CVII-TS: Binary encoding for VSS-based data transfer

This is a planning / work breakdown page for driving the development of a well defined efficient data serialization format for VSS data.

Background:

- **VISS v2** has a comprehensive feature set, and therefore also **defines a JSON message transfer format** that is full featured.
- JSON is the default and popular way to encode data in many web (and other) environments. Whenever JSON is applicable then reusing the VISS definition is recommended.
- JSON is however text-based and somewhat verbose. Memory and bandwidth efficiency is important in certain situations.
- Unlike VISS, many other protocols do not define the payload format. These require reuse of a defined format.
    - Reuse VISS message definition in JSON
    - ...or a new data serialization format.
- VISS v2 development have occasionally also discussed adding an alternative or "compressed" format in addition to VISS. The analysis /development here might potentially influence also that -> leading to a VISS+binary encoding variation
- Either way, there is a clear driver towards developing a common standard, and supporting tools, for serializing VSS data.
- Since this is not a new problem, a definition is likely to leverage existing binary serialization formats and simply document the message definitions that are applicable for data following the VSS data model.
- Additional rationale/background could be found in: Data serialization / value formats

Initial design thoughts:

- To complement the JSON approach, this is aiming for a highly space efficient (binary) representation of data.
- It should likely reuse some framework to define the serialization. **Protobuf** and **AVRO** are two popular choices:
    - GRPC is a popular choice of RPC framework, and that may influence in the direction of using **protobuf** also for data
    - **Protobuf** is used in SENSORIS definitions
    - **AVRO** is used in eSync Alliance and therefore a natural choice for our liaison / collaboration with eSync Alliance.
    - **AVRO** is part of Apache technology stack, and thus fits well into Kafka, Spark, NiFi... that are also popular data processing technologies.

⭐ Proposal: We should develop VSS data message descriptions in both **Protobuf** and **AVRO**.

Please note the difference between:

1. Describing a *VSS-formatted catalog of data* in a new format, e.g. VSS(YAML)JSON conversion that exists in vss-tools, or the VSS(YAML) Protobuf converter.
2. Describing the *payload format to transfer values* that have been measured, for VSS-defined data items.

We are here talking about 2), not 1).

VSS has a limited set of data types (numbers represented as integers of different bit size, floating point types, strings, etc.) so the serialization format of the data itself is likely trivial. But since the scope of this work is to define useful *payloads* in protocol transfer, it includes more information than only the data value itself. Prominently, of course, value carrying messages almost always have a time stamp. Agreeing on and implementing the message format is required, even if leveraging encoding technologies built into protobuf, AVRO, and similar.

A full background and the suggested definition of several message types is here: Data serialization / value formats (please provide input/improvements there, so that it can be the blueprint for the development described on this page.)