may not be able to faithfully reproduce the original XML if a reverse translation is required. This could result in difficulties in automatic interpretation of the resulting JSON formatted data for IEEE 2030.5-aware applications, and when re-translation back to XML format for re-transmission using a full IEEE 2030.5 compliant interface.

This approach also has the disadvantage of requiring updates to the UUDEX information structures whenever revisions to the IEEE 2030.5 result in information structure or element changes.

### 2.3.3 Encapsulating XML into JSON

The second approach is to keep the IEEE 2030.5 XML format and encapsulate the XML message inside of a UUDEX JSON string. This has significant advantages over the earlier approach as described below.

UUDEX can support exchanges of DER information using the IEEE 2030.5 XML format between the control center functions of organizations such as aggregators, BAs, GOPs, and RCs, for example, a GOP that receives an IEEE 2030.5 exchange from a DER unit and publishes the information to a BA or RC using UUDEX. Many of these organizations would not inherently have or need full IEEE 2030.5 compliant capabilities but may need to exchange the information contained in the IEEE 2030.5 structures. If UUDEX Subscribers need to interpret the contents of the IEEE 2030.5 document, they can use existing (including open source) implementations of IEEE 2030.5-compliant code to extract the data from the XML document (referred to as deserialization). If the UUDEX Subscriber does not need to interpret the IEEE 2030.5 XML data, it can simply publish it without any added formatting either as a UUDEX JSON document or extract the XML and forward it as an IEEE 2030.5 XML document.

This approach has the added benefit of automatically updating the information exchange format following a revision to IEEE 2030.5 without requiring any upgrades to the UUDEX standard.

A primary design goal of UUDEX was to include support for these kinds of information exchanges and extending UUDEX to include standard DER information models supplies the information in a near native format within the UUDEX exchange infrastructure.

Since UUDEX itself does not need to interpret or process the information, and it has provisions to embed information using other formatting or encoding into a UUDEX JSON-formatted exchange, keeping the message in IEEE 2030.5 XML format and embedding the native XML into the UUDEX JSON is prudent. However, addressing several formatting issues when embedding XML into a JSON structure is necessary. Those issues are:

1. When encapsulating XML into a JSON string, double quote characters (")[4] enclose the entire XML string. XML also uses a similar mechanism for its own strings but allows the use

[4] Note also that both XML and JSON require traditional single and double quote characters, not the “smart quote” characters that Microsoft Word prefers to use, so any code edited in Microsoft Word or copied from a document formatted using Microsoft Word may require the replacement of the smart quote characters with traditional characters.

of either a single quote (') or double quote character to delineate the strings. While JSON allows "escaping" the double quote characters (i.e., using \"), encoding in this manner may require a small amount of processing and buffering to replace the single quote character with the two-character escape sequence, and will add slightly to the length of the XML document string. It will also require processing to "de-escape" the quotes before passing the XML to the IEEE 2030.5 parser for interpretation.

An alternative is to use single quote characters in place of the double quote characters. XML allows this, would not require any added buffering or storage for the translation process, and is a simple and straightforward replacement of one character with another character while constructing the UUDEX JSON document. UUDEX Client processing can perform this conversion maintaining compliance with the IEEE 2030.5 XML format.

Note, however, that if the original XML document uses double-quoted strings that contain single quote (or apostrophe) characters, the embedded single quotes will need to be escaped using an XML escape sequence quoting (i.e., using `&apos;`) since the string transformation results in a single-quoted string. This could require additional buffering and storage for the translation process and is not a straightforward as a single character-for-character substitution but is not an onerous process either. A similar issue would occur if the XML document used single-quoted strings that contain double quotes, necessitating the substitution the double quotes with a different escape sequence (i.e., `&quot;`). While this does not appear to be an issue with IEEE 2030.5 XML documents (unless the XML string values for example in the "`description`" element, contain quote characters), it is a consideration for more generalized XML documents.

2. JSON does not allow for line breaks (formatted using a generic new line, line feed, or carriage return characters) within strings. As with the quotes, these characters could be escape formatted while constructing the UUDEX JSON document (i.e., using `\n` for new line, `\r` for carriage return, or `\f` for form feed[5]), but in most cases, the formatting allows human-readable XML data; automated parsers found in IEEE 2030.5 compliant DER devices and related supervisory control systems do not need or use the formatting and should be able to interpret any of these forms.

   Most XML parsers interpret XML structures as a stream of characters without formatting (except for formatting within quoted strings) and ignore any extraneous blank-space (i.e., multiple adjacent space or tab characters) or line breaks. When formatting the XML data for inclusion into a JSON string, the XML is compressed removing all the extraneous blank space and line-break characters (leaving a single space character to represent the extraneous blank space or line break) not in quoted strings; any required new-line characters included in an XML string are escaped using XML escape sequences (i.e., using `&#xA` [for line feed] or `&#xD` [for carriage return]) allowing that formatting to be maintained once extracted from the XML data. Note that other characters may also need to be escaped in XML (see [B6]), but that formatting should have already been performed when the original XML document was created.

   If necessary, XML formatting tools can re-construct the indented and formatted human-readable XML structures from the compressed string.

[5] JSON also supports using Unicode code points expressed as a hexadecimal escape sequence (as defined in ISO/IEC 10646 [B14]) wrapped in quotation marks, for example, `"\u000A"` as an alternative to `\n`, but this is seldom seen for these characters.

Using these formatting considerations, Figure 2.1 becomes Figure 2.7, Figure 2.2 becomes Figure 2.8, Figure 2.3 becomes Figure 2.9, and Figure 2.4 becomes Figure 2.10 as shown below.

```
<DERStatus xmlns='urn:ieee:std:2030.5:ns'><genConnectStatus><dateTime>l 4563 4
5000</dateTime><value>O</value></genConnectStatus><readingTime>l4 56345000</re
adingTime></DERStatus>
```

Figure 2.7: Example XML DER Status formatted for JSON string

```
<DERCapability xmlns='urn:ieee:std:2030.5:ns'><modesSupported>3FFFFF</modesSup
ported><rtgA><multiplier>O</multiplier><value>20</value></rtgA><rtgW><multipli
er>O</multiplier><value>SOOO</value></rtgW><type>4</type></DERCapability>
```

Figure 2.8: Example XML DER Capability formatted for JSON string

```
<DERSettings xmlns='urn:ieee:std:2030.5:ns'><setGradW>O</setGradW><setMaxA><mu
ltiplier>O</multiplier><value>20</value></setMaxA><setMaxW><multiplier>O</mult
iplier><value>SOOO</value></setMaxW><updatedTime>1483257600</updatedTime></DER
Settings>
```

Figure 2.9: Example XML DER Settings formatted for JSON string

```
<PowerStatus xmlns='http://zigbee.org/sep' xmlns:xsi='http://www.w3.org/2001/X
MLSchema-instance' href='/sep2/edev/96/ps'><batteryStatus>4</batteryStatus><ch
angedTime>1487812095</changedTime><currentPowerSource>1</currentPowerSource><e
stimatedChargeRemaining>9300</estimatedChargeRemaining><PEVInfo><chargingPower
Now><multiplier>3</multiplier><value>-5</value></chargingPowerNow><energyReque
stNow><multiplier>3</multiplier><value>22</value></energyRequestNow><maxForwar
dPower><multiplier>3</multiplier><value>7</value></maxForwardPower><minimumCha
rgingDuration>11280</minimumChargingDuration><targetStateOfCharge>10000</targe
tStateOfCharge><timeChargeIsNeeded>9223372036854775807</timeChargeIsNeeded><ti
meChargingStatusPEV>1487812095</timeChargingStatusPEV></PEVInfo></PowerStatus>
```

Figure 2.10: Example XML VOLTTRON Power Status formatted for JSON string

## 2.4 Naming

IEEE 2030.5 uses an internet Domain Name System (DNS) -based method (DNS-based Service Discovery – DNS-SD [B4]) of discovering, naming, and identifying DER devices in addition to the typical use of DNS for performing hostname to IP address resolution (see Clause 7 of [B11]). While a detailed discussion of those methods and their use in an IEEE 2030.5 infrastructure is beyond the scope of this report, the “Service Instance Name” (see 7.2 of [B11]) supplies a mapping to the services offered by the DER and their instance device number. A UUDEX Client could perform the same DNS and DNS-SD querying that an IEEE 2030.5 client would perform to discover the specific features and configuration of a DER device.

Alternatively, the UUDEX Client could be externally configured with DER device and service names and mappings through configuration files or other static or dynamic configuration methods. These methods could include indirect requesting (see section 2.6) of IEEE 2030.5 DER installations to determine service offerings, resources, or capabilities.

In either case, the format and content of the UUDEX exchanges would be the same.

## 2.5 UUDEX Examples

The following, mostly generic, UUDEX exchange structure shows how UUDEX can encapsulate XML data inside of a JSON formatted message. The UUDEX Information Structures specification [B17] describes the JSON fields. For this exchange, the “`name`” field has specific meaning – it is the name of the DER resource for which the IEEE 2030.5-formatted XML embedded data corresponds.

A minimal structure for a UUDEX exchange for DER data using the overall UUDEX structure from [B17] using the specific UUDEX JSON structure values from Table 2-1 is shown in Figure 2.11.

Table 2-1: JSON Names and Values

| Name | Value (if static) | Description |
|---|---|---|
| `messageID` | `<uuid>` | A unique identifier that specifies a specific UUDEX message. Message exchanges can use the `messageID` in conjunction with the `correlationID` to tie multiple UUDEX messages together. |
| `noun` | `"IEEE2030.5"` | Specifies that the JSON formatted data contains an IEEE 2030.5-formatted XML data as its `contents`.<br><br>Note – by default, UUDEX Clients use a generic current version of IEEE 2030.5 (generally the most current version); if the UUDEX Clients need a specific version of IEEE 2030.5, the noun may specify it, e.g., `IEEE2030.5-2013` for the previous version or `IEEE2030.5-2018` for the current version. |
| `verb` | `"CREATE"` | Specifies that UUDEX is to “create” a new JSON message. |
| `correlationID` | `<uuid>` | The identifier that UUDEX Clients use to connect the UUDEX message to another message, for example the request to which this UUDEX message is the reply. |
| `format` | `"XML"` | Specifies that the format of the data contained in the contents field is XML data. |
| `schema` | | Specifies the location of the JSON schema definition for the JSON structure. |
| `schemaVersion` | | Specifies the version of the JSON schema. |
| `name` | | Specifies the URI for the DER device to which the XML `contents` corresponds.<br><br>Note – the URI in the UUDEX `name` may also be contained in the `href` attribute of the top-level element in the XML document encapsulated in the `contents` of the UUDEX message. Since the `href` attribute is optional, the URI must be specified in the `name` field, even if an `href` is specified later in the XML document. |
| `contents` | | The XML formatted data containing the IEEE 2030.5-formatted information exchange. |

```
{
   "header":{
      "messageID":"b1e7cf98-402e-4f62-81b0-bd57e100b77d",
      "noun":"IEEE2030.5",
      "verb":"CREATE",
   },
   "dataSet":{
      "dataElements":[
         {
            "IEEE2030.5":{
               "schema":"https://www.uudex.org/uudex/0.1/IEEE2030.5",
               "schemaVersion":"0.1",
               "format":"XML",
               "name":"",
               "contents":"<XML data in IEEE 2030.5 'response' format…>"
               }
            }
         }
      ]
   }
}
```

Figure 2.11: Example Minimal UUDEX JSON for IEEE 2030.5 String

Additional parameters in the `header` (as documented in [B17]) can also be specified, for example, to indicate the original source of the data, a timestamp when the message was created, a message hash that can be used to detect message modification, an expiration time for the message.

The UUDEX message can specify other parameters in the `dataSet`, such as `compression` or `encryption` (as documented in [B17]). It can also specify other metadata information, such as a set of `tags` that can supply added context to aid the UUDEX Subscriber to further interpret the data.

The UUDEX message structure for the requested status about "`Inverter A`" from Figure 2.1 would be published by the inverter as shown in Figure 2.12, to be subscribed by a UUDEX Client at the controlling site (which could be a centralized control center, an intermediate control center, or a collection point controller responsible for multiple DER units). For consistency, the name used is the same as the reference in the IEEE 2030.5 `HTTP PUT` command (e.g., `/sep2/edev/1/der/1`), but the message does not need to include the specific element in the message (i.e., `ders`, `dercap`, or `derg` from the examples) since the message already includes it. Because the format uses IEEE 2030.5 formatted nomenclature, it is up to the UUDEX Subscriber to correlate the string "`/sep2/edev/1/der/1`" to the name "`Inverter A`" either through name and service discovery or configuration (see Section 2.4).

```
{
   "header":{
      "messageID":"d2582e46-720a-4f8c-93e0-6d255037395a",
      "noun":"IEEE2030.5",
      "verb":"CREATE"
   },
   "dataSet":{
      "dataElements":[
         {
            "IEEE2030.5":{
               "schema":"https://www.uudex.org/uudex/0.1/IEEE2030.5",
               "schemaVersion":"0.1",
               "format":"XML",
               "name":"/sep2/edev/1/der/1",
               "contents":"<DERStatus xmlns='urn:ieee:std:2030.5:ns'><genConn
ectStatus><dateTime>l 4563 45000</dateTime><value>O</value></genConnectStatus>
<readingTime>l4 56345000</readingTime></DERStatus>"
            }
         }
      ]
   }
}
```

Figure 2.12: Example UUDEX JSON for DER Status Reporting

Note that in the case of both JSON and XML, parsing processing ignores line breaks and blank space not inside of a quoted string since only a human reader uses the formatting. Note also that JSON encapsulates the entire XML formatted document into a JSON string using double quotes. The format processing translates all XML double quotes to single quotes, as supported by the XML specification. The format processing also removes any new-line characters from the XML document resulting in a continuous stream of characters for the XML "contents"

JSON documents transmitted between computers use "compressed format", removing all the extraneous blank-space (except if inside a JSON quoted string) and line-break characters. If necessary, JSON formatting tools can re-construct the indented and formatted human-readable JSON structures from the compressed format.

Figure 2.13 shows the compressed format JSON that UUDEX uses to publish the human-readable formatted JSON structure in Figure 2.12:

```
{"header":{"messageID":"d2582e46-720a-4f8c-93e0-6d255037395a","noun":"IEEE2030
.5","verb":"CREATE"},"dataSet":{"dataElements":[{"IEEE2030.5":{"schema":"https
://www.uudex.org/uudex/0.1/IEEE2030.5","schemaVersion":"0.1","format":"XML","n
ame":"/sep2/edev/1/der/1","contents":"<DERStatus xmlns='urn:ieee:std:2030.5:ns
'><genConnectStatus><dateTime>l 4563 45000</dateTime><value>O</value></genConn
ectStatus><readingTime>l4 56345000</readingTime></DERStatus>"}}]}}
```

Figure 2.13: Example UUDEX JSON for DER Status in compressed format

The UUDEX message structure for the requested status about "`Inverter A`" from Figure 2.2 would be published by the inverter as shown in Figure 2.14, to be subscribed by the UUDEX Client at the control site. For consistency, the name used is the same as the reference in the IEEE 2030.5 `HTTP PUT` command (e.g., `/sep2/edev/1/der/1`), but the message does not need to include the specific element in the message since as before the message already includes it. Because the format uses IEEE 2030.5 formatted nomenclature, it is up to the UUDEX Subscriber to correlate the string "`/sep2/edev/1/der/1`" to the name "`Inverter A`" as before either through name and service discovery or configuration (see Section 2.4).

```
{
   "header":{
      "messageID":"d2582e46-720a-4f8c-93e0-6d255037395a",
      "noun":"IEEE2030.5",
      "verb":"CREATE"
   },
   "dataSet":{
      "dataElements":[
         {
            "IEEE2030.5":{
               "schema":"https://www.uudex.org/uudex/0.1/IEEE2030.5",
               "schemaVersion":"0.1",
               "format":"XML",
               "name":"/sep2/edev/1/der/1",
               "contents":"<DERCapability xmlns='urn:ieee:std:2030.5:ns'><mod
esSupported>3FFFFF</modesSupported><rtgA><multiplier>O</multiplier><value>20</
value></rtgA><rtgW><multiplier>O</multiplier><value>SOOO</value></rtgW><type>4
</type></DERCapability>"
            }
         }
      ]
   }
}
```

Figure 2.14: Example UUDEX JSON for DER Capability Reporting

Figure 2.15 shows the compressed format JSON that UUDEX uses to publish the human-readable formatted JSON structure in Figure 2.14:

```
{"header":{"messageID":"d2582e46-720a-4f8c-93e0-6d255037395a","noun":"IEEE2030
.5","verb":"CREATE"},"dataSet":{"dataElements":[{"IEEE2030.5":{"schema":"https
://www.uudex.org/uudex/0.1/IEEE2030.5","schemaVersion":"0.1","format":"XML","n
ame":"/sep2/edev/1/der/1","contents":"<DERCapability xmlns='urn:ieee:std:2030.
5:ns'><modesSupported>3FFFFF</modesSupported><rtgA><multiplier>O</multiplier><
value>20</value></rtgA><rtgW><multiplier>O</multiplier><value>SOOO</value></rt
gW><type>4</type></DERCapability>"}}]}}
```

Figure 2.15: Example UUDEX JSON for DER Capability in compressed format

Since the `name` field of the UUDEX document specifies the DER resource, rather than specifying it in the URI as IEEE 2030.5 does, UUDEX can combine multiple XML documents into a single UUDEX document. In this case, the two XML documents are for the same device, using the DER device identified by the `name` field in the UUDEX document, allows different DER devices to be combined in a single UUDEX document. Similarly, since each embedded XML document describes the document type (e.g., `DERStatus` or `DERCapability`), the UUDEX receiver is able to properly parse the individual XML documents. Since separate XML documents were combined by the UUDEX Publisher, if they are forwarded on to an IEEE 2030.5 client by the UUDEX Subscriber, they will need to be separated and reformatted into individual IEEE 2030.5 XML documents.

Figure 2.16 shows a single UUDEX message that combines the XML document content from both Figure 2.12 and Figure 2.14.

```
{
   "header":{
      "messageID":"d2582e46-720a-4f8c-93e0-6d255037395a",
      "noun":"IEEE2030.5",
      "verb":"CREATE"
   },
   "dataSet":{
      "dataElements":[
         {
            "IEEE2030.5":{
               "schema":"https://www.uudex.org/uudex/0.1/IEEE2030.5",
               "schemaVersion":"0.1",
               "format":"XML",
               "name":"/sep2/edev/1/der/1",
               "contents":"<DERStatus xmlns='urn:ieee:std:2030.5:ns'><genConn
ectStatus><dateTime>l 4563 45000</dateTime><value>O</value></genConnectStatus>
<readingTime>l4 56345000</readingTime></DERStatus>"
            }
         },
         {
            "IEEE2030.5":{
               "schema":"https://www.uudex.org/uudex/0.1/IEEE2030.5",
               "schemaVersion":"0.1",
               "format":"XML",
               "name":"/sep2/edev/1/der/1",
               "contents":"<DERCapability xmlns='urn:ieee:std:2030.5:ns'><mod
esSupported>3FFFFF</modesSupported><rtgA><multiplier>O</multiplier><value>20</
value></rtgA><rtgW><multiplier>O</multiplier><value>SOOO</value></rtgW><type>4
</type></DERCapability>"
            }
         }
      ]
   }
}
```

Figure 2.16: Example UUDEX JSON for DER Status and Capability Reporting

Figure 2.17 shows the compressed format JSON that UUDEX uses to publish the human-readable formatted JSON structure in Figure 2.16.

```
{"header":{"messageID":"d2582e46-720a-4f8c-93e0-6d255037395a","noun":"IEEE2030
.5","verb":"CREATE"},"dataSet":{"dataElements":[{"IEEE2030.5":{"schema":"https
://www.uudex.org/uudex/0.1/IEEE2030.5","schemaVersion":"0.1","format":"XML","n
ame":"/sep2/edev/1/der/1","contents":"<DERStatus xmlns='urn:ieee:std:2030.5:ns
'><genConnectStatus><dateTime>l 4563 45000</dateTime><value>O</value></genConn
ectStatus><readingTime>l4 56345000</readingTime></DERStatus>"}},{"IEEE2030.5":
{"schema":"https://www.uudex.org/uudex/0.1/IEEE2030.5","schemaVersion":"0.1","
format":"XML","name":"/sep2/edev/1/der/1","contents":"<DERCapability xmlns='ur
n:ieee:std:2030.5:ns'><modesSupported>3FFFFF</modesSupported><rtgA><multiplier
>O</multiplier><value>20</value></rtgA><rtgW><multiplier>O</multiplier><value>
SOOO</value></rtgW><type>4</type></DERCapability>"}}]}}
```

Figure 2.17: Example UUDEX JSON for DER Capability and Status in compressed format

By combining the two XML documents, the total size of the UUDEX document is reduced from 965 bytes (457 for `DERStatus` and 508 for `DERCapability`) to 836 bytes, and the number of messages exchanged is reduced from two to one. The savings in byte count and number of messages can be even greater when additional DER devices or different IEEE 2030.5 XML documents are combined into a single UUDEX document.

Similarly, the UUDEX Client at the control site could update the settings for `Inverter A` (named by the IEEE 2030.5 nomenclature “`/sep2/edev/1/der/1`” from Figure 2.5 by publishing Figure 2.18 for subscription by `Inverter A`:

```
{
   "header":{
      "messageID":"ffcc7389-5d3f-43da-aa28-c1879ccac045"
      "noun":"IEEE2030.5",
      "verb":"CREATE"
   },
   "dataSet":{
      "dataElements":[
         {
            "IEEE2030.5":{
               "schema":"https://www.uudex.org/uudex/0.1/IEEE2030.5",
               "schemaVersion":"0.1",
               "format":"XML",
               "name":"/sep2/edev/1/der/1",
               "contents":"<DERSettings xmlns='urn:ieee:std:2030.5:ns'><setGr
adW>O</setGradW><setMaxA><multiplier>O</multiplier><value>20</value></setMaxA>
<setMaxW><multiplier>O</multiplier><value>SOOO</value></setMaxW><updatedTime>1
483257600</updatedTime></DERSettings>"
            }
         }
      ]
   }
}
```

Figure 2.18: Example UUDEX JSON for DER Settings

Figure 2.19 shows the UUDEX formatted document published by the EVSE encapsulating the VOLTTRON XML documentation about a Power Status resource in Figure 2.4:

```
{
   "header":{
      "messageID":"0fbc8798-d147-432b-85e4-26c27ce81471"
      "noun":"IEEE2030.5",
      "verb":"CREATE"
   },

   "dataSet":{
      "dataElements":[
         {
            "IEEE2030.5":{
               "schema":"https://www.uudex.org/uudex/0.1/IEEE2030.5",
               "schemaVersion":"0.1",
               "format":"XML",
               "name":"/sep2/edev/96/ps",
               "contents":"<PowerStatus xmlns='http://zigbee.org/sep' xmlns:x
si='http://www.w3.org/2001/XMLSchema-instance' href='/sep2/edev/96/ps'><batter
yStatus>4</batteryStatus><changedTime>1487812095</changedTime><currentPowerSou
rce>1</currentPowerSource><estimatedChargeRemaining>9300</estimatedChargeRemai
ning><PEVInfo><chargingPowerNow><multiplier>3</multiplier><value>-5</value></c
hargingPowerNow><energyRequestNow><multiplier>3</multiplier><value>22</value><
/energyRequestNow><maxForwardPower><multiplier>3</multiplier><value>7</value><
/maxForwardPower><minimumChargingDuration>11280</minimumChargingDuration><targ
etStateOfCharge>10000</targetStateOfCharge><timeChargeIsNeeded>922337203685477
5807</timeChargeIsNeeded><timeChargingStatusPEV>1487812095</timeChargingStatus
PEV></PEVInfo></PowerStatus>"
            }
         }
      ]
   }
}
```

Figure 2.19: Example UUDEX JSON for VOLTTRON Power Status

UUDEX can combine multiple IEEE 2030.5 XML documents for different DER instances into a single UUDEX message. This could provide additional efficiencies for an IEEE 2030.5 aggregator sending information about multiple DERs to a BA or GOP. Note that "`dataElements`" is an array of one or more JSON documents (each element of the array a JSON sub-document separated by commas), allowing exchange of multiple UUDEX JSON documents (in this case containing XML documents) within a single UUDEX message.

As an example, using the earlier XML documents Figure 2.3 and Figure 2.4, assume that "`Inverter A`" is somehow associated with "`EVSE 1`". An application can combine the two XML documents into a single UUDEX JSON document and transferred together as shown in Figure 2.20:

```
{
   "header":{
      "messageID":"ee206fa5-ca2f-4a37-8f2a-2681bbb5ae8a",
      "noun":"IEEE2030.5",
      "verb":"CREATE"
   },
   "dataSet":{
      "dataElements":[
         {
            "IEEE2030.5":{
               "schema":"https://www.uudex.org/uudex/0.1/IEEE2030.5",
               "schemaVersion":"0.1",
               "format":"XML",
               "name":"/sep2/edev/1/der/1",
               "contents":"<DERSettings xmlns='urn:ieee:std:2030.5:ns'><setGr
adW>O</setGradW><setMaxA><multiplier>O</multiplier><value>20</value></setMaxA>
<setMaxW><multiplier>O</multiplier><value>SOOO</value></setMaxW><updatedTime>1
483257600</updatedTime></DERSettings>"
            }
         },
         {
            "IEEE2030.5":{
               "schema":"https://www.uudex.org/uudex/0.1/IEEE2030.5",
               "schemaVersion":"0.1",
               "format":"XML",
               "name":"/sep2/edev/96/ps",
               "contents":"<PowerStatus xmlns='http://zigbee.org/sep' xmlns:x
si='http://www.w3.org/2001/XMLSchema-instance' href='/sep2/edev/96/ps'><batter
yStatus>4</batteryStatus><changedTime>1487812095</changedTime><currentPowerSou
rce>1</currentPowerSource><estimatedChargeRemaining>9300</estimatedChargeRemai
ning><PEVInfo><chargingPowerNow><multiplier>3</multiplier><value>-5</value></c
hargingPowerNow><energyRequestNow><multiplier>3</multiplier><value>22</value><
/energyRequestNow><maxForwardPower><multiplier>3</multiplier><value>7</value><
/maxForwardPower><minimumChargingDuration>11280</minimumChargingDuration><targ
etStateOfCharge>10000</targetStateOfCharge><timeChargeIsNeeded>922337203685477
5807</timeChargeIsNeeded><timeChargingStatusPEV>1487812095</timeChargingStatus
PEV></PEVInfo></PowerStatus>"
            }
         }
      ]
   }
}
```

Figure 2.20: Example UUDEX JSON Combining DER Status and VOLTTRON Power Status

## 2.6 Request and Reply

As noted in Section 2.2, UUDEX does not support negotiation, but it does support a request/reply capability by using the `messageID` in a UUDEX Request message as the `correlationID` field of the corresponding UUDEX Reply message. If the `correlationID` field is blank, the UUDEX message is not correlated to any other UUDEX message.

In order for UUDEX to implement this request/reply function two UUDEX Subjects must be established – one for the UUDEX Requestor to publish UUDEX Request messages and another for the UUDEX Responder to publish UUDEX Reply messages. Since each UUDEX Reply message is correlated to its corresponding UUDEX Request message using the `messageID` and `correlationID` fields, the same one-to-one, one-to-many, many-to-one and many-to-many interactions can be supported where it makes sense to the request/reply application.

When using IEEE 2030.5 the request is issued using an `HTTP GET` request. For example, an IEEE 2030.5 compliant server can request the HTTP links used to gather information from a DER device by issuing the following HTTP command:

```
HTTP GET /sep2/edev/1/der
```

Figure 2.21: Example REST Request for the DER List Link for Inverter A

The IEEE 2030.5 compliant DER would respond with a message as shown in Figure 2.22.

```
<DERList href="/sep2/edev/1/der" all="1" results="1" xmlns="urn:ieee:std:2030.
5:ns">
   <DER href="/sep2/edev/1/der/1">
      <DERAvailabilityLink href="/sep2/edev/1/der/1/dera"/>
      <DERCapabilityLink href="/sep2/edev/1/der/1/dercap"/>
      <DERSettingsLink href="/sep2/edev/1/der/1/derg"/>
      <DERStatusLink href="/sep2/edev/1/der/1/ders"/>
   </DER>
</DERList>
```

Figure 2.22: Example XML Response to the DER List Link for Inverter A Request

Since individual UUDEX message exchanges are stateless, UUDEX Responders cannot simply respond with the requested information in the same manner as the IEEE 2030.5 response. Rather, the UUDEX Response must publish a UUDEX Response message that specifies the request parameter as the `name` in the message, the specific HTTP command in the `contents` field, and supply the `messageID` of the UUDEX Request message as the `correlationID` of the UUDEX Response message.

Thus, the equivalent interaction using UUDEX Subjects would be a UUDEX Request message as shown in Figure 2.23 (the UUDEX equivalent of Figure 2.21), and the UUDEX Responder message as shown in Figure 2.24 (the UUDEX equivalent of Figure 2.22). Note that in these examples, the `messageID` from the request is the same as the `correlationID` in the reply, and the identifier for the specific DER (Inverter A) is its IEEE 2030.5 nomenclature "`/sep2/edev/1/der`".

Note also that there is a `status` reply that contains the HTTP response code ("`200`" in this case indicating a successful response) that an IEEE 2030.5 compliant server would return for the request. If the IEEE 2030.5 compliant server cannot successfully complete the request, the `status` in the UUDEX response message will contain the appropriate HTTP response code from the IEEE 2030.5 compliant server, and the `contents` will be either blank or contain supplemental error information provided by the IEEE 2030.5 compliant server.

```
{
   "header":{
      "messageID":"80e61c08-c82b-4a04-9d3e-ca0547ea4e8a",
      "noun":"IEEE2030.5_GET",
      "verb":"REQUEST",
   },
   "dataSet":{
      "dataElements":[
         {
            "IEEE2030.5_GET":{
               "schema":"https://www.uudex.org/uudex/0.1/IEEE2030.5_GET",
               "schemaVersion":"0.1",
               "name":"/sep2/edev/1/der"
               "contents":"GET /sep2/edev/1/der"
               }
            }
         }
      ]
   }
}
```

Figure 2.23: Example UUDEX Request message for the DER List Link for Inverter A

```
{
   "header":{
      "messageID":"0dc0209b-90d0-4dbe-9f41-bde28d708fe1",
      "correlationID":"80e61c08-c82b-4a04-9d3e-ca0547ea4e8a",
      "noun":"IEEE2030.5_GET",
      "verb":"REPLY",
   },
   "dataSet":{
      "dataElements":[
         {
            "IEEE2030.5_GET":{
               "schema":"https://www.uudex.org/uudex/0.1/IEEE2030.5_GET",
               "schemaVersion":"0.1",
               "name":"/sep2/edev/1/der"
               "status":"200"
               "contents":"<DERList href='/sep2/edev/1/der' all='1' results='
1' xmlns='urn:ieee:std:2030.5:ns'><DER href='/sep2/edev/1/der/1'><DERAvailabil
ityLink href='/sep2/edev/1/der/1/dera'/><DERCapabilityLink href='/sep2/edev/1/
der/1/ dercap'/> <DERSettingsLink href='/sep2/edev/1/der/1/derg'/><DERStatusLin
k href='/sep2/edev/1/der/1/ders'/></DER></DERList>"
               }
            }
         }
      ]
   }
}
```

Figure 2.24: Example UUDEX Reply message to the DER List Link for Inverter A Request

Note that since UUDEX interactions are asynchronous, there is no inherent timeout mechanism supported. If the UUDEX responder is directly interfacing with an IEEE 2030.5 compliant device, the IEEE 2030.5 interaction may timeout and provide an indication that the IEEE 2030.5 device was unresponsive. However, a UUDEX Requestor cannot determine whether a UUDEX Message has been retrieved by a UUDEX Responder, or if the UUDEX Responder is unable to form a UUDEX Reply message, the UUDEX Requestor is responsible for providing a timeout process (e.g., marking a `messageID` as no longer expecting a `correlationID` message), and handling any UUDEX Reply messages that contain `correlationIDs` that should no longer be processed.

UUDEX can implement other IEEE 2030.5 request/reply interactions, including Discovery interactions, in a comparable manner.

# 3.0 Conclusion

Using an existing format for exchange of DER related information using UUDEX can reduce implementation efforts. This is especially true if UUDEX Publishers or UUDEX Subscribers already use the chosen format, or if code for interpreting the existing format is readily available for use by the UUDEX Publishers or UUDEX Subscribers.

This report selected the IEEE 2030.5 format since it is a published, widely implemented, and relatively mature format (having initially been developed by the ZigBee Alliance and HomePlug Alliance as the Smart Energy Profile 2.0 [B25], and subsequently been through at least one revision by the IEEE standardization process).

As noted, the proposed UUDEX approach re-uses much of the format and syntax for the messages and interactions from IEEE 2030.5 as practical, so implementing this UUDEX equivalent alongside of an existing IEEE 2030.5 implementation should be straightforward, not only for the information models, but as well for the name syntax.

The IEEE standardization effort for *IEEE Std P2030.103, Draft Standard for Universal Utility Data Exchange (UUDEX)* will use concepts from this report to describe DER-specific communications so that when published, UUDEX will support DER information exchanges in addition to those already included in the standard

# 4.0 References

[B1] Apache Kafla. Available at https://kafka.apache.org/documentation/ (accessed 02/24/2023)

[B2] “Application integration at electric utilities - System interfaces for distribution management”, IEC 61968 (multiple parts), International Electrotechnical Commission, Geneva, Switzerland. Available from https://webstore.iec.ch.

[B3] Bray, T., Ed., “The JavaScript Object Notation (JSON) Data Interchange Format”, STD 90, RFC 8259, DOI 10.17487/RFC8259, December 2017, Available at https://www.rfc-editor.org/info/rfc8259 (accessed 02/28/2023). [See also European Computer Manufacturers Association standard ECMA-404, “The JSON (JavaScript Object Notation) Data Interchange Syntax” available at https://www.ecma-international.org/publications-and-standards/standards/ecma-404/ (accessed 04/17/2023)].

[B4] Cheshire, S. and M. Krochmal, "DNS-Based Service Discovery", RFC 6763, DOI 10.17487/RFC6763, February 2013, https://www.rfc-editor.org/info/rfc6763.

[B5] “Energy management system application program interface (EMS-API) - ALL PARTS”, IEC 61970 (all parts), International Electrotechnical Commission, Geneva, Switzerland, 2023. Available from https://webstore.iec.ch.

[B6] Extensible Markup Language (XML) 1.0 (Fifth Edition), World Wide Web Consortium, 2008 (updated 2013). Available at https://www.w3.org/TR/xml/ (accessed 02/297/2023)

[B7] Fielding, R. T., “Architectural Styles and the Design of Network-based Software Architectures. ”Ph.D. diss., University of California, Irvine, 2000 Available at http://www.ics.uci.edu/~fielding/pubs/dissertation/top.htm (accessed 03/07/2023).

[B8] “Framework for energy market communications”, IEC 62325 (multiple parts), International Electrotechnical Commission, Geneva, Switzerland. Available from https://webstore.iec.ch.

[B9] *Guidelines for Inter-Control Center Communications Protocol (ICCP) Implementation: Plant Controls to Dispatch Computer*, EPRI, Palo Alto, CA: 1999. TR-113652. Available at https://restservice.epri.com/publicdownload/TR-113652/0/Product (accessed 02/24/2023)

[B10] "IEEE Standard for Electric Power Systems Communications-Distributed Network Protocol (DNP3)," in IEEE Std 1815-2012 (Revision of IEEE Std 1815-2010), vol., no., pp.1-821, 10 Oct. 2012, doi: 10.1109/IEEESTD.2012.6327578.

[B11] "IEEE Standard for Smart Energy Profile Application Protocol," in IEEE Std 2030.5-2018 (Revision of IEEE Std 2030.5-2013), vol., no., pp.1-361, 21 Dec. 2018, doi: 10.1109/IEEESTD.2018.8608044.

[B12] “IEEE Standard for Universal Utility Data Exchange (UUDEX)” in Draft IEEE Std P2030.103. Unpublished. Available at https://standards.ieee.org/ieee/2030.103/10884/ (accessed 03/22/2023)

[B13] “Information technology Open Systems Interconnection The Directory: Public key and attribute certificate frameworks” Rec. ITU T X.509: ISO/IEC 9594-8, International Electrotechnical Commission, Geneva, Switzerland, 19 Feb. 2019. Previous version freely available at https://www.itu.int/rec/dologin_pub.asp?lang=e&id=T-REC-X.509-201910-I!!PDF-E&type=items (accessed 03/28/2023).

[B14] “Information technology — Universal coded character set (UCS)”, ISO/IEC 10646:2020, International Electrotechnical Commission, Geneva, Switzerland, 2020.

[B15] J. Johnson, B. Fox, K. Kaur, and J. Anandan, "Evaluation of Interoperable Distributed Energy Resources to IEEE 1547.1 Using SunSpec Modbus, IEEE 1815, and IEEE 2030.5," in IEEE Access, vol. 9, pp. 142129-142146, 2021, doi: 10.1109/ACCESS.2021.3120304.

[B16] Mix, Scott R., Rice, Mark J., Neumann, Scott, Schmidt, Charles M., Sridhar, Siddharth, Singh, Surya V., Gonzales-Perez, Carlos, Cohen, Michael L., Peloquin, Chris, and Bobka, Trevor. 2021. "Universal Utility Data Exchange (UUDEX) Functional Design Requirements - Rev 1". United States. https://doi.org/10.2172/1839604. Available at https://www.osti.gov/servlets/purl/1839604 (accessed 03/28/2023).

[B17] Mix, Scott R., Rice, Mark J., Sridhar, Siddharth, Schmidt, Charles M., Raju, Srini, Gonzales-Perez, Carlos, and Bharadwaj, Debraj. 2021. "Universal Utility Data Exchange (UUDEX) Information Structures - Rev. 1". United States. https://doi.org/10.2172/1839617. Available at https://www.osti.gov/servlets/purl/1839617 (accessed 03/28/2023).

[B18] Mix, Scott R., Welsh, Jeffrey D., Schmi dt, Charles M., and Farnsworth, Ted. 2021. "Universal Utility Data Exchange (UUDEX) – Security and Administration - Rev.1". United States. https://doi.org/10.2172/1839601. Available at https://www.osti.gov/servlets/purl/1839601 (accessed 03/28/2023).

[B19] NERC, 2018 “NERC Functional Model”. United States. Available at https://www.nerc.com/pa/Stand/Functional%20Model%20Advisory%20Group%20DL/Functional_Model_V5.1_clean_10082019.pdf, (accessed 01/31/2023)

[B20] Neumann, Scott, Lochner, Jeramy L., Sridhar, Siddharth, Mix, Scott R., Kuchar, Olga A., Singh, Surya V., Rice, Mark J., and Schmidt, Charles M. 2021. "Universal Utility Data Exchange (UUDEX) – Protocol Design - Rev 1". United States. https://doi.org/10.2172/1839599. Available at https://www.osti.gov/servlets/purl/1839599 (accessed 03/28/2023).

[B21] Neumann, Scott, Schmidt, Charles M., Cohen, Michael L., Sridhar, Siddharth, and Mix, Scott R., 2021. "Universal Utility Data Exchange (UUDEX) - Workflow Design - Rev 1". United States. https://doi.org/10.2172/1839613. Available at https://www.osti.gov/servlets/purl/1839613 (accessed 03/28/2023).

[B22] RabbitMQ. [Online] available at https://www.rabbitmq.com/documentation.html (accessed 02/24/2023)

[B23] “SunSpec Modbus”, Available at https://sunspec.org/sunspec-modbus-specifications/, (accessed 01/24/2023)

[B24] The VOLTTRON Community, “IEEE 2030.5 DER Support” Available at https://volttron.readthedocs.io/en/releases-7.x/specifications/ieee2030_5_agent.html (accessed 01/24/2023)

[B25] “Smart Energy Profile 2.0 Public Application Protocol Specification”, ZigBee Alliance and HomePlug Powerline Alliance liaison, 2021. Available at https://sunspec.org/wp-content/uploads/2015/06/Zigbee-SEP-2-docs-11-0167-18-seed-app-spec-draft-for-editors-review.pdf (accessed 02/24/2023)