Considering extensions to enumeration tables using Schema files

Written by: David Herron - Evoke Systems - david@davidherron.com or david.herron@evokesystems.com

Date: February 1, 2025

Introduction

An important hallmark of the OpenADR 3.x protocol is flexibility. Rather than a rigid attitude of "THERE IS ONLY ONE TRUE WAY TO USE OPENADR", the realization is that many times demand/response programs are customized.

In Definition.md, the principle is described this way:

Extensibility

The OpenADR 3.0 protocol allows servers and clients to interoperate without custom integration. It is intended to provide a functional footprint that is sufficient to accommodate all common demand response use cases. However, some demand response program developers may find it useful to use content that cannot be expressed using the constructs of the specification, or could be expressed in a better form with an extension.

There are two extension mechanisms offered by OpenADR 3.0: model extensions, and private strings.

Expressing the Enumeration tables in JSON Schema format allows the OpenADR ecosystem an additional avenue of extension.

Namely, it becomes easy to define new enumerations by adding entries to the enumeration schemas. It's then possible to communicate those extensions among cooperating partners simply by sharing the updated JSON Schema files.

Thoughts on implementing extensions to enumeration schema's

The official OpenADR enumerations are listed in tables in Definition.md. Those tables are, in some cases, created directly from the enumeration schema files.

As a result we can think of the enumeration schema files as the concrete specification, and the text in Definition.md as the human-friendly form.

A given demand/response project might wish to extend some of the enumerated tables.

A very likely area of extensions is for custom report payloads. For example, a program related to electric vehicle charging may want VENs to report similar to what's described in the EV Charging Use Data Specification.

In other words, the program administrator could develop a private copy of `report-payloads.schema.yaml`. The VTN would request reports using those payload definitions, and the VEN would respond with matching data.

The customized schema file allows everyone to be on the same page, because the extensions are concretely described.

A VTN or VEN could dynamically generate the arrays of enumeration validation functions at runtime.