# OpenADR 3 Program Description - ADVISOR Information Model

Programmatic exchange of service capabilities of DERs between DER aggregators and distribution system operators


Contributors
Introduction and Context
OpenADR 3 Transmission Framework
    Roles and Object Model
    OpenADR 3 Program Description
    Conveying Meaning with Enumerations
    Security Model
    OpenADR 3 Connection Flow
        Step 1: VEN Onboarding
        Step 2: VEN Authentication
OpenADR 3 Mapping for ADVISOR
    Primary System Roles in ADVISOR
    ADVISOR Program Usage
    Mapping ADVISOR Parameters to the OpenADR 3 resource Object
    VEN Submission of ADVISOR Data
    Utility Retrieval and Use of ADVISOR Data
    Maintaining ADVISOR Records Over Time
Appendix A
    Diagram script for https://sequencediagram.org/


## Contributors


- Frank Sandoval - OpenADR Alliance
- AHM Jakaria - EPRI
- Daniel Lowe - EPRI


## Introduction and Context

Today, utilities often lack sufficient visibility into distributed energy resources (DER) that are already connected to the grid and are capable of providing grid services within a given service area. To effectively account for DER potential in planning activities and to define appropriate technical requirements for service programs, utilities require access to a core set of foundational information, including:

- Where DERs are located
- What their capabilities and types are
- How their capabilities can be managed at scale

EPRI, along with industry stakeholders such as utilities and DER aggregators, has defined the Automated DER Visibility of Service-Oriented Resources (ADVISOR) data elements to represent such foundational information. These data elements are basically a set of attributes of individual or cluster of DERs, and are referred to as ADVISOR parameters in this document.

In current practice, DER aggregators often provide this information through presentations, documents, spreadsheets, emails, and meetings. These manual exchanges are time-consuming, difficult to standardize, and not scalable across many utilities, aggregators, programs, and DER portfolios.

ADVISOR addresses this gap by defining a consistent, machine-readable information model for periodic DER visibility updates. It gives aggregators an easier way to submit and maintain information about their service-capable DERs while enabling utilities to automate validation and integrate the information into their systems of record.

OpenADR 3 was selected as the communication protocol for this information exchange because it aligns well with these requirements, particularly for periodic information exchange use cases rather than continuous, real-time operational control and monitoring.

Key considerations include:
- Information may be provided periodically, rather than continuously.
- Systems do not need to maintain permanent connectivity.
- A standardized format is essential to ensure consistent interpretation across organizations.

This document describes how OpenADR 3 can be used as the transmission framework for ADVISOR information. It first introduces the basic API concepts needed to understand OpenADR 3, then explains the OpenADR 3 roles and security model, and then focuses on the mapping of ADVISOR parameters into OpenADR 3 resources. The central mapping described in this document is the use of the OpenADR 3 VEN resource to represent the DER population or portfolio for which ADVISOR awareness information is being submitted. ADVISOR-specific parameters are carried as structured attributes within the VEN payload, allowing the VTN to store the information and allowing utility business logic to retrieve and use it for planning, screening, analytics, and RFP design.

The mapping intentionally separates transport from content. OpenADR 3 provides the API structure, authentication and authorization framework, object model, request/response handling, and JSON encoding. ADVISOR provides the content semantics: location, device type, service magnitude, timing response, availability, and communication/access information. This separation allows implementers to use an existing standards-based communication framework

while preserving the ability to define and standardize ADVISOR-specific information items over time.

# OpenADR 3 Transmission Framework

This chapter introduces only the OpenADR 3 concepts needed to understand the ADVISOR implementation described in the next chapter. OpenADR 3 is a standardized, REST-based API framework in which a client sends an HTTP request to a defined endpoint and a server returns a structured response, typically encoded as JSON. The framework uses standard HTTP methods, published OpenAPI schemas, defined object representations, and consistent error handling to support interoperable information exchange.

Key characteristics of OpenADR 3 include:

- Use of standard HTTP/HTTPS communication
- Structured data encoded in JSON
- Formal definitions provided through OpenAPI (Swagger) specifications
- Clearly defined endpoints, schemas, and error handling rules

The following subsections explain the roles, object model, Program Description concept, enumerations, security, and connection flow that form the common OpenADR 3 foundation. Chapter 3 then defines how ADVISOR applies that foundation.

## Roles and Object Model

The Virtual Top Node (VTN) hosts the OpenADR 3 API and acts as the server. The Virtual End Node (VEN) invokes that API and acts as the client. In this document, **VEN** and **VTN** refer to these software roles, while **ven object** and **resource object** refer to representations stored or exchanged through the API. A resource object represents an electrical load or other system associated with a ven object. OpenADR 3 also defines programs, reports, events, subscriptions, and other objects; the applicable Program Description determines which are used for a particular business scenario.

## OpenADR 3 Program Description

An OpenADR 3 Program Description specifies how the common OpenADR 3 framework is applied to a particular business scenario. It does not redefine the protocol; it identifies the required objects, message sequence, permitted enumeration identifiers or private strings, payload conventions, and application-level behavior needed for interoperable implementation. This document serves as the Program Description for ADVISOR and defines those choices in the next chapter.
Implementers should use this Program Description with the applicable OpenADR 3 specification, API definition, Definitions document, and User Guide available from openadr.org.

# Conveying Meaning with Enumerations

An enumeration is a standardized label used by a Program Description to define the meaning of a value. In an HTTP request payload, a **valuesMap** object - such as an entry in a resource's **attributes** array - uses a **type** field for the enumeration label and a **values** array for the related data. The enumeration definition may also specify the data type, format, units, or allowed values so the VEN and VTN interpret the information in the same way.

For example, the value 6700 has no clear meaning by itself. If the **type** is ADVISOR_MAX_GENUP_LOADDOWN and its definition specifies watts as the unit, both systems understand it as the maximum generation-up or load-down capability. This common definition also supports automated validation.

The OpenADR Alliance publishes human- and machine-readable enumeration schemas in its public OpenADR 3 schema repository, including ADVISOR-specific definitions that provide common identifiers and value constraints for implementers.

# Security Model

OpenADR 3 uses TLS to protect data in transit and OAuth 2.0 to authorize access to protected API operations. Mutual TLS is optional and is used only when required by the selected security profile or implementation. When mutual TLS is enabled, the VEN and VTN authenticate one another with digital certificates during connection establishment. For protected API operations, the VEN obtains a bearer token and includes it in the Authorization header of each protected request. TLS protects the connection, while the token conveys the client's permitted scopes.

Note that OpenADR 3 v1.1.0 will define enhanced security profiles, including mTLS and message signing for those VTNs that wish to implement additional security features.

# OpenADR 3 Connection Flow

The connection flow separates infrequent setup from recurring runtime activity: the parties first complete onboarding, the VEN then obtains an access token, and the VEN uses that token to invoke authorized VTN API endpoints. The examples use cURL only to illustrate the HTTP exchanges; production implementations may use any conforming client library or integration framework.

## Step 1: VEN Onboarding

VEN onboarding is the setup stage that occurs before the VEN sends OpenADR API requests and before it authenticates to obtain a token. It is typically performed out-of-band through business and IT processes such as a web portal, email-based intake, contract workflow, or vendor registration process rather than through the OpenADR protocol itself. The onboarding process includes the following:

- **Identify and register the VEN provider:** The VTN operator confirms which organization is acting as the VEN provider and creates the appropriate business and support records.
- **Confirm scope and data-sharing expectations:** The parties agree on what information will be exchanged, how often it will be updated, what population it represents, and what the receiving organization may use it for.
- **Establish policy and legal agreements:** Required terms of use, nondisclosure agreements, data privacy requirements, cybersecurity policies, data retention rules, and sensitive-information handling rules are completed.
- **Complete security setup:** The parties establish how the VEN will prove its identity and connect securely, including certificate exchange for mutual TLS when that option is selected, OAuth configuration, and API scopes or permissions.
- **Provision access and environments:** The VTN operator provides sandbox or production access, connection prerequisites, IP allowlists, firewall requirements, and implementation checklists as needed.
- **Provide technical connection details:** At minimum, onboarding gives the VEN the VTN base URL, client ID, client secret or equivalent credential, and the authentication-service address if it is not discoverable from the VTN endpoint.

In simple terms, onboarding answers who the VEN provider is, what information will be exchanged and under what rules, and how the VEN software will connect securely.

## Step 2: VEN Authentication

After onboarding, the VEN discovers or uses the assigned token endpoint, submits its client credentials, and receives a bearer token with authorized scopes. It then includes that token in subsequent protected requests. The following example illustrates these steps.

1. The VEN determines where to request an access token. This may be done by calling the VTN's authentication-server discovery endpoint, as shown in the example, or by using the authentication-service URL for tokens that was provided during onboarding.

   **Example request:**
   ```
   curl -X GET https://{VTN_BASE_URL}/OpenADR3/1.1.0/auth/server
   ```

   **Example response:**
   ```
   {
    "tokenURL": "https://{VTN_BASE_URL}/OpenADR3/1.1.0/auth/token"
   }
   ```

2. The VEN requests a bearer token from the authentication service using its assigned client credentials.

   **Example request:**
   ```
   curl -X POST -H "Content-Type:
   application/x-www-form-urlencoded" -d @token_request_body.txt
   https://{VTN_BASE_URL}/OpenADR3/1.1.0/auth/token
   ```

```
token_request_body.txt =
grant_type=client_credentials&client_id=<client_id>&client_secr
et=<client_secret>&scope=write_vens
```

**Example response:**

```
{
  "access_token": "dmVuX2NsaWVudDo5OTk=",
  "scope": [
    "read_all",
    "read_ven_objects",
    "read_targets",
    "write_reports",
    "write_subscriptions",
    "write_vens"
    ],
  "token_type": "Bearer"
}
```

3. The VEN includes the bearer token in the authorization header for subsequent API requests to create, update, read, or delete protected OpenADR resources.

Figure 1 summarizes the authorization flow used to obtain a bearer token before the VEN accesses protected VTN endpoints.

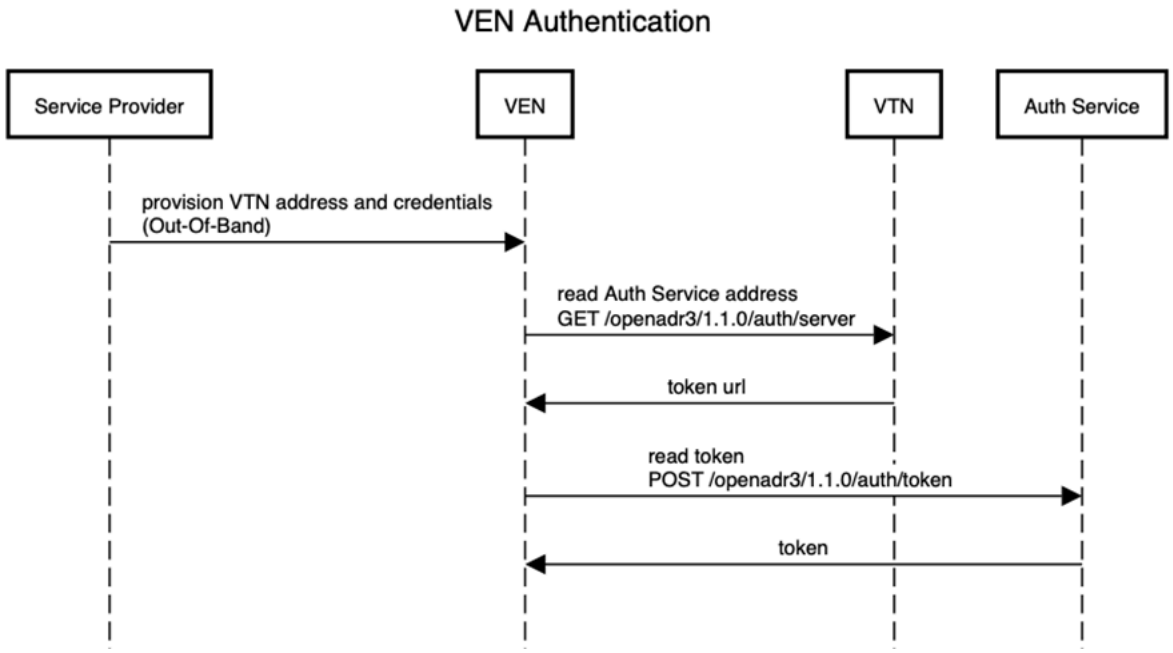



Figure 1: Message sequence for VEN to obtain a security token from the VTN's authentication service.

With these common OpenADR 3 concepts established, the next chapter defines the ADVISOR-specific roles, object usage, enumeration definitions, payload structure, and message sequences.

# OpenADR 3 Mapping for ADVISOR

This chapter defines how the ADVISOR Program Description applies the OpenADR 3 object and transport framework. In the recommended deployment model, the aggregator operates the VEN client and the utility hosts the VTN. A **ven object** identifies the submitting VEN, and each associated **resource object** represents one logical DER resource or cluster. ADVISOR parameters are carried as **valuesMap** entries in the resource object's **attributes** array. The exchange supports periodic service-capability and visibility updates rather than real-time telemetry or dispatch control.

## Primary System Roles in ADVISOR

- **VTN (Virtual Top Node)**: Typically, a utility or system operator that hosts OpenADR as the server and exposes the API endpoints used to receive or publish information.
- **VEN (Virtual End Node)**: Typically, an aggregator, vendor, or DER service provider that acts as the client and sends DER information to the VTN.

As a general practice, and for the deployment model recommended in this document, the utility hosts the VTN and the aggregator hosts the VEN. However, the parties may reverse these roles when mutually agreed upon, provided that their implementation, security, and data-exchange responsibilities are clearly defined. The message sequences in this document describe the recommended deployment model.

## ADVISOR Program Usage

As explained in the earlier **OpenADR 3 Program Description** section, a Program Description defines how the common OpenADR 3 framework is applied to a specific business scenario. For ADVISOR, that scenario is the exchange of DER awareness and service-capability information between aggregators and utilities. The aggregator identifies one or more logical DER resources or clusters, prepares and maintains the corresponding ADVISOR information, and submits it through its VEN. The utility receives the information through its VTN and makes it available to authorized systems for validation and downstream use.

Each utility-aggregator relationship may be implemented separately. A utility may operate or designate its own VTN endpoint, and an aggregator working with multiple utilities may maintain separate VEN-to-VTN integrations for each relationship. The parties may establish utility-specific endpoints, credentials, authorization scopes, accepted resource groupings, submission requirements, and update expectations during onboarding.

ADVISOR information may be submitted during initial onboarding or portfolio registration, at an agreed periodic interval, and when a material change occurs. Examples of material changes include changes in device count, location, service capability, availability, timing characteristics, or communication status. These submissions maintain an updated view of service potential without requiring continuous connectivity or real-time telemetry.

After receiving the information, the utility may validate the submitted attributes, integrate them into its system of record, and apply its own planning, screening, forecasting, analytics, program-design, or eligibility rules. Submission through ADVISOR does not require the utility to accept a resource into a program, and the VTN does not necessarily serve as the utility's permanent system of record.
The submitted information describes the potential capability of a logical DER resource or cluster; it does not confirm that the full capability is available at every moment or commit the resource to provide a service. Program enrollment, dispatch, operational telemetry, settlement, and performance verification are outside the scope of this Program Description. The next section defines how this ADVISOR information is represented using OpenADR 3 ven and resource objects.

# Mapping ADVISOR Parameters to the OpenADR 3 resource Object

OpenADR 3 defines a resource object that can be created, read, updated, and deleted through the VTN API. Each resource object is associated with one ven object, which must exist on the VTN before the resource is created. The ven object identifies the submitting VEN, while the resource object contains the ADVISOR information for a logical DER resource or cluster. One ven object may therefore be associated with multiple resource objects.

A logical resource may represent a single DER, several similar DERs, or a larger cluster that is described and maintained as one service-capable resource. For example, an aggregator could create one resource object for a portfolio of residential thermostats in a defined geographic area and another for a group of commercial batteries with different capabilities or availability. Separate resource objects allow each grouping to have its own location, device composition, service magnitude, timing, availability, and communication information.

Each ADVISOR parameter is represented by one **valuesMap** entry in the resource object's **attributes** array. The **type** field identifies the parameter, and the **values** array contains its data. For example, **ADVISOR_NUM_DEVICES_REPRESENTED** may use **values** of [142], while **ADVISOR_DEVICE_TYPE** may use values such as ["THERMOSTAT_COOLING_ONLY", "THERMOSTAT_HEAT_PUMP"]. Multiple valuesMap entries can be included in the same attributes array to describe the resource fully.

The ADVISOR enumeration definitions provide the standardized **type** identifiers and the rules for their values, including the expected data type, units, format, or allowed choices. These definitions allow the VEN to construct consistent payloads and the VTN to validate and interpret

them. For example, a latitude parameter must contain a numeric value, while a device-type parameter must use one or more recognized text values. The complete ADVISOR schema is available in the OpenADR public GitHub repository. This approach adds ADVISOR-specific content without changing the core OpenADR 3 resource model.
ADVISOR parameters can be logically grouped into categories. These groupings are not encoded in the interface but are offered here to aid comprehension of the design intent behind parameter definitions. Table 1 summarizes the major ADVISOR information categories.

Table 1. ADVISOR parameter categories and their usage

| **ADVISOR Category** | **What It Describes** | **Utility Value** |
|---|---|---|
| Device Type | Types and counts of DER devices represented in the submitted portfolio or cluster. | Supports assessment of portfolio composition and applicable service capabilities. |
| Location | Geographic footprint of aggregated DER, typically using bounding areas rather than individual device locations. | Supports geographic assessment while preserving aggregation. |
| Quantity | Real and reactive power service capability and directional service potential. | Enables suitability assessment for planning or operational needs. |
| Timing | Ramp rates, service duration, recovery time, and advance notice requirements. | Supports matching DER capability to appropriate grid services. |
| Availability | Seasonal, daily, and participation-limit constraints on when resources may be available. | Supports realistic program design and forecasting. |
| Communications | Communication status, access methods, control paths, and fallback behavior. | Provides confidence that signals can be delivered reliably. |

## VEN Submission of ADVISOR Data

After onboarding and authentication, the VEN submits ADVISOR information in two stages. It first creates or identifies its ven object through the VTN's **/vens** endpoint. It then creates one **resource object** through the **/resources** endpoint for each logical DER resource or cluster being described. The VEN uses the identifier returned for the ven object as the **venID** in each associated resource request.

Each protected request includes a valid bearer token in the Authorization header and a JSON payload that conforms to the applicable OpenADR 3 schema and ADVISOR enumeration definitions. The examples below use illustrative OpenADR 3 v1.1.0-style endpoint paths, identifiers, dates, and values. They show the creation of a ven object followed by a resource object containing representative ADVISOR parameters for device composition, location, service magnitude, timing, availability, and communication access.

**Sample request by VEN to create a ven object:**

```
curl -X POST
-H "Content-Type: application/json"
-H "Authorization: Bearer <ven_access_token>"
-d @advisor_oadr_ven_request.json
https://{VTN_BASE_URL}/OpenADR3/1.1.0/vens
advisor_oadr_ven_request =
{
    "venName": "ADVISOR-VEN-HQNSB8",
    "attributes": null
}
```

**Sample response from VTN:**

```
{
    "id": "ven-object-999",
    "createdDateTime": "2026-06-15T09:30:00.000Z",
    "modificationDateTime": "2026-06-15T09:30:00.000Z",
    "objectType": "VEN",
    "clientID": "249rj49jiej",
    "targets": null,
    "venName": "ADVISOR-VEN-HQNSB8",
    "attributes": null
}
```

**Sample request by the VEN to create a resource object:**
Note: The VEN repeats this request for each logical resource or cluster.

```
curl -X POST
-H "Content-Type: application/json"
-H "Authorization: Bearer <ven_access_token>"
-d @advisor_oadr_resource_request.json
https://{VTN_BASE_URL}/OpenADR3/1.1.0/resources
advisor_oadr_resource_request =
{
  "resourceName": "ADVISOR-HQNSB8-RESOURCE-00001",
  "venID": "ven-object-999",
  "attributes": [
    {
      "type": "ADVISOR_NUM_DEVICES_REPRESENTED",
      "values": [142]
    },
    {
      "type": "ADVISOR_DEVICE_TYPE",
      "values": [
        "THERMOSTAT_COOLING_ONLY",
```

```
      "THERMOSTAT_HEAT_PUMP"
    ]
  },
  {
    "type": "ADVISOR_FACILITY_TYPE",
    "values": [
      "RESIDENTIAL",
      "COMMERCIAL"
    ]
  },
  {
    "type": "ADVISOR_LAT_LON_ACCURACY",
    "values": ["APPROXIMATE"]
  },
  {
    "type": "ADVISOR_MIN_LATITUDE",
    "values": [24.216735]
  },
  {
    "type": "ADVISOR_MAX_LATITUDE",
    "values": [24.569989]
  },
  {
    "type": "ADVISOR_MIN_LONGITUDE",
    "values": [-105.903454]
  },
  {
    "type": "ADVISOR_MAX_LONGITUDE",
    "values": [-84.047537]
  },
  {
    "type": "ADVISOR_SERVICE_CAPABILITY_TYPE",
    "values": [
      "ACTIVE_POWER",
      "REACTIVE_POWER"
    ]
  },
  {
    "type": "ADVISOR_MAX_GENUP_LOADDOWN",
    "values": [5100]
  },
  {
    "type": "ADVISOR_MAX_LOADUP_GENDOWN",
    "values": [2500]
  },
  {
    "type": "ADVISOR_ADVANCE_NOTIFICATION_REQUIRED",
```

```
      "values": [true]
    },
    {
      "type": "ADVISOR_MIN_ADVANCE_NOTIFICATION_TIME",
      "values": [300]
    },
    {
      "type": "ADVISOR_MAX_GENUP_LOADDOWN_DURATION",
      "values": [7200]
    },
    {
      "type": "ADVISOR_MAX_GENDOWN_LOADUP_DURATION",
      "values": [3600]
    },
    {
      "type": "ADVISOR_REQUIRED_RECOVERY_TIME",
      "values": [3600]
    },
    {
      "type": "ADVISOR_COMMUNICATION_STATUS",
      "values": [
        "DEVICES_COMMUNICATION_CONNECTED"
      ]
    },
    {
      "type": "ADVISOR_AVAILABLE_ACCESS_METHODS",
      "values": [
        "DIRECT_ACCESS_TO_DEVICE_AVAILABLE",
        "ACCESS_THROUGH_MANUFACTURER_AGGREGATOR_AVAILABLE",
        "ACCESS_THROUGH_THIRD_PARTY_AGGREGATOR_AVAILABLE"
      ]
    },
    {
      "type": "ADVISOR_DEVICE_LEVEL_PROTOCOL",
      "values": [
        "PROPRIETARY_VENDOR_DEFINED"
      ]
    },
    {
      "type": "ADVISOR_AGGREGATOR_VPP_LEVEL_PROTOCOL",
      "values": [
        "OPENADR_3_1"
      ]
    }
  ]
}
```

**Sample response from VTN:**
Note: The venID in the request associates the new resource object with the previously created ven object. The VTN returns that association in the stored resource representation.

```
{
  "id": "0",
  "createdDateTime": "2026-08-15T16:00:00Z",
  "modificationDateTime": "2026-08-15T16:00:00Z",
  "objectType": "RESOURCE",
  "resourceName": "ADVISOR-HQNSB8-RESOURCE-00001",
  "attributes": [
    "<attributes as above>"
  ],
  "venID": "ven-object-999"
}
```

# Utility Retrieval and Use of ADVISOR Data

After the VTN receives the ven and resource objects, authorized utility business logic can retrieve and process the submitted ADVISOR data. This processing may validate attributes, apply utility-specific rules, and transfer accepted information into the utility's system of record for planning, program design, forecasting, screening, or analytics. The VTN provides the exchange interface and object access but does not need to be the permanent system of record.

The examples below first retrieve ven objects and then retrieve the resource objects associated with a selected **venID**. An implementation may retrieve a collection or a specific object, subject to the endpoints, filters, and authorization scopes supported by the VTN.

**Sample request by utility business logic to retrieve ven(s):**

```
curl -X GET
-H "Accept: application/json"
-H "Authorization: Bearer <business_logic_access_token>"
https://{VTN_BASE_URL}/OpenADR3/1.1.0/vens
```

**Sample response from VTN with ven information:**

```
[
  {
    "id": "ven-object-999",
    "createdDateTime": "2026-06-15T09:30:00.000Z",
    "modificationDateTime": "2026-06-15T09:30:00.000Z",
    "objectType": "VEN",
    "clientID": "249rj49jiej",
    "targets": null,
    "venName": "ADVISOR-VEN-HQNSB8",
    "attributes": null
  }
]
```

The business logic can then read one or more resources that are tied to a specific ven object.

**Sample request by utility business logic to retrieve resources for a specific ven:**

```
curl -X GET
-H "Accept: application/json"
-H "Authorization: Bearer <business_logic_access_token>"
https://{VTN_BASE_URL}/OpenADR3/1.1.0/resources?venID={venID}
```

**Sample response from VTN with resource information for a specific ven:**

```
[
  {
    "id": "resource-object-001",
    "createdDateTime": "2026-06-15T09:30:00.000Z",
    "modificationDateTime": "2026-06-15T09:30:00.000Z",
    "objectType": "RESOURCE",
    "resourceName": "ADVISOR-HQNSB8-RESOURCE-00001",
    "venID": "ven-object-999",
    "attributes": "<attributes as above>"
  }
]
```

Figure 2 illustrates the end-to-end sequence in which the VEN creates or identifies its ven object, submits one or more associated resource objects containing ADVISOR information, and makes those objects available for authorized utility retrieval.

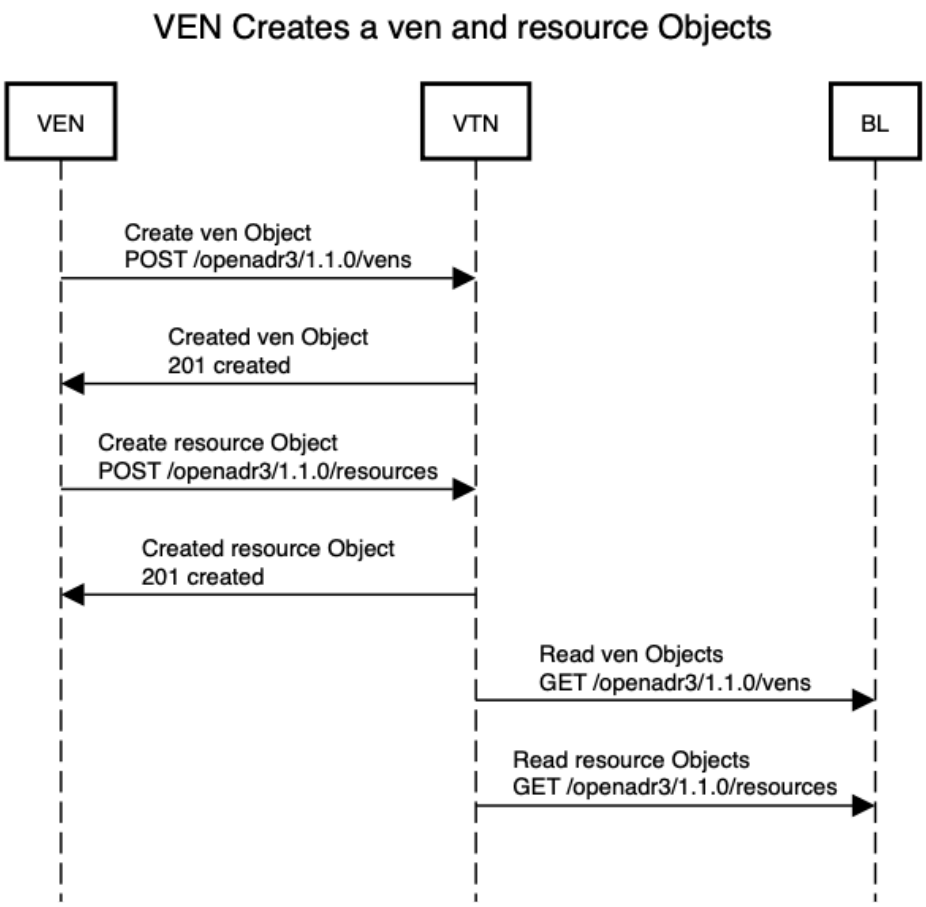



Figure 2: Sequence for submitting and retrieving ADVISOR information.

# Maintaining ADVISOR Records Over Time

ADVISOR exchange is a recurring data-maintenance process rather than a one-time submission. The VEN should update a resource object when a material characteristic changes, such as device count, capability, availability, location, timing, or communication status. The VEN may delete a resource object when the represented resource or cluster is no longer applicable. Before deleting a ven object, the VEN should delete or otherwise resolve its associated resource objects according to the VTN's supported behavior.

The example below uses PUT as a full replacement of the resource representation. Its shortened **attributes** array highlights two changed parameters only; in an actual request, the VEN should merge those changes into the complete current resource and include all unchanged attributes that must be retained. If a VTN supports partial updates, the parties should document the applicable method and behavior separately.

**Sample request by the VEN to replace an existing resource object:**

```
curl -X PUT -H "Content-Type: application/json" -H "Authorization:
Bearer <ven_access_token>" -d @advisor_oadr_resource_update.json
https://{VTN_BASE_URL}/OpenADR3/1.1.0/resources/{resourceID}
advisor_oadr_resource_update =
{
  "resourceName": "ADVISOR-HQNSB8-RESOURCE-00001",
  "venID": "ven-object-999",
  "attributes": [
    {
      "type": "ADVISOR_NUM_DEVICES_REPRESENTED",
      "values": [155]
    },
    {
      "type": "ADVISOR_MAX_GENUP_LOADDOWN",
      "values": [6500]
    },
      …
  ]
}
```

**Sample request by VEN to delete an existing ven object:**

```
curl -X DELETE -H "Authorization: Bearer <ven_access_token>"
https://{VTN_BASE_URL}/OpenADR3/1.1.0/vens/{venID}
```

# Appendix A

## Diagram script for https://sequencediagram.org/

```
title VEN Authentication

Service Provider->VEN: provision VTN address and
credentials\n(Out-Of-Band)

VEN->VTN:read Auth Service address\nGET /openadr3/1.1.0/auth/server

VTN->VEN: token url

VEN->Auth Service:read token\nPOST /openadr3/1.1.0/auth/token

Auth Service->VEN: token


title VEN Creates a ven and resource Objects

VEN->VTN: Create ven Object\nPOST /openadr3/1.1.0/vens

VTN->VEN: Created ven Object\n201 created
```

```
VEN->VTN: Create resource Object\nPOST /openadr3/1.1.0/resources

VTN->VEN: Created resource Object\n201 created

BL<-VTN: Read ven Objects\nGET /openadr3/1.1.0/vens

BL<-VTN: Read resource Objects\nGET /openadr3/1.1.0/resources
```

—------- End —-------