

# Test- and Measurement Data as First-Class ML Citizens

Mixed Reality – Bridging the Real and Digital Worlds

Explore to Innovate 2025 - Benningen – July 3<sup>rd</sup> 2025  
Institute for Driver Assistance and Connected Mobility (IFM)

Author

Stefan Romainczyk  
Senior Product Manager

[LinkedIn](#)  
[www.peak-solution.de](http://www.peak-solution.de)

# Test- and Measurement Data as First-Class ML Citizens

## Abstract

Test- and measurement data is acquired during the development process of every product from simple toothbrushes up to complex machines and vehicles. Often, this data is captured in various formats such as XLSX, CSV, and MDF, making it challenging to add value, especially for further analysis and machine learning (ML).

In this talk, we will present how ASAM ODS can help integrate different measurement data files into a holistic data view. Lightweight ETL pipelines based on microservices extract the existing data and enrich it with additional metadata during the import process. The imported data files can remain in their original location and format, preserving existing tool chains.

Once the data arrives in the ASAM ODS server, the existing HTTP(s)-REST-API provides secure data access from any location. However, in order to integrate this data with existing analysis and ML tools, another building block is needed to expose the data to the lingua franca of data scientists and data analysts – Python.

This gap is filled by the open-source ASAM ODSBox. This lightweight Python wrapper on top of the ASAM ODS HTTP-API introduces a simple JSON query language that makes it easy to learn and use ASAM ODS data. Additionally, the measurement data is returned as pandas.DataFrames, which naturally integrate with existing Python analytics and ML libraries.

This solution offers even more benefits. With support for the Python language, the data can now be easily used in interactive notebooks such as Jupyter or data analysis tools like Microsoft Power BI or MATLAB®.

# Test- and Measurement Data as First-Class ML Citizens

## Abstract

- **TEST- AND MEASUREMENT DATA PROBLEM:**

Test- and measurement data is acquired during the development process of every product from simple toothbrushes up to complex machines and vehicles. Often, this **data is captured in various formats such as XLSX, CSV, and MDF, making it challenging to add value, especially for further analysis and machine learning (ML).**

- **SOLUTION:**

In this talk, we will present how **ASAM ODS** can help integrate different measurement data files into **a holistic data view.**

- **PATH TO SOLUTION:**

Lightweight ETL pipelines based on microservices extract the existing data and enrich it with additional metadata during the import process. The imported data files can remain in their original location and format, preserving existing tool chains.

Once the data arrives in the ASAM ODS server, the existing HTTP(s)-REST-API provides secure data access from any location. However, in order to integrate this data with existing analysis and ML tools, another building block is needed to expose the data to the lingua franca of data scientists and data analysts – Python.

This gap is filled by the open-source ASAM ODSBox. This lightweight Python wrapper on top of the ASAM ODS HTTP-API introduces a simple JSON query language that makes it easy to learn and use ASAM ODS data. Additionally, the measurement data is returned as pandas.DataFrames, which naturally integrate with existing Python analytics and ML libraries.

This solution offers even more benefits. With support for the Python language, the data can now be easily used in interactive notebooks such as Jupyter or data analysis tools like Microsoft Power BI or MATLAB®.

# Test- and Measurement Data Problem

# Test- and Measurement Data Problem

## Problem Analysis

### Data Format Diversity (Variety, Velocity, Volume)

- Data comes in dozens of formats
- Structured, Semi-structured, Unstructured
- 10–30 different formats across systems

### Data Quality (Veracity)

- Not cleansed
- Inaccurate
- Inconsistently formatted
- Missing Meta Data

### Data Accessibility (Value)

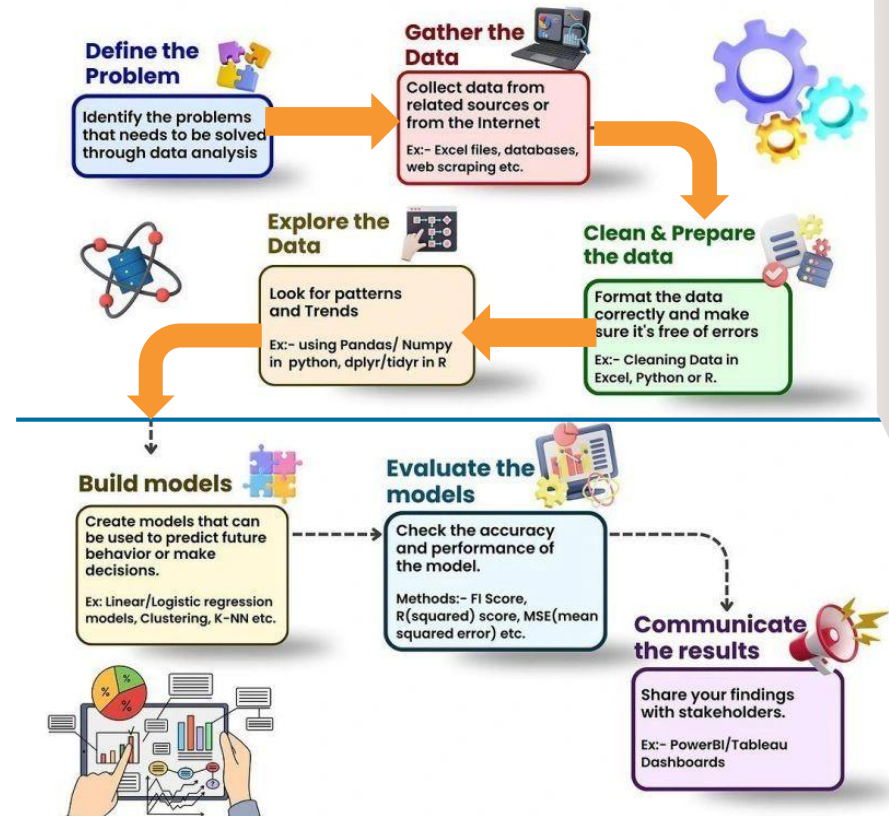
- Data silos
- IT boundaries
- Missing/proprietary interfaces

### Data Analysis Rate

- 5% to 20% of collected test and measurement data is typically analyzed (1)
- "less than 0.5% of all data is ever analyzed and used" (2)

# Test- and Measurement Data Problem

## Process of Data Analysis



Curated by tutort<sup>®</sup> academy

(1)

Your Data is here...

ML starts here...

Data Format Diversity

Data Accessibility

Data Quality

# Path to Solution

# Extract Data from Different Data Formats

## Challenges

Data Format  
Diversity

- Data captured in different formats (different tool vendors)
  - Test- and measurement data mostly stored in files
  - Access to proprietary data by libraries in different programming languages
  - CSV is the biggest PITA
- Existing toolchains limits flexibility
  - Prevent data movement or conversion
  - Breaking existing toolchains introduce additional costs
- Data volume prohibits data copies
  - “Doubles” the needed storage space
  - Conversion may be needed for performance reasons



# Extract Data from Different Data Formats

Solution

ODS Data  
Plugins



## ASAM ODS (External) Data Plugins

- Lean micro services (Google Protobuf)
- GetMetaData (channel name, description, units)
- GetChannelData (bulk data)
- Support for “any” programming language
- Small R&D Investment
- IP-Protection possible

```
def data(self) -> pd.DataFrame:
    """
    Read the data from the file and return it as a pandas DataFrame.
    :return: DataFrame containing the data from the file.
    """
    if self._df is None:
        self.log.info("Reading file: %s", self.__file_path)

        # The first three rows are containing the metadata
        with open(self.__file_path, 'r', encoding='utf-8-sig') as file:
            self.__names= [item.strip().strip('\"') for item in file.readline().split(';')]
            self.__descriptions= [item.strip().strip('\"') for item in file.readline().split(';')]
            self.__units= [item.strip().strip('\"') for item in file.readline().split(';')]

        # Read the data from the file - skip the header (already read)
        df = pd.read_csv(self.__file_path, sep=';', decimal=',', header=None, skiprows=3)
        df[2] = pd.to_datetime(df[2] + 'T' + df[3] + '.' + df[4].astype(str))

        self._df = df

    return self._df
```



ODS Data  
Plugins



Index	Act_Power	d	Impuls	Lamp_ON	light_sensor	m	s	t	Temp_Lamp	u
0	1.21293	2025-01-03T00:00:40Z	0	0	3.05185	0	2984700	00:00:40	32.3295	1735858840000000
1	1.21318	2025-01-03T00:01:40Z	0	0	3.05185	0	2984760	00:01:40	34.0549	1735858900000000
2	73.275	2025-01-03T00:02:40Z	0	1	10.3763	0	2984820	00:02:40	32.0057	1735858960000000
3	1.21297	2025-01-03T00:03:40Z	0	0	3.05185	0	2984880	00:03:40	33.3794	1735859020000000
4	73.0473	2025-01-03T00:04:40Z	0	1	9.76592	0	2984940	00:04:40	33.6214	1735859080000000
5	1.21292	2025-01-03T00:05:40Z	0	0	3.05185	0	2985000	00:05:40	32.7849	1735859140000000
6	1.21332	2025-01-03T00:06:40Z	0	0	3.05185	0	2985060	00:06:40	34.5491	1735859200000000
7	1.21291	2025-01-03T00:07:40Z	0	0	3.05185	0	2985120	00:07:40	32.2119	1735859260000000
8	1.21313	2025-01-03T00:08:40Z	0	0	3.05185	0	2985180	00:08:40	33.9687	1735859320000000
9	73.6874	2025-01-03T00:09:40Z	0	1	10.3763	0	2985240	00:09:40	32.1913	1735859380000000
10	1.21292	2025-01-03T00:10:40Z	0	0	3.05185	0	2985300	00:10:40	32.2936	1735859440000000
11	72.9054	2025-01-03T00:11:40Z	1	0	10.3763	0	2985360	00:11:40	34.0349	1735859500000000
12	1.21289	2025-01-03T00:12:40Z	0	0	2.44148	0	2985420	00:12:40	32.6154	1735859560000000
13	1.21322	2025-01-03T00:13:40Z	0	0	3.05185	0	2985480	00:13:40	34.3837	1735859620000000
14	1.21287	2025-01-03T00:14:40Z	0	0	3.05185	0	2985540	00:14:40	32.0316	1735859680000000
15	1.213	2025-01-03T00:15:40Z	0	0	3.05185	0	2985600	00:15:40	33.7423	1735859740000000
16	73.3348	2025-01-03T00:16:40Z	0	1	10.3763	0	2985660	00:16:40	32.71	1735859800000000
17	1.2129	2025-01-03T00:17:40Z	0	0	3.05185	0	2985720	00:17:40	33.1066	1735859860000000
18	1.21314	2025-01-03T00:18:40Z	0	0	3.05185	0	2985780	00:18:40	34.4926	1735859920000000
19	1.21304	2025-01-03T00:19:40Z	0	0	3.05185	0	2985840	00:19:40	33.4146	1735859980000000

# Extract Data from Different Data Formats

## Advantages

ODS Data  
Plugins



- **Data Format Homogenization using a Standardized API**
  - All file formats look similar through the lens of the API
- **No data conversion or data movement necessary!**
  - Existing toolchains stay intact
  - No data duplication needed
- **Small R&D Investment**
  - Google Protobuf-Technology supports “any” Programming Language
  - Fast Implementation due to Code Generation and existing Examples

Examples: [https://peak-solution.github.io/data\\_management\\_learning\\_path/exd\\_api/overview.html](https://peak-solution.github.io/data_management_learning_path/exd_api/overview.html)

# Format and Store the Data Correctly

## Challenges

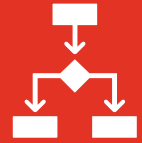
Data Quality

- Not Cleansed or Inaccurate Data
  - Missing or Miss-spelled Data
  - Wrong data values
- Missing Meta Data and Data Context
  - Limited organizational and searching functionality
  - Limited analytic functionality (think Power BI, ML, ...)
- Inconsistently formatted Data
  - ✓ ASAM ODS (External) Data Plugins

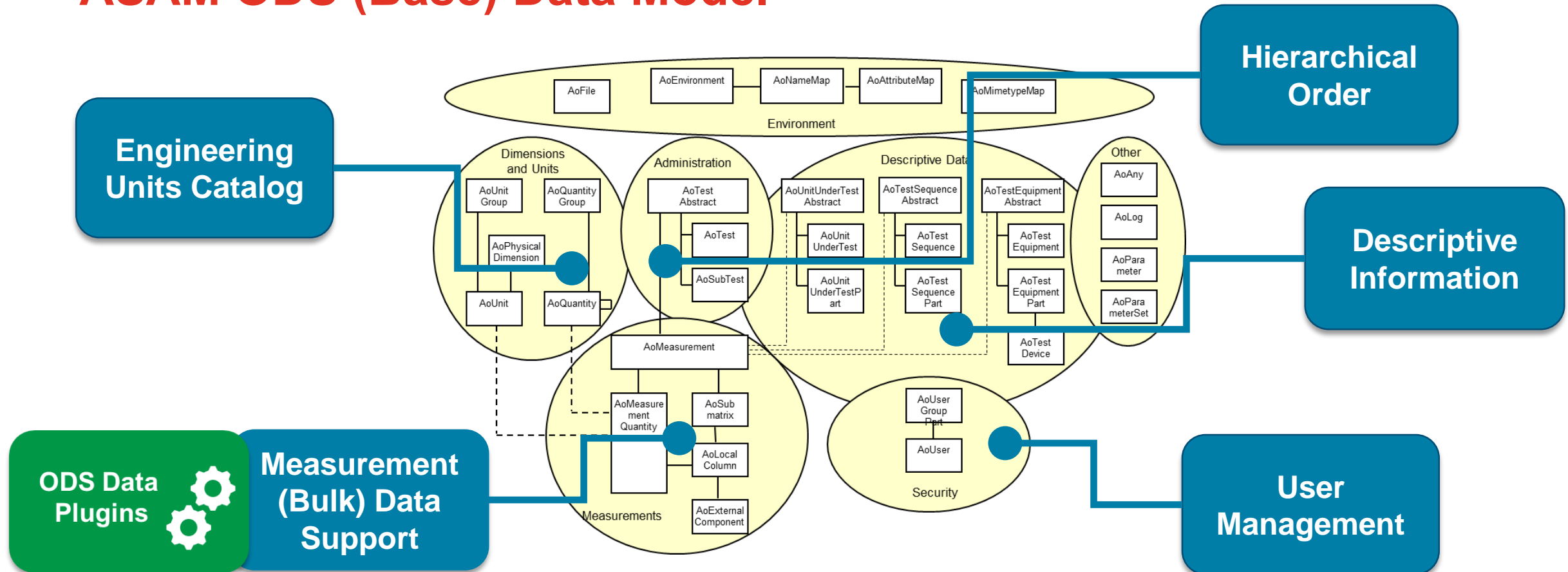
# Format and Store the Data Correctly

Solution

ODS Data  
Model



## ASAM ODS (Base) Data Model



# Format and Store the Data Correctly

## Advantages

ODS Data  
Model



- **Detection of Not Cleansed or Inaccurate Data**
  - Data Catalogs to identify Missing or Miss-spelled Data
  - Limits, NAN and NULL to mark wrong data values
- **Extended Meta Data and Data Context**
  - Navigation Hierarchy to support Data Organization
  - Descriptive Data for improved Analysis and Searching Capabilities
- **Machine Learning Advantages**
  - Base Data Model for **Data Semantic**
  - Base Entities and Relations define a **Measurement Data Ontology**

# Provide Suitable Data Access

## Challenges

## Data Accessibility



Data Scientist

### Explore the Data



Look for patterns and Trends

Ex:- using Pandas/ Numpy in python, dplyr/tidyr in R

## Job

- Exploratory Data Analysis (EDA)
- Processing, Cleaning and Verifying the Integrity of data.
- Identify trends in data and make unique predictions.
- Develop an understanding of using Machine Learning Techniques.

## Skills

- Understanding of Machine Learning Algorithm and Techniques.
- Hands-on Data Visualization tools such as **Tableau** and **Power BI**.
- Experience in Big data tools like **Spark** and **Hadoop**.
- Understanding of **Python** or **R** and Expert in **SQL**.

  **Data Source must support connectivity to the highlighted tools**

# Provide Suitable Data Access

Solution

ASAM  
ODSBox



## Analyze

- Python
- DataFrames
- Standardized
- AI Connectivity



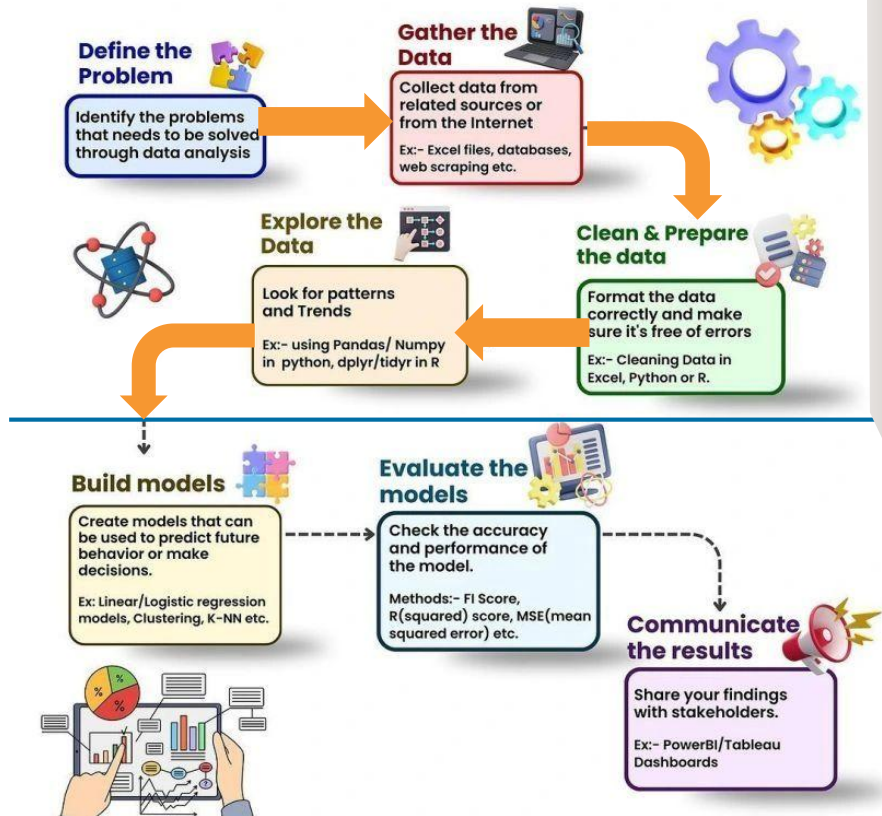
## Data Access

## ASAM ODSBox

- Access ASAM ODS server using Python
- Light-weight wrapper on HTTP/Protobuf
- Provide data as pandas.DataFrames
- Simple Query Language (mongoDB - like)
- Pip-install
- Open Source

# Test- and Measurement Data as First-Class ML Citizens

## Process of Data Analysis

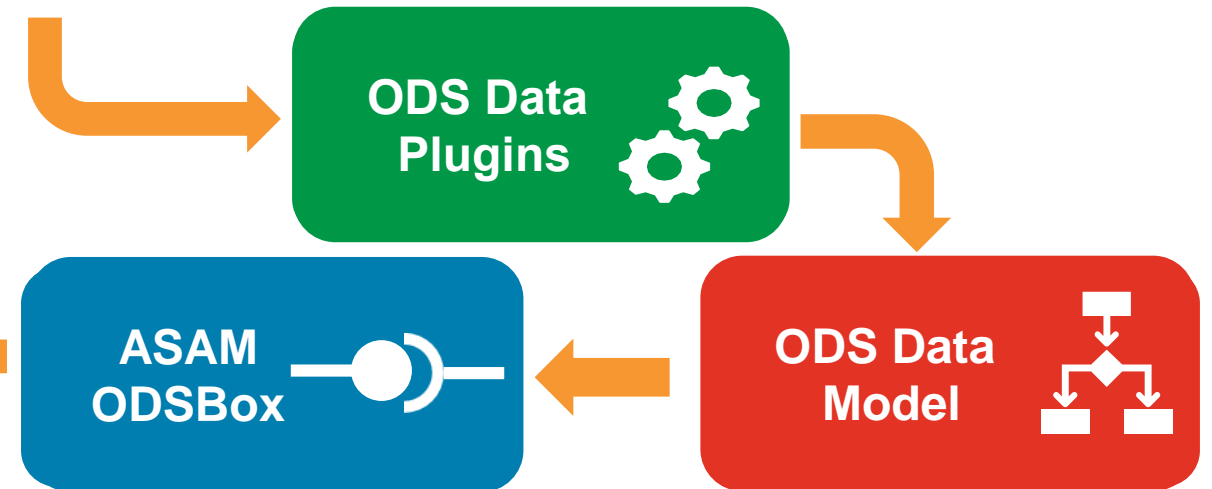


Curated by tutort<sup>®</sup> academy

(1)

Your Data  
is here...

ML starts here...



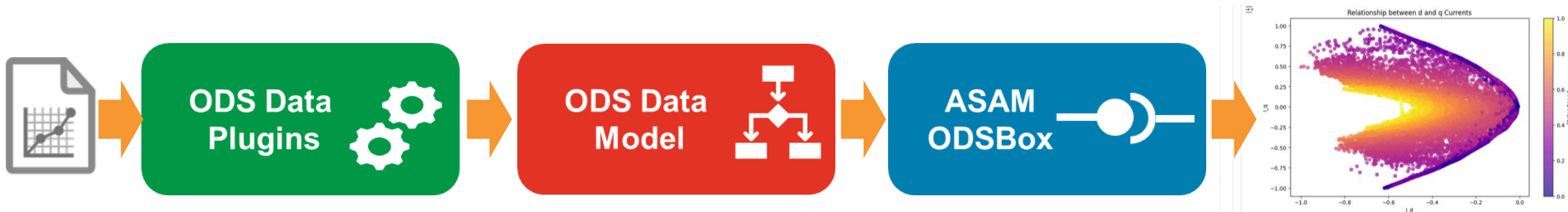


# Solution - Summary

# Summary

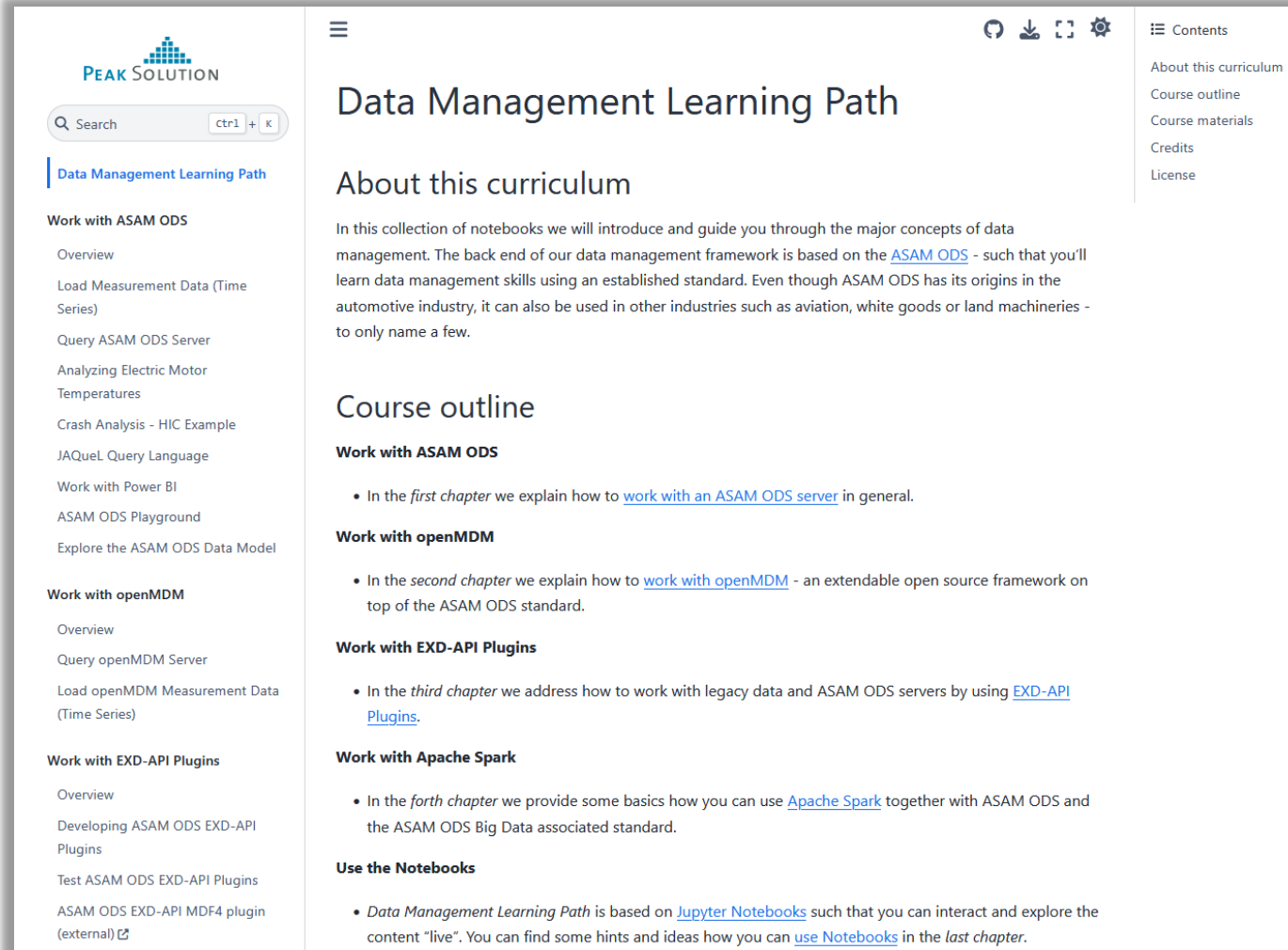
## Advantages of the ASAM ODS Standard

- The combination of **ASAM ODS DataPlugins**, with the **ASAM ODS (Base) Data Model** and the **ASAM ODSBox** provide the needed functionalities and capabilities making **Test- and Measurement Data a First Class ML Citizen**.
- Typical **Machine Learning Tools** can now be used by **Data Scientists** for “any data”.
- Furthermore, the introduced tool chain bridges the gap to Microsoft **Copilot** or Google **Gemini** or other **AI Assistants** to create solutions faster and more efficient – even for **non-Data Scientists**.
- The **ASAM ODS Standard** helps integrating different measurement data files into **a holistic data view**.



# Where to start?

Interactively explore ASAM ODS Solutions

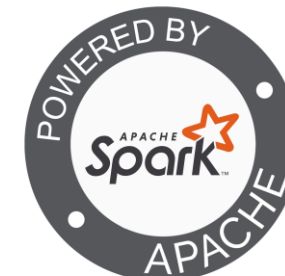


The screenshot shows the 'Data Management Learning Path' page. The left sidebar contains a search bar and a list of topics under three main sections: 'Work with ASAM ODS', 'Work with openMDM', and 'Work with EXD-API Plugins'. The main content area is titled 'Data Management Learning Path' and includes a 'Contents' sidebar on the right. The main text describes the curriculum, which is organized into chapters. The 'About this curriculum' section states that the notebooks introduce major concepts of data management, based on the ASAM ODS standard. The 'Course outline' section lists the following topics:

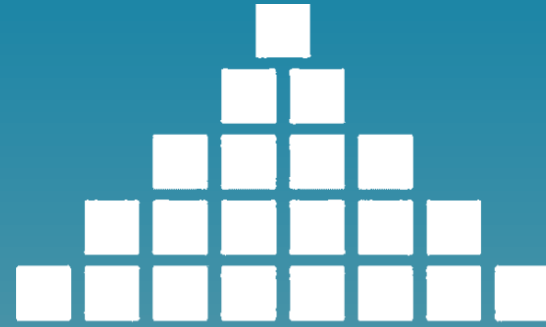
- Work with ASAM ODS**
  - In the *first chapter* we explain how to [work with an ASAM ODS server](#) in general.
- Work with openMDM**
  - In the *second chapter* we explain how to [work with openMDM](#) - an extendable open source framework on top of the ASAM ODS standard.
- Work with EXD-API Plugins**
  - In the *third chapter* we address how to work with legacy data and ASAM ODS servers by using [EXD-API Plugins](#).
- Work with Apache Spark**
  - In the *forth chapter* we provide some basics how you can use [Apache Spark](#) together with ASAM ODS and the ASAM ODS Big Data associated standard.
- Use the Notebooks**
  - Data Management Learning Path* is based on [Jupyter Notebooks](#) such that you can interact and explore the content "live". You can find some hints and ideas how you can [use Notebooks](#) in the *last chapter*.

## Open Source Training Material

- Jupyter Notebooks
- Python API Examples
- ExD Plugin Examples
- Microsoft Power BI
- Apache Spark
- Github Codespaces
- Google Colab



Data Management Learning Path: [https://peak-solution.github.io/data\\_management\\_learning\\_path/index.html](https://peak-solution.github.io/data_management_learning_path/index.html)



# PEAK SOLUTION

Test Processes - Test Data - Digital Identities

[www.peak-solution.de](http://www.peak-solution.de)

**Thank you for your attention!**