Getting the most out of AVEVA's Industrial Platform

Platform Developer Deep Dive

Collin Bardini
Konstantin Chudnovskiy
EV Car Chargers
• Code demos available on GitHub
• https://github.com/osisoft/demos-aveva_world_2023
Setting Up Data Infrastructure
Can I use an Adapter or PI to Data Hub?

Adapters

- Standard products for sending data to AVEVA Data Hub
- Connect to the following data sources:
  - MQTT (Sparkplug B or generic)
  - OPC UA
  - Azure Event Hubs
  - BACnet
  - DNP3
  - Modbus TCP
  - RDBMS
  - Structured Data Files (csv, json, or XML)

PI To Data Hub

- Forms a native connection to an on-premises PI Server
- Can replicate AF Elements as Assets
- Standard product
OMF or SDS writes?

Data collection: OMF vs. SDS data writes

When collecting data programatically, you have the option of using either the Open Message Format (OMF) or Sequential Data Store (SDS) data writes. OMF is typically the best option for simple and efficient data collection. However, each option has its pros and cons, and this topic highlights the key differences.

Data collection option comparison

The following table lists the factors that you should consider when choosing between OMF and SDS, as well as which option is the better for each factor. Some factors (Security and Performance) have no inherently superior option and require additional consideration. For more information on these factors, see Situation dependent factors.

<table>
<thead>
<tr>
<th>Factor</th>
<th>OMF</th>
<th>SDS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cross-platform</td>
<td>✔</td>
<td></td>
</tr>
<tr>
<td>Ease-of-use</td>
<td>✔</td>
<td></td>
</tr>
<tr>
<td>Flexibility</td>
<td></td>
<td>✔</td>
</tr>
<tr>
<td>Security</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Performance</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

Performance best practices for OMF

Critical

• Avoid out of order (ooo) data

Very Important

• Maximize content per stream in a message
• Maximize Content per OMF message (192KB)
• Use gzip compression
• You can send data in parallel across steams (still need to account for ooo data)

Nice to Have

• Exclude default values
• Avoid duplicate entries for streams/containers
msg_body = gzip.compress(bytes(json.dumps(omf_message_json), 'utf-8'))
headers = self.getHeaders(message_type, action)

return self.request('POST',
                     self.OMFEndpoint,
                     headers=headers,
                     data=msg_body,
                     verify=self.VerifySSL,
                     timeout=600)

if (response.status_code == 504 or response.status_code == 503):
    if (failures >= 0 and failures >= max_retries):
        logging.error('Server error. No more retries available. ')
        return response
    else:
        timeout = 3600 if failures >= 12 else 2 ** failures
        logging.warning('Server error. Retrying...')
        time.sleep(timeout)
        failures += 1
else:
    success = True
# Get or create types in data hub
response = omf_client.omfRequest(
    OMFMessageType.Type, OMFMessageAction.Create,
    charger_data_client.getTypes())
omf_client.verifySuccessfulResponse(response, 'Error creating types')

# Get or create streams from data source in data hub
streams = charger_data_client.getStreams()
response = omf_client.omfRequest(
    OMFMessageType.Container, OMFMessageAction.Create, streams)
omf_client.verifySuccessfulResponse(response, 'Error creating containers')
event: ChargerData = queue.pop()
if event.Id not in payload:
    payload[event.Id] = [
        {'Timestamp': event.Timestamp, 'Value': event.Value}]
else:
    payload[event.Id].append(
        {'Timestamp': event.Timestamp, 'Value': event.Value})
process_partitions = []
for i in range(0, len(streams), int(len(streams)/max_processes)):
    process_partitions.append(streams[i:i + int(len(streams)/max_processes)])

processes = []
for process_partition in process_partitions:
    process = Process(target=processManager, args=(appsettings, process_partition))
    processes.append(process)
    process.start()
AVEVA Data Hub

Streams

Asset Rules

Assets

EV Charger Data

Ingress App
Complex Types vs. Simple Types with Assets

Complex Stream
Typeld: ComplexType
Name: Timestamp SdsType: DateTime
Name: Temperature SdsType: Double
Name: Pressure SdsType: Double
Name: Status SdsType: String

Temperature Stream
Typeld: DoubleType
Name: Timestamp SdsType: DateTime
Name: Value SdsType: Double

Status Stream
Typeld: StringType
Name: Timestamp SdsType: DateTime
Name: Value SdsType: String

Asset
Id: <GUID>
Name: Asset
Name: Temperature StreamRef: Temperature Stream
Name: Pressure StreamRef: Pressure Stream
Name: Status StreamRef: Status Stream
Car Charger Asset

Id: ChargingStation_0061_01_02_04_T1
Name: Charging Station T1
Metadata:
- Stream Name: 0061-01-02-04-T1-Tesla-PilotSignalAmps
- EVSE Type: Tesla
- Site: 01
- Site Account: 0061

Properties:
- Name: Pilot Signal Amps
  - StreamId: 0061-01-02-04-T1-Tesla-PilotSignalAmps
- Name: Average Kwh Delivered
  - StreamId: 0061-01-02-04-T1-Tesla-AvgKwhDelivered
- Name: Status
  - StreamId: 0061-01-02-04-T1-Tesla-Status
## Asset Rules

### Name: SLTC-EV-Chargers-2  Asset Type: Charging Station

<table>
<thead>
<tr>
<th>Id</th>
<th>Type</th>
<th>Name</th>
<th>Description</th>
<th>Site Accou</th>
<th>Site</th>
<th>EVSE Type</th>
<th>Started Size</th>
<th>Total Site Capacity</th>
<th>Total Cost</th>
<th>Total Return</th>
<th>Total Micro</th>
<th>Total High</th>
<th>Total Low</th>
<th>Total Duration</th>
<th>Avg Session Duration Min</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Charging Station</td>
<td>Charging Station C06</td>
<td></td>
<td>Westlake</td>
<td>520 S El Ca...</td>
<td>Tesla</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
</tr>
<tr>
<td></td>
<td>Charging Station</td>
<td>Charging Station C07</td>
<td></td>
<td>Westlake</td>
<td>520 S El Ca...</td>
<td>Webasto DX</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
</tr>
<tr>
<td></td>
<td>Charging Station</td>
<td>Charging Station C08</td>
<td></td>
<td>Westlake</td>
<td>520 S El Ca...</td>
<td>Tesla</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
</tr>
<tr>
<td></td>
<td>Charging Station</td>
<td>Charging Station C09</td>
<td></td>
<td>Westlake</td>
<td>520 S El Ca...</td>
<td>Tesla</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
</tr>
<tr>
<td></td>
<td>Charging Station</td>
<td>Charging Station C10</td>
<td></td>
<td>Westlake</td>
<td>520 S El Ca...</td>
<td>Tesla</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
</tr>
<tr>
<td></td>
<td>Charging Station</td>
<td>Charging Station C11</td>
<td></td>
<td>Westlake</td>
<td>520 S El Ca...</td>
<td>Tesla</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
</tr>
<tr>
<td></td>
<td>Charging Station</td>
<td>Charging Station C12</td>
<td></td>
<td>Westlake</td>
<td>520 S El Ca...</td>
<td>Tesla</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
</tr>
<tr>
<td></td>
<td>Charging Station</td>
<td>Charging Station C13</td>
<td></td>
<td>Westlake</td>
<td>520 S El Ca...</td>
<td>Webasto DX</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
</tr>
<tr>
<td></td>
<td>Charging Station</td>
<td>Charging Station C14</td>
<td></td>
<td>Westlake</td>
<td>520 S El Ca...</td>
<td>Webasto DX</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
</tr>
<tr>
<td></td>
<td>Charging Station</td>
<td>Charging Station C15</td>
<td></td>
<td>Westlake</td>
<td>520 S El Ca...</td>
<td>Tesla</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
</tr>
<tr>
<td></td>
<td>Charging Station</td>
<td>Charging Station C16</td>
<td></td>
<td>Westlake</td>
<td>520 S El Ca...</td>
<td>Tesla</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
</tr>
<tr>
<td></td>
<td>Charging Station</td>
<td>Charging Station C17</td>
<td></td>
<td>Westlake</td>
<td>520 S El Ca...</td>
<td>Tesla</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
</tr>
<tr>
<td></td>
<td>Charging Station</td>
<td>Charging Station C18</td>
<td></td>
<td>Westlake</td>
<td>520 S El Ca...</td>
<td>Webasto DX</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
</tr>
<tr>
<td></td>
<td>Charging Station</td>
<td>Charging Station C19</td>
<td></td>
<td>Westlake</td>
<td>520 S El Ca...</td>
<td>Webasto DX</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
</tr>
<tr>
<td></td>
<td>Charging Station</td>
<td>Charging Station C20</td>
<td></td>
<td>Westlake</td>
<td>520 S El Ca...</td>
<td>Webasto DX</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
</tr>
<tr>
<td></td>
<td>Charging Station</td>
<td>Charging Station C21</td>
<td></td>
<td>Westlake</td>
<td>520 S El Ca...</td>
<td>Webasto DX</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
<td>061-02-02...</td>
</tr>
</tbody>
</table>

### Statistics
- Assets Processed: 72
- Streams Processed: 44417
- Streams Matched: 1296
- Execution Errors: 0
- Metadata: 216
- Stream References: 1296

### Progress
- Status: Complete
- Percent Done: 100%
Creating a Custom Visualization
Streamlit

• A fast way to build/deploy web applications using Python
• Comes with built in components
• The community creates and shares custom components
• Useful for creating data visualizations/dashboards
resource = st.secrets['resource']
api_version = st.secrets['api_version']
tenant_id = st.secrets['tenant_id']
namespace_id = st.secrets['namespace_id']

# Read client from session state or create one if it does not already exist
if 'client' in st.session_state:
    client = st.session_state['client']
else:
    client_id = st.secrets['client_id']
    client = ADHClient(api_version, tenant_id, resource, client_id, accept_verbosity=True)
    st.session_state['client'] = client
# Set metrics data
charger_data = client.Assets.getAssetLastData(
    namespace_id,
    selected_charger.Id
).Results

...  

# Set trend data
raw_data = client.Assets.getAssetSampledData(namespace_id, selected_charger.Id, start_index=(
    datetime.utcnow() - timedelta(hours=2)).isoformat(),
end_index=datetime.utcnow().isoformat(), intervals=350).Results
amps_data_frame = pd.DataFrame.from_dict(raw_data["PilotSignalAmps"])
amps_data_frame['Timestamp'] = pd.to_datetime(amps_data_frame['Timestamp'])
Working with Partner to Perform Calculation
### AVEVA Data Hub > Communities

#### EV Sustainability Calculations

- **ID**: 74d13401-f98e-4c77-9c0e-9ee092bd68a2
- **All Tenants Active**
- **Created**: Sep 14, 2023

<table>
<thead>
<tr>
<th>Tenants</th>
<th>My Members</th>
<th>My Groups</th>
<th>My Clients</th>
<th>Usage</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Filter tenants</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

#### Administrative Tenant

<table>
<thead>
<tr>
<th>Name</th>
<th>Status</th>
<th>Users (43)</th>
<th>Clients (1)</th>
<th>Streams (1369)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Americas-A0x-LH1</td>
<td>Active</td>
<td>34</td>
<td>0</td>
<td>1297</td>
</tr>
<tr>
<td>DataHubProductionSandbox1</td>
<td>Active</td>
<td>9</td>
<td>1</td>
<td>72</td>
</tr>
</tbody>
</table>
public class SdsStream
{
    public string Id { get; set; }
    public string Name { get; set; }
    public string Description { get; set; }
    public string TypeId { get; set; }
    public IList<SdsStreamIndex> Indexes { get; set; }
    public SdsInterpolationMode? InterpolationMode { get; set; }
    public SdsExtrapolationMode? ExtrapolationMode { get; set; }
    public IList<SdsStreamPropertyOverride> PropertyOverrides { get; set; }
}
private static readonly string TENANT_ID
    = Environment.GetEnvironmentVariable("ADH_PROD_TENANT_ID");
private static readonly string NAMESPACE_ID
    = Environment.GetEnvironmentVariable("ADH_PROD_NAMESPACE_ID");
private static readonly string COMMUNITY_ID
    = Environment.GetEnvironmentVariable("ADH_PROD_COMMUNITY_ID");
private static readonly string CLIENT_ID
    = Environment.GetEnvironmentVariable("ADH_PROD_CLIENT_ID");
private static readonly string CLIENT_SECRET
    = Environment.GetEnvironmentVariable("ADH_PROD_CLIENT_SECRET");
double energy = (chargerAmps.ElementAt(i + 1).Value +
chargerAmps.ElementAt(i).Value) / 2 *
(chargerAmps.ElementAt(i + 1).Timestamp -
chargerAmps.ElementAt(i).Timestamp).TotalSeconds *
CHARGE_VOLTAGE * WATTSECONDS_TO_MEGAWATTHOURS; // MWh
double emissions = energy * totalEmissions.ElementAt(i).Value /
demand.ElementAt(i).Value * MEGATONS_TO_KG; // Kg
double savedEmissions = energy * ENERGY_TO_SAVED_EMISSIONS;
// Kg
double netSavedEmissions = savedEmissions - emissions;
totalizedSavedEmissions += netSavedEmissions;
// Retrieve charger amperage data
using (HttpRequestMessage request = new (HttpMethod.Get, new Uri($"{stream(Self)}/Data/Interpolated?startIndex={sliceStartIndex.ToString(T_FORMAT)}&endIndex={sliceEndIndex.ToString(T_FORMAT)}&count={count}")))
{
    request.Headers.Add("Community-Id", COMMUNITY_ID);
    response = await httpClient.SendAsync(request).ConfigureAwait(false);
}
// Retrieve electricity supply data
default = await httpClient.GetAsync(
    new Uri($"api/v1/Tenants/{TENANT_ID}/Namespaces/{NAMESPACE_ID}/Streams/CAISO-
    TotalDemand/Data/Interpolated?startIndex={sliceStartIndex.ToString(T_FORMAT)}&endIndex={
    sliceEndIndex.ToString(T_FORMAT)}&count={count}'', UriKind.Relative));

CheckIfResponseWasSuccessful(response);

IList<TimeIndexedDouble> demand =
    JsonConvert.DeserializeObject<IList<TimeIndexedDouble>>(await 
    response.Content.ReadAsStringAsync());
response = await httpClient.PutAsync(
    new
    Uri($"api/v1/Tenants/\{TENANT_ID\}/Namespaces/\{NAMESPACE_ID\}/Streams/\{outputStreamId\}/Data ", UriKind.Relative),
    content1)
    .ConfigureAwait(false);
CheckIfResponseWasSuccessful(response);
Collin Bardini
Senior Software Developer - Product Readiness
• AVEVA
• collin.bardini@aveva.com

Konstantin Chudnovskiy
Software Architect
• AVEVA
• konstantin.chudnovskiy@aveva.com
This presentation may include predictions, estimates, intentions, beliefs and other statements that are or may be construed as being forward-looking. While these forward-looking statements represent our current judgment on what the future holds, they are subject to risks and uncertainties that could result in actual outcomes differing materially from those projected in these statements. No statement contained herein constitutes a commitment by AVEVA to perform any particular action or to deliver any particular product or product features. Readers are cautioned not to place undue reliance on these forward-looking statements, which reflect our opinions only as of the date of this presentation.

The Company shall not be obliged to disclose any revision to these forward-looking statements to reflect events or circumstances occurring after the date on which they are made or to reflect the occurrence of future events.
ABOUT AVEVA

AVEVA is a world leader in industrial software, providing engineering and operational solutions across multiple industries, including oil and gas, chemical, pharmaceutical, power and utilities, marine, renewables, and food and beverage. Our agnostic and open architecture helps organizations design, build, operate, maintain and optimize the complete lifecycle of complex industrial assets, from production plants and offshore platforms to manufactured consumer goods.

Over 20,000 enterprises in over 100 countries rely on AVEVA to help them deliver life’s essentials: safe and reliable energy, food, medicines, infrastructure and more. By connecting people with trusted information and AI-enriched insights, AVEVA enables teams to engineer efficiently and optimize operations, driving growth and sustainability.

Named as one of the world’s most innovative companies, AVEVA supports customers with open solutions and the expertise of more than 6,400 employees, 5,000 partners and 5,700 certified developers. The company is headquartered in Cambridge, UK.

Learn more at www.aveva.com

© 2023 AVEVA Group plc and its subsidiaries. All rights reserved.