

デモセッション

テストのオーダが出されました。

MCツールからECUデータと計測データがMDF4フォーマットで生み出されて、それぞれのデータはODSに保存されます。

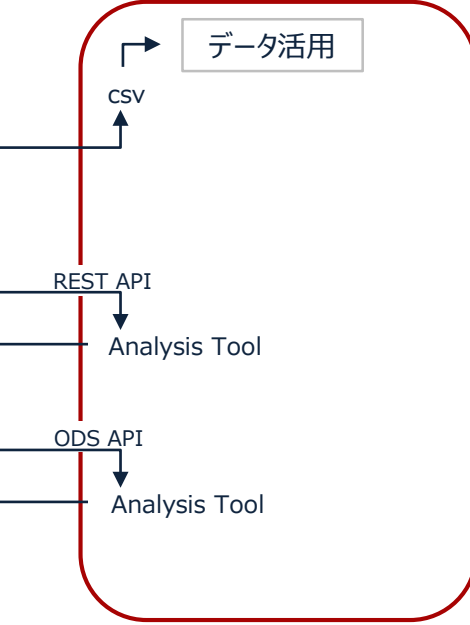
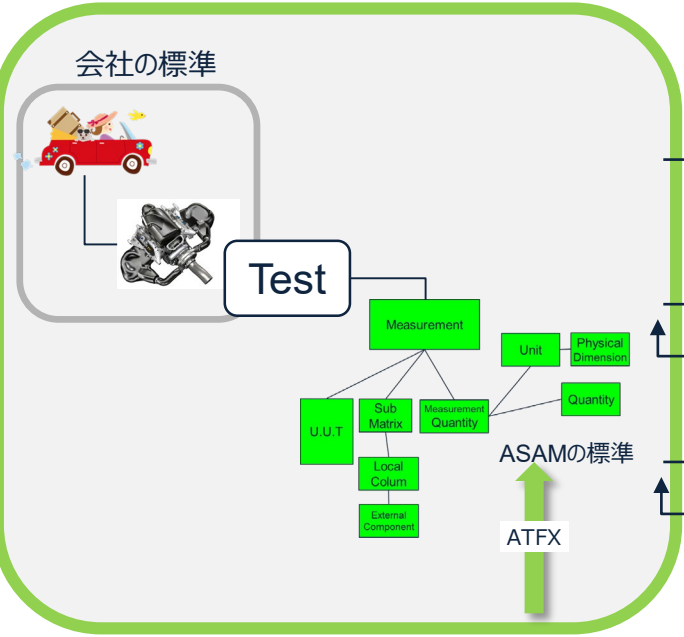
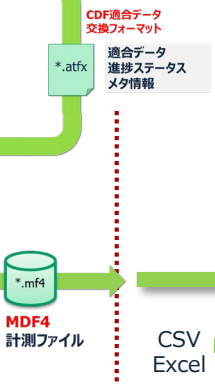
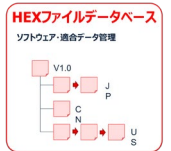
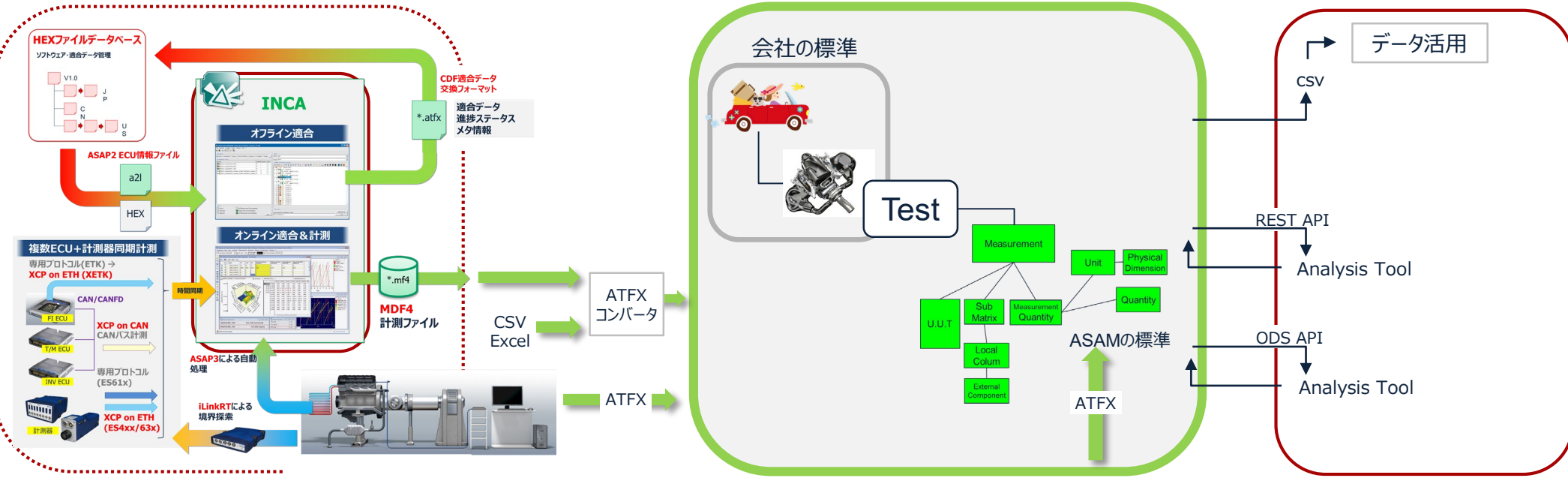
ODSからRESTでデータを取得し解析を、データ変換関数を読み取り生データと変換したデータの両方をグラフにする事を行います。

つながるData

外側 (API) の標準
MCツール

内側 (データモデル) の標準
ODS

活用の広がり



ODSサーバのMDFファイル対応（変換関数の追加）

株式会社 東陽テクニカ 岡田 真澄

ODSとMDFの互換性

ODSとMDFではASAM標準になる前からそれぞれ独自の歴史があり、互換性は考慮されていませんでした

思想の違い：

- MDFファイル

リアルタイム性を重視している

データの取りこぼしを最小限に抑えるために記録時間の最小化に焦点が当てられている。

生データを記録しておき、変換関数により値を補正する場合がある。

- ODSサーバ

メタ情報を中心としたデータベースで、データの有効活用を実現している。

計測データについては、チャンネルごとに格納するなど可読性に焦点が当てられている。

今までは、MDFで使用されているRAT_FUNC変換関数等の対応がODSの仕様中に含まれていませんでした。

変換関数による補正

補正の具体的な用途の例

CASE1: 環境の温度・湿度・気圧で左右されるような値を修正する場合

真の値は、実際の値に修正係数を掛けた値となります。

例えば、仕向け地の高度や温度などを考慮して、どれくらいの出力がでるのかななどを計算で求めることが可能になります。

CASE2 : 排出ガス法規制に基づく排出ガスを算出する場合

各種排出ガスを算出する用途で様々な補正を使用します。

例：各種排出ガス成分を重量として算出する際に使用します。

大気圧力補正

湿度補正

PMフィルタ重量算出時の浮力補正

変換関数による補正

補正の具体的な用途の例

CASE3:

- 温度、アクセル開度、アクチュエータ移動量などの物理データはECUメモリ内ではバイナリとして扱われます。またこれらのアナログ信号はECUのA/D入力回路を経由して取得されます。
- 具体的には物理現象とECU内の扱いは以下ようになります。
[現象] [センサ] [ECU] [ECU] [ECU] [ECU]
物理量 → アナログ出力 → A/D入力値 → (フィルタ) → ECU内でデータ量子化 → ECUソフトで計算
- アクセル開度、0-100% (0-255) 分解能0.1とした場合1ビットあたり約0.3%
- A/D入力値、10ビット分解能, 0-12V入力の仕様の場合は1ビットあたり11.7mV
- これを物理量として取り扱う場合は、A2Lファイルの情報を使用して1バイトのバイナリデータから物理量に変換しています。
- A2Lの情報はそのままMDFの補正情報として引き継がれます。

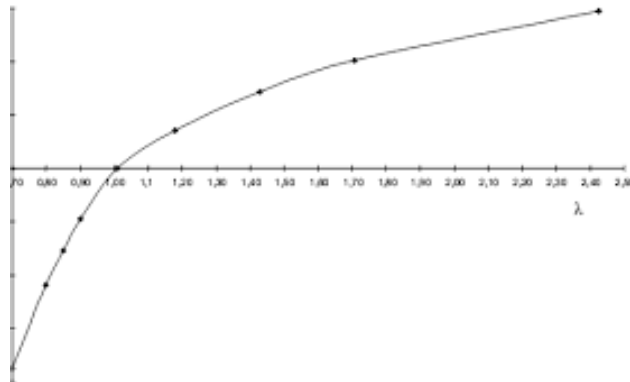
RAT_FUNC変換関数

RAT_FUNC補正の具体的な用途（ODS6.0で追加）

ECUのほとんどの信号は1次関数にて表現出来ます。

しかし、空燃比などはセンサの特性にて、リッチ領域では急変しリーン領域ではなだらかな曲線を持ちます。

これに類似する信号はRAT_FUNCで変換する必要が発生します。



ASAM-ODSで追加された変換関数

ASAM-ODS6.0ではMDFに対応するため、新たにraw_rationalが追加されました

| | |
|--------------------------------|-------|
| explicit | (=0) |
| implicit_constant | (=1) |
| implicit_linear | (=2) |
| implicit_saw | (=3) |
| raw_linear | (=4) |
| raw_polynomial | (=5) |
| formula | (=6) |
| external_component | (=7) |
| raw_linear_external | (=8) |
| raw_polynomial_external | (=9) |
| raw_linear_calibrated | (=10) |
| raw_linear_calibrated_external | (=11) |
| raw_rational | (=12) |
| raw_rational_external | (=13) |

| | |
|--------------------------------|---|
| raw_linear_external | $x_n = p_1 + p_2 \cdot r_n$ |
| raw_polynomial_external | $x_n = p_2 + p_3 \cdot r_n + p_4 \cdot r_n^2 + p_{2+p_1} \cdot r_n^{p_1}$ |
| formula | --- |
| raw_linear_calibrated | $x_n = (p_1 + p_2 \cdot r_n) \cdot p_3$ |
| raw_linear_calibrated_external | $x_n = (p_1 + p_2 \cdot r_n) \cdot p_3$ |
| raw_rational | $x_n = \frac{p_1 \cdot r_n^2 + p_2 \cdot r_n + p_3}{p_4 \cdot r_n^2 + p_5 \cdot r_n + p_6}$ |
| raw_rational_external | $x_n = \frac{p_1 \cdot r_n^2 + p_2 \cdot r_n + p_3}{p_4 \cdot r_n^2 + p_5 \cdot r_n + p_6}$ |

デモに使用するMDFデータの内容 (RAT_FUNC部分)

シグナルリスト

- \$ActiveCalibrationPage#EtasCalibrationDefaultRecordingDevice
- \$CalibrationLog#EtasCalibrationDefaultRecordingDevice
- ADC_Val#XCP:1**
- ADC_Conv#XCP:1
- adc.Demo_State.DemoComponent#XCP:1
- ADC_val_dec.Ports#XCP:1
- \$EVENT_COMMENTS

シグナルの詳細

| キー | 値 |
|---------|---|
| レート | 10ms_Raster |
| 注釈 | |
| コメント | <コメントなし> |
| ロングネーム | [1] ADC_Val#XCP:1 |
| 演算 | Yes |
| MIN/MAX | 0 / 0 |
| 変換式 | RAT (->raw) P[1] = 0 P[2] = 0.00488758553274682 P[3] = 0 P[4] = 0 P[5] = 0 P[6] = 1 |

シグナルリスト

- \$ActiveCalibrationPage#EtasCalibrationDefaultRecordingDevice
- \$CalibrationLog#EtasCalibrationDefaultRecordingDevice
- ADC_Val#XCP:1
- ADC_Conv#XCP:1**
- adc.Demo_State.DemoComponent#XCP:1
- ADC_val_dec.Ports#XCP:1
- \$EVENT_COMMENTS

シグナルの詳細

| キー | 値 |
|---------|--|
| レート | 10ms_Raster |
| 注釈 | |
| コメント | <コメントなし> |
| ロングネーム | [1] ADC_Conv#XCP:1 |
| 演算 | Yes |
| MIN/MAX | 0 / 0 |
| 変換式 | RAT (->raw) P[1] = 0 P[2] = 0.0146520146520147 P[3] = -10 P[4] = 0 P[5] = 0 P[6] = 1 |

ODSサーバに登録したサンプルデータの内容

The screenshot displays the MDM Database software interface. On the left, a tree view shows the project structure under 'ASAM-ODS'. A blue callout box highlights the 'ATFX GUI' folder, and another blue callout box highlights the 'MDF' folder. The 'MDF' folder contains a file named '2019-01-22 11_03_50measure.mf4', which is further expanded to show 'ADC_Val' and 'ADC_Conv'.

In the center, the 'Measurement results' table shows the following data:

| Measurement... | Path |
|--|---------------------------------|
| <input type="checkbox"/> 2019-01-22 11_03_50measure / ho | 2019-01-22 11_03_50measure / ho |
| <input checked="" type="checkbox"/> DA1 | ODS_WG_DEMO_LA4Cold / 2019 / |

Below the table, the 'Y-Axis' and 'X-Axis' configuration panels are visible. The Y-Axis table shows 'DA1TargetSp...' selected, and the X-Axis table shows 'SampleNo' selected.

On the right, a line graph titled 'Diagram' shows the 'DA1TargetSpeed [r]' over 'SampleNo [-]'. The graph displays a red line representing the target speed, which fluctuates between approximately 25 and 95 rpm over 11,000 samples. A legend at the bottom of the graph identifies the data series as 'DA1TargetSpeed (DA1 / ODS_WG_DEMO_LA4Cold / 2019 / ODS_WG / ATFX GUI)'.

At the bottom of the interface, the status bar indicates 'Logged in as: sa, MDM basic example [sa]' and 'Database: MDMDEMO [MDM]'.

デモの内容

◆ ODS6.0 で新たに追加された機能確認

- RAT_FUNC
- HTTP API
- Protocol Buffers

◆ MDFファイルのインポート、表示

◆ 各社ODSサーバの互換性確認

- Peak ODSサーバ (株式会社東陽テクニカ / Peak Solution)
- BRIX ODSサーバ (iASYS Technology Solutions株式会社)
- NI DataFinder Server (日本ナショナルインスツルメンツ株式会社)

ASAM ODS Study WG

ASAM ODS 6.0 REST API 活用 デモ

株式会社 堀場製作所 三十木 努

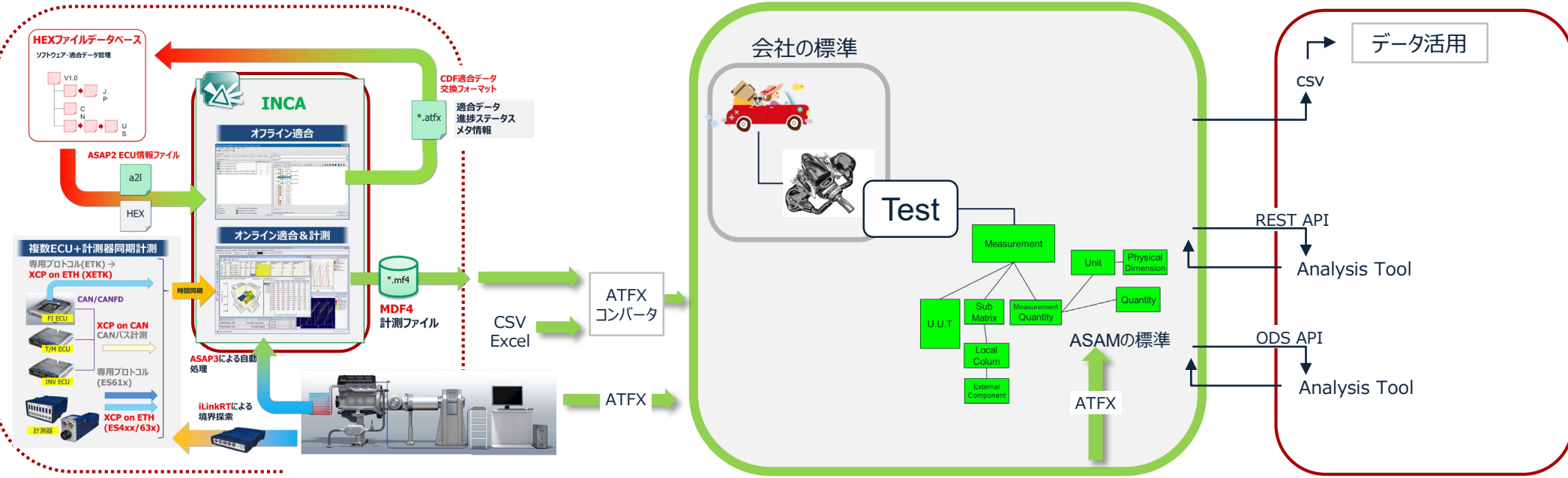
2019.JUL.27

つながるData

外側 (API) の標準
MCツール

内側 (データモデル) の標準
ODS

活用の広がり



ODS 6.0 REST API 活用

◆ 今までのCORBAと比較してメリット

- 多くの最新プログラミング言語での開発が可能 (CORBAは実質C++, Javaのみだった)
- HTTPSなどのセキュリティ通信、リバースプロキシの使用が可能
- APIが統合されて設計がシンプルになり使いやすくなった
- Notifications機能追加に伴いイベント連携が可能

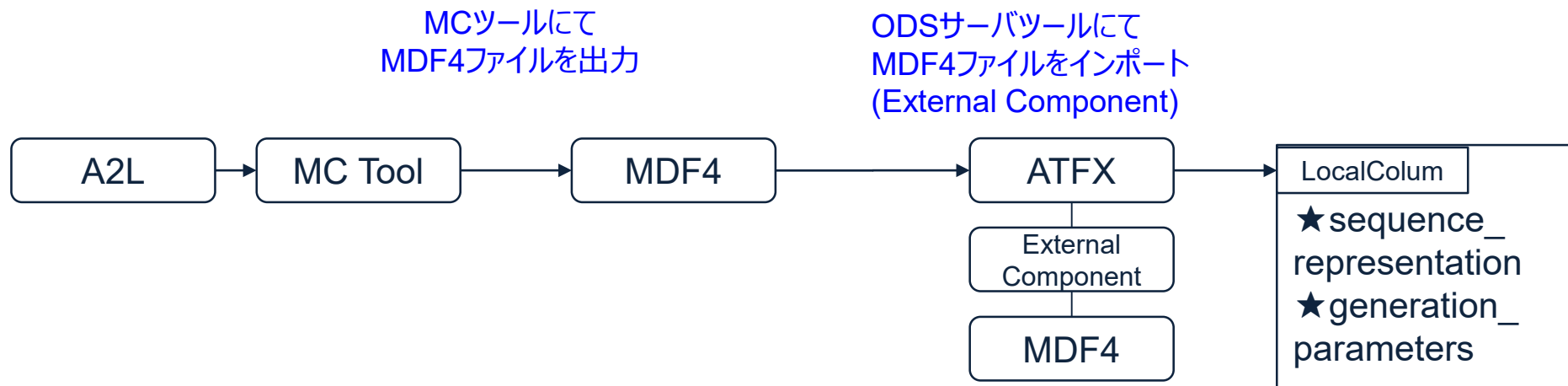
◆ ODS 6.0 REST APIを使ったシステム活用ユースケース

- 計測システムとの連携
CORBAからREST APIになることにより、計測システムアプリケーションからの利用が可能
- データ解析や統計などのWebアプリケーションとの連携
CORBAからREST APIになることにより、Webクライアントアプリケーションからの利用が可能
- データ分析基盤(Big Data Technologies)との連携
CORBAからREST APIになることにより、上位インデックスサービスとの連携が可能
- 他のシステムとの連携
CORBAからREST APIになることにより、API Gatewayなどを経由して他システム連携が可能

⇒ システムアプリケーションとの連携の敷居が下がりました。

ODS 6.0 デモ用データ (MDF4ファイル)

- ◆ MC Toolより生成された変換関数を含むMDF4ファイルを使用
- ◆ 事前に各ODSサーバ上にデータをインポート済み



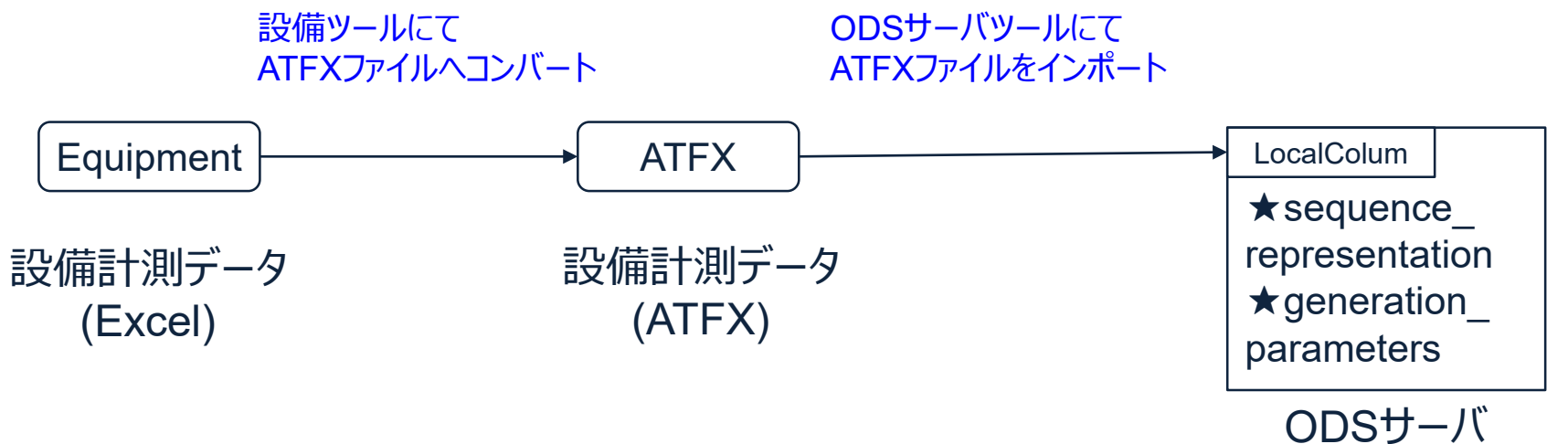
A2Lで変換関数
を持っている

データ生値と変換関数を
MDF4に保存

データ生値 + 変換関数が
ODSサーバに保存

ODS 6.0 デモ用データ (ATFXファイル)

- ◆ 計測システムより生成されたデータ(Excelファイル)をATFXファイルに変換したものを使用
- ◆ 事前に各ODSサーバ上にデータをインポート済み



データ生値と変換関数が
ATFXに保存

データ生値 + 変換関数が
ODSサーバに保存

ODS 6.0 サーバ デモ環境

- ◆ Peak ODSサーバ (株式会社東陽テクニカ / Peak Solution)
 - デモノートPC上のPeak ODSサーバにhttps接続

- ◆ BRIX ODSサーバ (iASYS Technology Solutions株式会社)
 - iASYS Technology Solutions株式会社のBRIX ODSサーバに4G回線にてhttp接続

- ◆ NI DataFinder Server (日本ナショナルインスツルメンツ株式会社)
 - 株式会社スカイテクノロジーのNI DataFinder Serverに4G回線にてhttp接続

ODS HTTP API デモクライアント (Python)

```

import sys
import os
import json
import requests
import urllib3
import logging
import argparse
import time
import random
import string

# 設定
BASE_URL = "http://localhost:8080"
API_PATH = "/api/v1/ods"

# ログ設定
logger = logging.getLogger(__name__)

def get_headers():
    """取得するヘッダ"""
    headers = {
        "Content-Type": "application/x-asamods+protobuf",
        "Accept": "application/x-asamods+json"
    }
    return headers

def get_data(endpoint):
    """データを取得"""
    url = BASE_URL + API_PATH + endpoint
    headers = get_headers()
    response = requests.get(url, headers=headers)
    return response

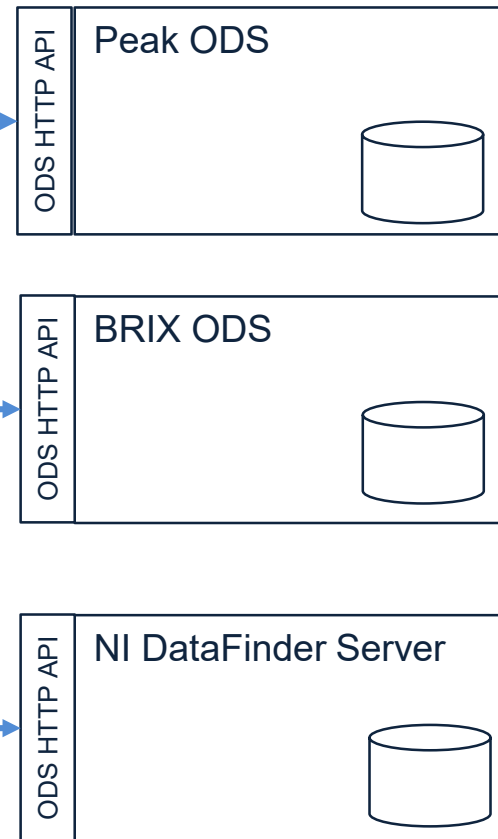
def main():
    parser = argparse.ArgumentParser()
    parser.add_argument("endpoint", help="API エンドポイント")
    args = parser.parse_args()

    response = get_data(args.endpoint)
    if response.status_code == 200:
        data = response.json()
        print(json.dumps(data, indent=2))
    else:
        logger.error(f"API エンドポイント {args.endpoint} にアクセスできませんでした。ステータスコード: {response.status_code}")

if __name__ == "__main__":
    main()
    
```

Python Demo Client

application/x-asamods+protobuf
or
Application/x-asamods+json



Python デモ内容

- ◆ データ解析にて標準的に使用されているPythonを使い、各社ODSサーバに対して共通のスク립トにて同じデータを取得できることを確認する。
- ◆ ユースケースとしては下記の2つのパターンを使用する
 - ① 計測データ一覧作成(CSVファイル保存)
 - ② 計測データIDを指定してデータ取得 (CSVファイル保存、グラフ表示)

The screenshot displays a Python IDE with a script for connecting to an ODS server. The code includes headers for application/x-asamodjson, a request to /api/v1/measurements, and a response that returns a list of measurement data. The data is then processed and printed to the console.

| measurement_id | measurement_name | submatrix_id | submatrix_name | | |
|----------------|------------------|----------------------|----------------|-------|----------------------|
| 79 | 10000 | ETASmeasure06.mf4 | 81 | 10000 | sm_00001 |
| 79 | 10000 | ETASmeasure06.mf4 | 81 | 10001 | sm_00002 |
| 79 | 10000 | ETASmeasure06.mf4 | 81 | 10002 | sm_00003 |
| 79 | 10000 | ETASmeasure06.mf4 | 81 | 10003 | sm_00004 |
| 79 | 10000 | ETASmeasure06.mf4 | 81 | 10004 | sm_00005 |
| 79 | 10000 | ETASmeasure06.mf4 | 81 | 10005 | sm_00006 |
| 79 | 10000 | ETASmeasure06.mf4 | 81 | 10006 | sm_00007 |
| 79 | 10000 | ETASmeasure06.mf4 | 81 | 10007 | sm_00008 |
| 79 | 10000 | ETASmeasure06.mf4 | 81 | 10008 | sm_00009 |
| 79 | 10001 | measure06.mf4_lookup | 81 | 10009 | Longitude_hemisphere |
| 79 | 10001 | measure06.mf4_lookup | 81 | 10010 | Latitude_hemisphere |
| 79 | 10100 | DA1 | 81 | 10100 | Measure |
| 79 | 10101 | 411 | 81 | 10101 | Measure |
| 79 | 10102 | GAS1 | 81 | 10102 | Measure |
| 79 | 10103 | CDM1 | 81 | 10103 | Measure |
| 79 | 10104 | SEQREP1 | 81 | 10104 | Measure |

The graph, titled "AoSMatrix Plot DA1 (valuematrix-read)", shows two data series over time (0 to 1000 seconds). The blue line represents "DATargetSpeed" and the green line represents "DAActualSpeed". The blue line shows a highly oscillatory pattern, while the green line shows a smoother, lower-frequency oscillation.

ODS HTTP API デモクライアント (C#)

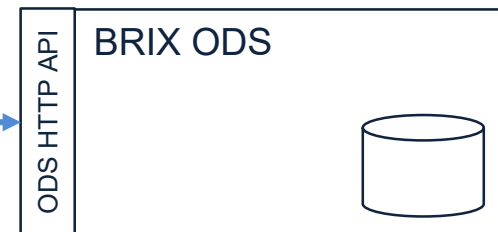
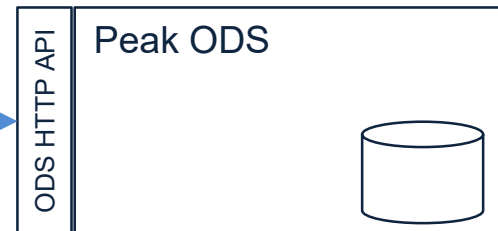
HORIBA ODS HTTP Data Viewer

URI:
Body:

| Instance Name | Application Model | Base Model | Application... | Instance Id |
|-------------------|-------------------|---------------|----------------|-------------|
| dummy | FunctionalGroup | AoTest | 21 | 1 |
| Tool Chain | Project | AoSubTest | 37 | 2 |
| Etas_TestPlan | TestPlan | AoSubTest | 44 | 3 |
| rec1_002 | Test | AoSubTest | 49 | 4 |
| Mea_00001_ | MeaResult | AoMeasurement | 51 | 24 |
| sm_00001 | SubMatrix | AoSubmatrix | 54 | 19 |
| Toyota_TestPlan | TestPlan | AoSubTest | 44 | 4 |
| Horiba_TestPlan | TestPlan | AoSubTest | 44 | 2 |
| Test1 | Test | AoSubTest | 49 | 2 |
| DA1 | MeaResult | AoMeasurement | 51 | 3 |
| Measure | SubMatrix | AoSubmatrix | 54 | 3 |
| GAS1 | MeaResult | AoMeasurement | 51 | 4 |
| CDM1 | MeaResult | AoMeasurement | 51 | 5 |
| SEQREP1 | MeaResult | AoMeasurement | 51 | 6 |
| AI1 | MeaResult | AoMeasurement | 51 | 7 |
| Test2 | Test | AoSubTest | 49 | 3 |
| Onosokki_TestPlan | TestPlan | AoSubTest | 44 | 1 |

C# Demo Client

application/x-asamods+protobuf

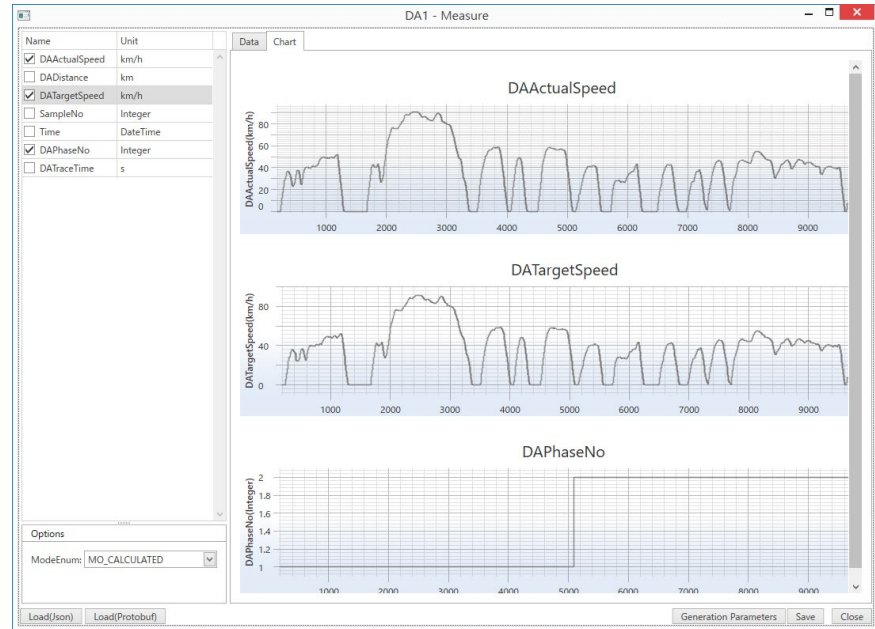


C# デモ内容

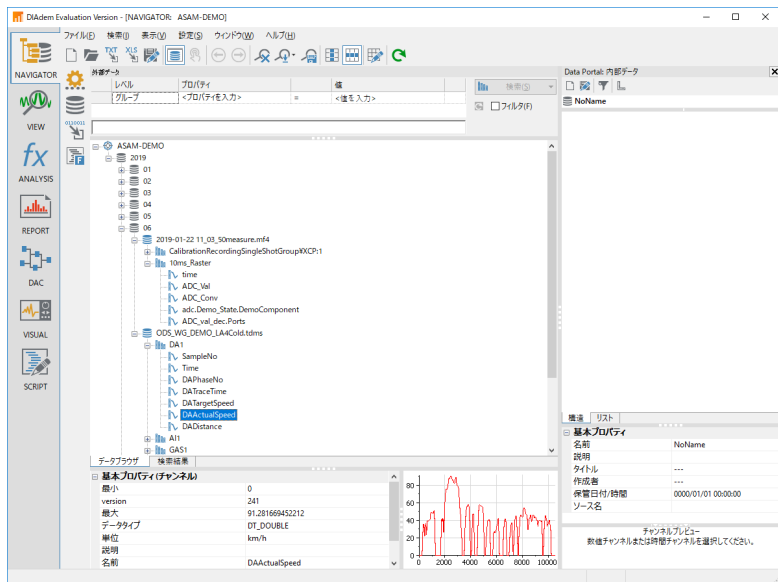
- ◆ PCアプリ開発にて標準的に使用されているC#を使い、各社ODSサーバに対して共通アプリにて同じデータをProtobufにて取得できることを確認する。
- ◆ ユースケースとしては下記の2つのパターンを使用する
 - ① 計測データ一覧作成/表示
 - ② ①の計測データIDを選択してデータ表示/グラフ表示

The screenshot shows the 'HORIBA ODS HTTP Data Viewer' application. It has a URI field set to 'http://114.143.140.82:8086/briv/ods' and a 'Connect' button. Below is a tree view of data instances. The 'Measure' item under 'Horiba_TestPlan' is selected.

| Instance Name | Application Model | Base Model | Application... | Instance Id |
|-------------------|-------------------|---------------|----------------|-------------|
| dummy | FunctionalGroup | AoTest | | 21 |
| Tool Chain | Project | AoSubTest | | 37 |
| Etas_TestPlan | TestPlan | AoSubTest | | 44 |
| rec1_002 | Test | AoSubTest | | 49 |
| Mea_00001 | MeaResult | AoMeasurement | | 51 |
| sm_00001 | SubMatrix | AoSubmatrix | | 54 |
| Toyota_TestPlan | TestPlan | AoSubTest | | 44 |
| Horiba_TestPlan | TestPlan | AoSubTest | | 44 |
| Test1 | Test | AoSubTest | | 49 |
| DA1 | MeaResult | AoMeasurement | | 51 |
| GAS1 | MeaResult | AoMeasurement | | 51 |
| CDM1 | MeaResult | AoMeasurement | | 51 |
| SEQREP1 | MeaResult | AoMeasurement | | 51 |
| Measure | SubMatrix | AoSubmatrix | | 54 |
| All1 | MeaResult | AoMeasurement | | 51 |
| Test2 | Test | AoSubTest | | 49 |
| Onosokki_TestPlan | TestPlan | AoSubTest | | 44 |

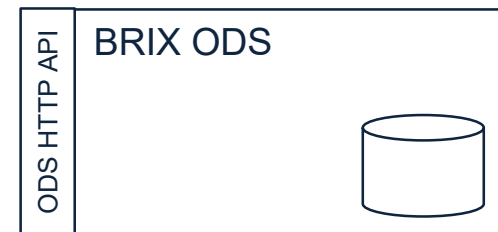
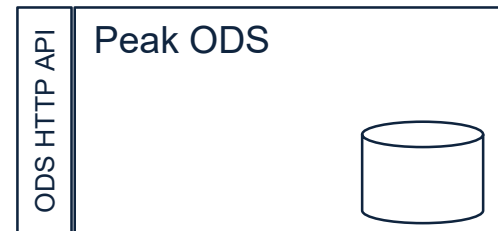


ODS HTTP API デモクライアント (解析ツール NI DIADem)



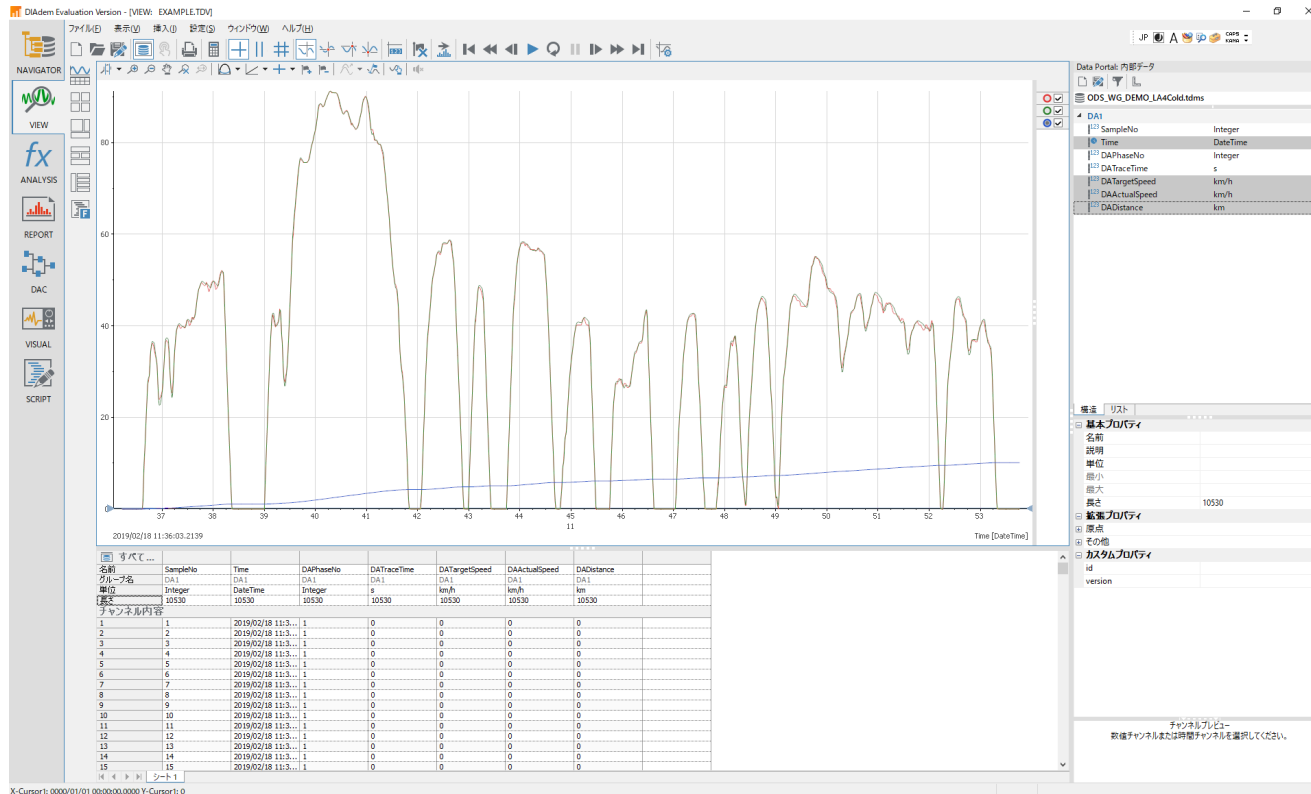
NI DIADem

application/x-asamods+protobuf

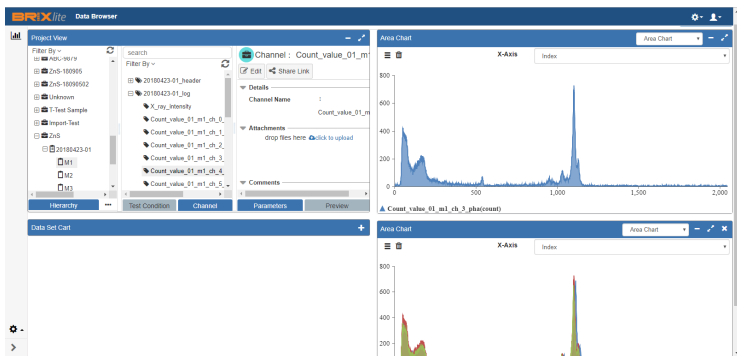


解析ツール NI DIADem 2019 デモ内容

- ◆ 最新NI DIADem 2019にてNI DataFinder ServerにODS 6.0 HTTP APIにて接続し、データ取得してグラフ表示を行う。



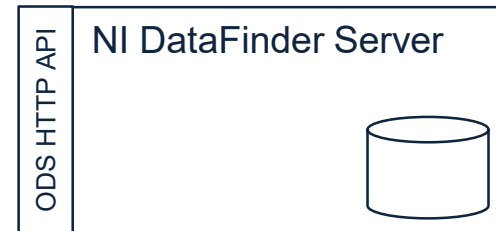
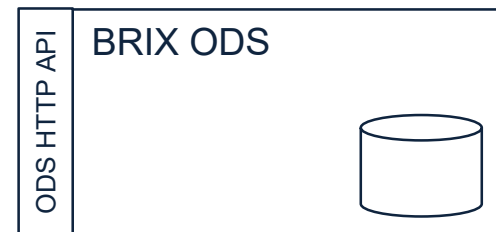
ODS HTTP API デモクライアント (Brix Viewer)



Brix Viewer

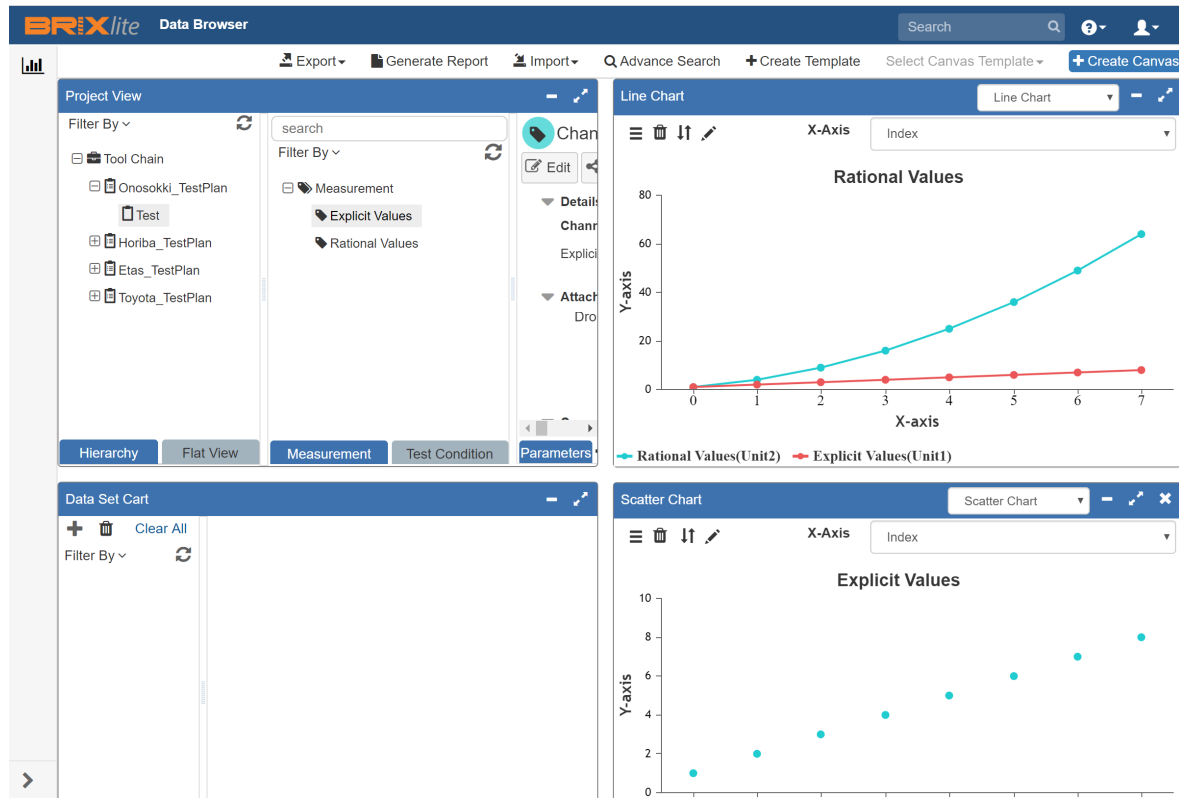
BRiX

application/x-asamods+protobuf



ODSサーバビューワー Brix Viewer デモ内容

- ◆ Brix ViewerにてiASYS Brix ODS ServerにODS 6.0 HTTP APIにて接続し、データ取得してグラフ表示を行う。



Python デモ環境

- ◆ Python 3.7 : Python本体

- ◆ Python package
 - ① Requests : http通信ライブラリ (pip install requests)
 - ② Pandas : データ解析ライブラリ (pip install pandas)
 - ③ Matplotlib : グラフ描画ライブラリ (pip install matplotlib)
 - ④ Protobuf : Protocol Buffer用ライブラリ (pip install protobuf)

- ◆ Jupyter Lab : データ分析作業ツール (pip install jupyterlab)

Protocol Buffers

◆ Protocol Buffersとは

- Google製のインターフェース定義言語(IDL)で構造を定義するデータ通信・永続化のためのシリアライズフォーマット多くの言語でサポートされている。
- プロトコル定義ファイル(.proto)でデザインし、protocプログラムによりコンパイルされる。コンパイルにより目的の言語のコード(.cs, .pyなど)が生成される。

◆ Protocol BuffersがJSON/XMLに勝る点

- データサイズが小さいため通信が高速化
- エンコード、デコードが速くて省メモリ
- 型安全に通信処理が書ける

◆ Protocol BuffersがJSON/XMLに劣る点

- 通信がバイナリのためで可読性に問題がある。
- Javascriptなどでバイナリの扱いは向かないのでフロントエンドではJSONを使用する方が良い
- Protocol Buffersのバージョンに依存する可能性がある (ODSはProto3を使用する)

デモで使用するODS 6.0 HTTP API (データアクセスのみ)

<接続/切断>

POST /ods : セッションID取得 (Establish Connection)
DELETE /ods : セッションID解放 (Close Connection)

<モデル情報>

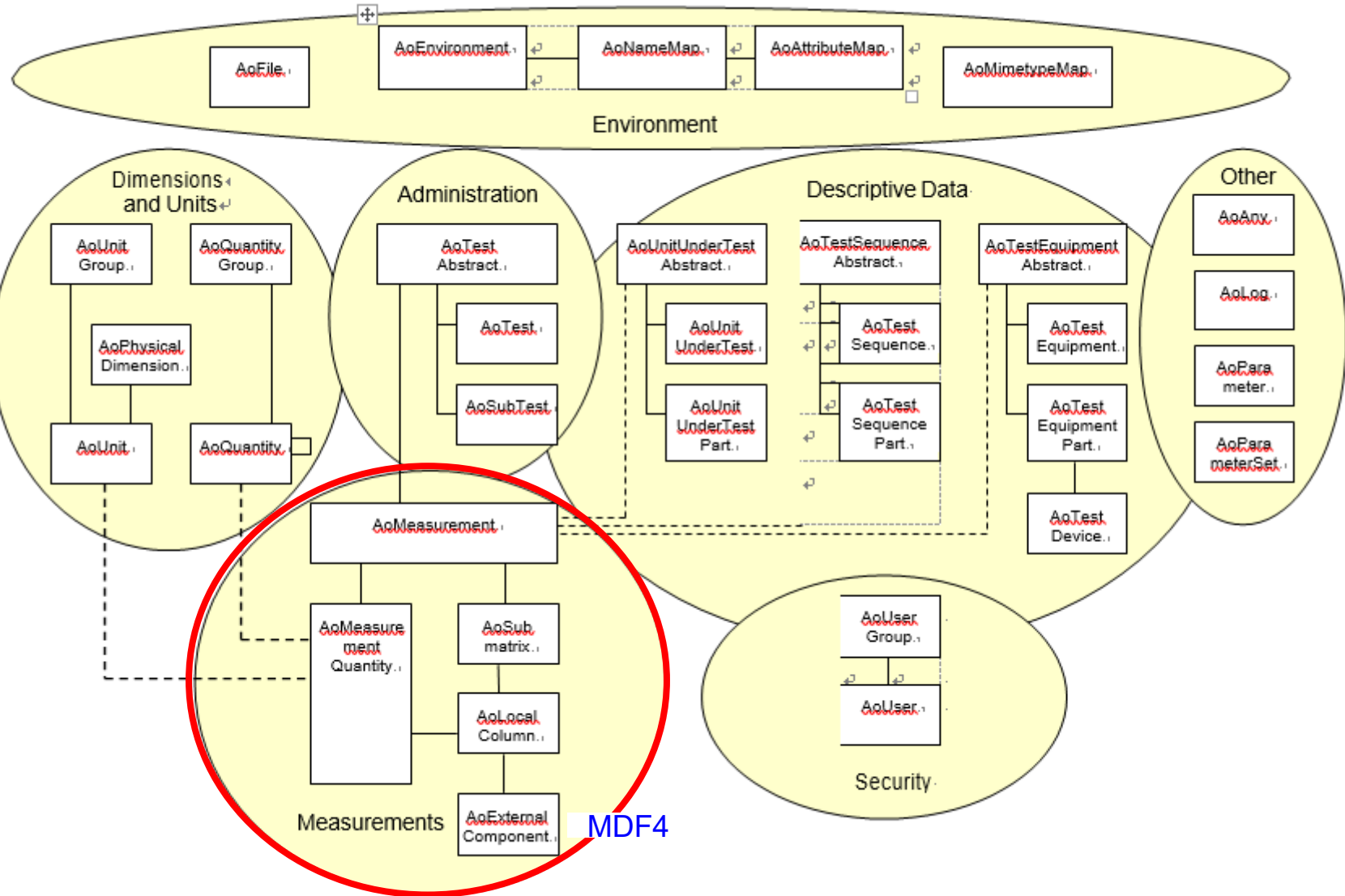
POST /ods/{conl}/basemodel-read : ベースモデル取得
POST /ods/{conl}/model-read : アプリケーションモデル取得

<データ検索>

POST /ods/{conl}/data-read : インスタンス検索/Matrixデータ取得

<データ取得>

POST /ods/{conl}/valuematrix-read : Matrixデータ取得



MDF4

計測データ一覧作成の流れ

1. セッションIDを取得 (`POST /ods`)
2. アプリケーションモデルを取得 (`POST /ods/{con1}/model-read`)
モデルID(aid)、モデル名(name)、属性名(attribute name)、
リレーション名(relation name)、モデルリレーション(relations)を取得する。
⇒ インスタンス検索やデータ取得で必要となる。
3. アプリケーションモデルを元にインスタンス階層を検索 (`POST /ods/{con1}/data-read`)
AoTest iid → AoSubtest iid → … → AoMeasurement iid → AoSubMatrix iid
⇒ 検索条件を付けてインスタンスの絞り込みを行うことも可能
4. インスタンス検索結果よりテーブルを作成してCSVファイルに保存
⇒ ここで取得した計測データID (AoSubMatrix iid)を使ってデータを取得
5. セッションIDを解放 (`DELETE /ods/{con1}`)
明示的に開放しないとサーバ側で自動破棄されるまでセッションは解放されない

計測データ一覧作成のデモ

JupyterLab interface showing a file browser on the left and a data table in the main area.

File Browser (Left Panel):

| Name | Last Modified |
|--|----------------|
| doc | 25 days ago |
| output | 2 minutes ago |
| py | 2 minutes ago |
| venv | 2 months ago |
| OdSHttpDemo_BRIX_MeasurementList.ipynb | 4 days ago |
| OdSHttpDemo_BRIX_valuematrix-read_DA1_json.ipynb | 4 days ago |
| OdSHttpDemo_BRIX_valuematrix-read_DA1_protobuf.ipynb | 4 days ago |
| OdSHttpDemo_BRIX_valuematrix-read_SEQREP1.ipynb | 4 days ago |
| OdSHttpDemo_NI_MeasurementList.ipynb | 4 days ago |
| OdSHttpDemo_NI_valuematrix-read_DA1_json.ipynb | 3 days ago |
| OdSHttpDemo_NI_valuematrix-read_DA1_protobuf.ipynb | 9 minutes ago |
| OdSHttpDemo_NI_valuematrix-read_SEQREP1.ipynb | 4 days ago |
| OdSHttpDemo_PEAK_MeasurementList.ipynb | 4 days ago |
| OdSHttpDemo_PEAK_valuematrix-read_DA1_json.ipynb | 13 minutes ago |
| OdSHttpDemo_PEAK_valuematrix-read_DA1_protobuf.ipynb | 13 minutes ago |
| OdSHttpDemo_PEAK_valuematrix-read_SEQREP1.ipynb | 12 minutes ago |
| application_model_BRIX.json | 2 minutes ago |
| base_model_BRIX.json | 2 minutes ago |
| measurementquantity_instance_BRIX.csv | a minute ago |
| ods_notification_pb2.py | 20 days ago |
| ods_pb2.py | 20 days ago |
| ods_security_pb2.py | 20 days ago |
| submatrix_instance_BRIX.csv | a minute ago |
| submatrix_plot_DA1_MO_CALCULATED_BRIX_Protobuf.csv | a minute ago |
| submatrix_plot_DA1_MO_CALCULATED_BRIX_Protobuf.pdf | a minute ago |
| submatrix_plot_DA1_MO_CALCULATED_BRIX.csv | a minute ago |
| submatrix_plot_DA1_MO_CALCULATED_BRIX.pdf | a minute ago |
| units_instance_BRIX.csv | 2 minutes ago |

Data Table (Main Panel):

| | test_aid | test_id | test_name | subtest1_aid | subtest1_id | subtest1_name | subtest2_aid | subtest2_id | subtest2_name |
|----|----------|---------|-----------|--------------|-------------|---------------|--------------|-------------|-----------------|
| 1 | 21 | 1 | dummy | 37 | 2 | Tool Chain | 44 | 3 | Etas_TestPlan |
| 2 | 21 | 1 | dummy | 37 | 2 | Tool Chain | 44 | 3 | Etas_TestPlan |
| 3 | 21 | 1 | dummy | 37 | 2 | Tool Chain | 44 | 3 | Etas_TestPlan |
| 4 | 21 | 1 | dummy | 37 | 2 | Tool Chain | 44 | 3 | Etas_TestPlan |
| 5 | 21 | 1 | dummy | 37 | 2 | Tool Chain | 44 | 4 | Toyota_TestPlan |
| 6 | 21 | 1 | dummy | 37 | 2 | Tool Chain | 44 | 4 | Toyota_TestPlan |
| 7 | 21 | 1 | dummy | 37 | 2 | Tool Chain | 44 | 4 | Toyota_TestPlan |
| 8 | 21 | 1 | dummy | 37 | 2 | Tool Chain | 44 | 4 | Toyota_TestPlan |
| 9 | 21 | 1 | dummy | 37 | 2 | Tool Chain | 44 | 4 | Toyota_TestPlan |
| 10 | 21 | 1 | dummy | 37 | 2 | Tool Chain | 44 | 4 | Toyota_TestPlan |
| 11 | 21 | 1 | dummy | 37 | 2 | Tool Chain | 44 | 4 | Toyota_TestPlan |
| 12 | 21 | 1 | dummy | 37 | 2 | Tool Chain | 44 | 4 | Toyota_TestPlan |
| 13 | 21 | 1 | dummy | 37 | 2 | Tool Chain | 44 | 2 | Horiba_TestPlan |
| 14 | 21 | 1 | dummy | 37 | 2 | Tool Chain | 44 | 2 | Horiba_TestPlan |
| 15 | 21 | 1 | dummy | 37 | 2 | Tool Chain | 44 | 2 | Horiba_TestPlan |
| 16 | 21 | 1 | dummy | 37 | 2 | Tool Chain | 44 | 2 | Horiba_TestPlan |
| 17 | 21 | 1 | dummy | 37 | 2 | Tool Chain | 44 | 2 | Horiba_TestPlan |
| 18 | 21 | 1 | dummy | 37 | 2 | Tool Chain | 44 | 2 | Horiba_TestPlan |
| 19 | 21 | 1 | dummy | 37 | 2 | Tool Chain | 44 | 2 | Horiba_TestPlan |
| 20 | 21 | 1 | dummy | 37 | 2 | Tool Chain | 44 | 2 | Horiba_TestPlan |
| 21 | 21 | 1 | dummy | 37 | 2 | Tool Chain | 44 | 2 | Horiba_TestPlan |
| 22 | 21 | 1 | dummy | 37 | 2 | Tool Chain | 44 | 2 | Horiba_TestPlan |
| 23 | 21 | 1 | dummy | 37 | 2 | Tool Chain | 44 | 2 | Horiba_TestPlan |
| 24 | 21 | 1 | dummy | 37 | 2 | Tool Chain | 44 | 2 | Horiba_TestPlan |
| 25 | 21 | 1 | dummy | 37 | 2 | Tool Chain | 44 | 2 | Horiba_TestPlan |
| 26 | 21 | 1 | dummy | 37 | 2 | Tool Chain | 44 | 2 | Horiba_TestPlan |
| 27 | 21 | 1 | dummy | 37 | 2 | Tool Chain | 44 | 2 | Horiba_TestPlan |
| 28 | 21 | 1 | dummy | 37 | 2 | Tool Chain | 44 | 2 | Horiba_TestPlan |
| 29 | 21 | 1 | dummy | 37 | 2 | Tool Chain | 44 | 2 | Horiba_TestPlan |
| 30 | 21 | 1 | dummy | 37 | 2 | Tool Chain | 44 | 2 | Horiba_TestPlan |
| 31 | 21 | 1 | dummy | 37 | 2 | Tool Chain | 44 | 2 | Horiba_TestPlan |
| 32 | 21 | 1 | dummy | 37 | 2 | Tool Chain | 44 | 2 | Horiba_TestPlan |
| 33 | 21 | 1 | dummy | 37 | 2 | Tool Chain | 44 | 2 | Horiba_TestPlan |
| 34 | 21 | 1 | dummy | 37 | 2 | Tool Chain | 44 | 2 | Horiba_TestPlan |
| 35 | 21 | 1 | dummy | 37 | 2 | Tool Chain | 44 | 2 | Horiba_TestPlan |
| 36 | 21 | 1 | dummy | 37 | 2 | Tool Chain | 44 | 2 | Horiba_TestPlan |
| 37 | 21 | 1 | dummy | 37 | 2 | Tool Chain | 44 | 2 | Horiba_TestPlan |
| 38 | 21 | 1 | dummy | 37 | 2 | Tool Chain | 44 | 2 | Horiba_TestPlan |
| 39 | 21 | 1 | dummy | 37 | 2 | Tool Chain | 44 | 2 | Horiba_TestPlan |

POST /ods/{con1}/data-read SelectStatement

Table 44 - SelectStatement

| ID | oneOf | Mult | Datatype | Name | Description |
|----|-------|------|-------------------------------|--------------|---|
| 1 | | 0..n | AttributItem | columns | requested attributes; optionally with aggregation and/or unit conversion; |
| 2 | | 0..n | ConditionItem | where | select constraint / "SQL where clause" |
| 3 | | 0..n | JoinItem | joins | join conditions for connecting instances of multiple Application Elements / "SQL join" |
| 4 | | 0..n | OrderByItem | order_by | ordering *of the full result set* according to provided attributes / "SQL order by" |
| 5 | | 0..n | GroupByItem | group_by | aggregation specification / "SQL group by" |
| 6 | | 0..1 | int64 | row_limit | max. number of instances, starting at row_start (i.e. "page size"); default/0 means all |
| 7 | | 0..1 | int64 | row_start | specifies the index of the first result row (0-based) |
| 8 | | 0..1 | int64 | values_limit | only for 'values', 'flags'; max number of values; default/0 means all |
| 9 | | 0..1 | int64 | values_start | only for 'values', 'flags'; index of first value to return (0-based). |

POST /ods/{conl}/data-read サンプル

【例】AoSubtest aid:77, iid:10200に属するAoMeasurement aid/nameの一覧を取得

POST /ods/{conl}/data-read

Request body :

```
{
  "columns": [
    {"aid": "79", "attribute": "Id"},
    {"aid": "79", "attribute": "Name"}],
  "where": [
    {"condition": {"aid": "77", "attribute": "Id", "longlong_array": {"values": ["10200"]}}}]
}
```

取得したいattributeを指定する
(SQLのSELECTフィールド相当、"*"も対応)

インスタンス検索の集約条件を指定する
(SQLのWHERE句相当)

Response body :

```
{
  "matrices": [
    {"name": "MeaResult", "aid": "79",
     "columns": [{
       "name": "Id",
       "dataType": "DT_LONGLONG",
       "isNull": [false, false, false, false, false],
       "longlongArray": {"values": ["10100", "10101", "10102", "10103", "10104"]}
     }, {
       "name": "Name",
       "dataType": "DT_STRING",
       "isNull": [false, false, false, false, false],
       "stringArray": {"values": ["DA1", "AI1", "GAS1", "CDM1", "SEQREP1"]}
     }
    ]
  }
}
```

計測データIDを指定してデータ取得の流れ

1. セッションIDを取得 (`POST /ods`)
2. モデルID/インスタンスIdを指定してデータ取得(`POST /ods/{con1}/valuematrix-read`)
RAT_FUNCTION(generation_parameters)のサーバでの演算有無はクエリパラメータのモード設定(`MO_CALCULATED:演算値`、`MO_STORAGE:保存値`) により選択が可能
3. 取得したデータよりテーブルを作成してCSVファイルに保存
⇒ 時系列計測データやバッチ計測データなど
4. 作成したテーブルを元にグラフを作成して表示/保存
5. セッションIDを解放 (`DELETE /ods/{con1}`)

計測データIDを指定してデータ取得のデモ

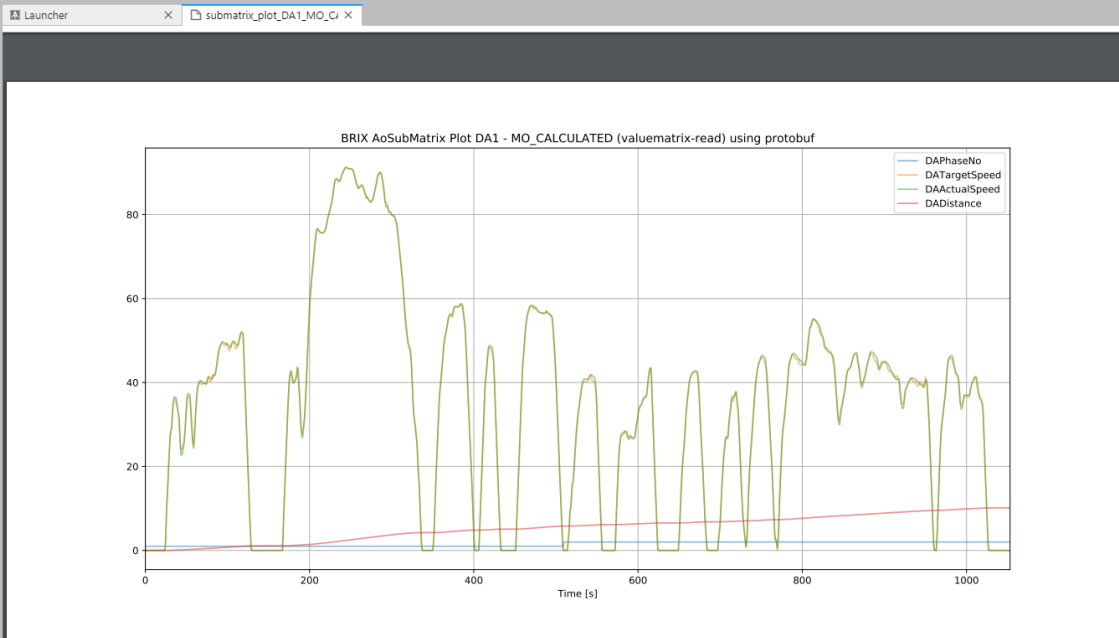
JupyterLab interface showing a file browser on the left and a plot on the right.

The file browser displays a list of files and folders, including:

- doc (25 days ago)
- output (3 minutes ago)
- py (3 minutes ago)
- venv (2 months ago)
- OdsHttpDemo_BRiX_MeasurementList.ipynb (4 days ago)
- OdsHttpDemo_BRiX_valuematrix-read_DA1_json.ipynb (4 days ago)
- OdsHttpDemo_BRiX_valuematrix-read_DA1_protobuf.ipynb (4 days ago)
- OdsHttpDemo_BRiX_valuematrix-read_SEQREP1.ipynb (4 days ago)
- OdsHttpDemo_NI_MeasurementList.ipynb (4 days ago)
- OdsHttpDemo_NI_valuematrix-read_DA1_json.ipynb (3 days ago)
- OdsHttpDemo_NI_valuematrix-read_DA1_protobuf.ipynb (10 minutes ago)
- OdsHttpDemo_NI_valuematrix-read_SEQREP1.ipynb (4 days ago)
- OdsHttpDemo_PEAK_MeasurementList.ipynb (4 days ago)
- OdsHttpDemo_PEAK_valuematrix-read_DA1_json.ipynb (14 minutes ago)
- OdsHttpDemo_PEAK_valuematrix-read_DA1_protobuf.ipynb (14 minutes ago)
- OdsHttpDemo_PEAK_valuematrix-read_SEQREP1.ipynb (13 minutes ago)
- application_model_BRiX.json (2 minutes ago)
- base_model_BRiX.json (2 minutes ago)
- measurementquantity_instance_BRiX.csv (2 minutes ago)
- ods_notification_pb2.py (20 days ago)
- ods_pb2.py (20 days ago)
- ods_security_pb2.py (20 days ago)
- submatrix_instance_BRiX.csv (2 minutes ago)
- submatrix_plot_DA1_MO_CALCULATED_BRiX_Protobuf.csv (2 minutes ago)
- submatrix_plot_DA1_MO_CALCULATED_BRiX_Protobuf.pdf (2 minutes ago)**
- submatrix_plot_DA1_MO_CALCULATED_BRiX.csv (2 minutes ago)
- submatrix_plot_DA1_MO_CALCULATED_BRiX.pdf (2 minutes ago)
- units_instance_BRiX.csv (2 minutes ago)

The plot on the right is titled "BRiX AoSubMatrix Plot DA1 - MO_CALCULATED (valuematrix-read) using protobuf". The x-axis is labeled "Time [s]" and ranges from 0 to 1000. The y-axis ranges from 0 to 80. The plot shows four data series:

- DAPhaseNo (blue line): A constant horizontal line near 0.
- DATargetSpeed (orange line): A constant horizontal line near 0.
- DAActualSpeed (green line): A highly oscillatory signal that peaks around 80 and drops to 0.
- DAADistance (red line): A slowly increasing linear signal from 0 to approximately 10.



POST /ods/{con1}/valuematrix-read SelectStatement

Table 51 - ValueMatrixRequestStruct

| ID | oneOf | Mult | Datatype | Name | Description |
|----|-------|------|------------|--------------|---|
| 1 | | 0..1 | ModeEnum | mode | |
| 2 | | 0..1 | int64 | aid | Application Element Id of type AoMeasurement, AoSubMatrix |
| 3 | | 0..1 | int64 | iid | |
| 4 | | 0..n | string | attributes | Only: 'name', 'values', 'flags', 'generation_parameters' |
| 5 | | 0..n | ColumnItem | columns | |
| 6 | | 0..1 | int64 | values_limit | max number of values; default/0 means all |
| 7 | | 0..1 | int64 | values_start | index of first value to return (0-based). |

POST /ods/{conl}/valuematrix-read SelectStatement

Table 12 - ModeEnum

| ID | Name | Description |
|----|---------------|-------------|
| 0 | MO_CALCULATED | |
| 1 | MO_STORAGE | |

Table 52 - ColumnItem

| ID | oneOf | Mult | Datatype | Name | Description |
|----|-------|------|----------|---------|---|
| 1 | | 0..1 | string | name | name Pattern of the measurement quantity / local column. May contain wildcards. |
| 2 | | 0..1 | int64 | unit_id | unit in which the measurement values shall be returned; 0 means default |

POST /ods/{conl}/valuematrix-read サンプル 1

【例1】 AoSubmatrix aid:81, iid:10104のLocalColumn Name:"RawLinear"のRAT_FUNCTION演算値データを取得

POST /ods/{conl}/valuematrix-read

Request body :

```
{
  "aid": "81", "iid": 10104,
  "mode": "MO_CALCULATED",
  "attributes": ["name", "values", "flags", "generation_parameters"],
  "values_start": 0,
  "values_limit": 10,
  "columns": [{"name": "RawLinear", "unit_id": 0}]
}
```

- ← 対象のSubmatrix aid, iid
- ← モード(MO_CALCULATED:演算値)の選択
- ← 取得するattribute指定 (name, values, flags, generation_parameters)
- ← 取得するデータ範囲指定 (value_start:開始位置、value_limit:取得数上限)
- ← 取得するデータ指定 ("unit_id"は取得する単位IDの指定(0:保存値))

Response body :

```
{
  "matrices": [{
    "name": "LocalColumn", "baseName": "AoLocalColumn", "aid": "82",
    "columns": [
      {"name": "Name", "baseName": "name", "dataType": "DT_STRING", "stringArray": {"values": ["RawLinear"]}},
      {"name": "Values", "baseName": "values",
        "unknownArrays": {"values": [{"dataType": "DT_DOUBLE", "unitId": "76",
          "doubleArray": {"values": [1.1, 1.2, 1.3, 1.4, 1.5, 1.6, 1.7, 1.8, 1.9, 2.0]}}}},
      {"name": "Flags", "baseName": "flags", "dataType": "DS_SHORT",
        "longArrays": {"values": [{"values": [15, 15, 15, 15, 15, 15, 15, 15, 15]}}}},
      {"name": "GenerationParameters", "baseName": "generation_parameters",
        "unknownArrays": {"values": [{"dataType": "DT_DOUBLE", "doubleArray": {}}}}
    ]
  }
}
```

- ← モード(MO_CALCULATED)選択時は、GenerationParametersはサーバ側で使用されるため演算値データが返される。
- ← モード(MO_CALCULATED)選択時は、GenerationParametersはサーバ側で使用されるためレスポンスには入らない。

POST /ods/{conl}/valuematrix-read サンプル 2

【例1】AoSubmatrix aid:81, iid:10104のLocalColumn Name:"RawLinear"のRAT_FUNCTION演算値データを取得

POST /ods/{conl}/valuematrix-read

Request body :

```
{
  "aid": "81", "iid": 10104,
  "mode": "MO_STORAGE",
  "attributes": ["name", "values", "flags", "generation_parameters"],
  "values_start": 0,
  "values_limit": 10,
  "columns": [{"name": "RawLinear", "unit_id": 0}]
}
```

- ← 対象のSubmatrix aid, iid
- ← モード(MO_STORAGE:保存値)の選択
- ← 取得するattribute指定 (name, values, flags, generation_parameters)
- ← 取得するデータ範囲指定 (value_start:開始位置、value_limit:取得数上限)
- ← 取得するデータ指定 ("unit_id"は取得する単位IDの指定(0:保存値))

Response body :

```
{
  "matrices": [{
    "name": "LocalColumn", "baseName": "AoLocalColumn", "aid": "82",
    "columns": [
      {"name": "Name", "baseName": "name", "dataType": "DT_STRING", "stringArray": {"values": ["RawLinear"]}},
      {"name": "Values", "baseName": "values",
        "unknownArrays": {"values": [{"dataType": "DT_DOUBLE", "unitId": "76",
          "doubleArray": {"values": [1.0, 2.0, 3.0, 4.0, 5.0, 6.0, 7.0, 8.0, 9.0, 10.0]}]}},
      {"name": "Flags", "baseName": "flags", "dataType": "DS_SHORT",
        "longArrays": {"values": [{"values": [15, 15, 15, 15, 15, 15, 15, 15, 15]}]}},
      {"name": "GenerationParameters", "baseName": "generation_parameters",
        "unknownArrays": {"values": [{"dataType": "DT_DOUBLE", "doubleArray": {"values": [1.0, 0.1]}]}]}
    ]
  }
]
```

- ← モード(MO_STORAGE)選択時は、GenerationParametersはサーバ側で使用されないため保存値データが返される。
- ← モード(MO_STORAGE)選択時は、GenerationParametersはサーバ側で使用されないためレスポンスに含まれる。

変換関数への対応 デモ (MDF4)

Mea_00001_ - sm_00001

| Name | Unit |
|--|------|
| <input checked="" type="checkbox"/> MainFunctionTask2_102400ns | - |
| <input checked="" type="checkbox"/> test_ods_ram1 | rpm |
| <input checked="" type="checkbox"/> test_ods_ram2 | °C |
| <input checked="" type="checkbox"/> test_ods_flag1 | - |

Generation Parameters

| Name | GenerationParameters |
|------------------|----------------------|
| MainFunctionT... | |
| test_ods_ram1 | 0 0.78125 |
| test_ods_ram2 | -40 0.00244140625 |
| test_ods_flag1 | |

Options

ModeEnum: MO_STORAGE

Load(/json) Load(Protobuf) Generation Parameters Save Close

| Data | Chart |
|-----------------------|--|
| Main Function Task... | test_ods_ram1 test_ods_ram2 test_ods_flag1 |
| 0 | 3381 53252 |
| 0.00010192588565... | 3381 53252 |
| 0.00020424683289... | 3381 53252 |
| 0.00030617271854... | 3381 53252 |
| 0.00040809860420... | 3381 53252 |
| 0.00051002448986... | 3381 53252 |
| 0.00061195037552... | 3381 53252 |
| 0.00071427132275... | 3381 53252 |
| 0.00081619720841... | 3381 53252 |
| 0.00091812309407... | 3381 53252 |
| 0.00102004897973... | 3381 53252 |
| 0.00112197486539... | 3381 53252 |
| 0.00122429581262... | 3381 53252 |
| 0.00132622169828... | 3381 53252 |
| 0.00142814758394... | 3381 53252 |
| 0.00153007346960... | 3381 53252 |
| 0.00163199935525... | 3381 53252 |
| 0.00173392524091... | 3381 53252 |
| 0.00183624618814... | 3381 53252 |
| 0.00193817207380... | 3381 53252 |
| 0.00204009795946... | 3381 53252 |
| 0.00214202384512... | 3381 53252 |
| 0.00224394973078... | 3381 53252 |
| 0.00234627067801... | 3381 53252 |
| 0.00244819656367... | 3381 53252 |
| 0.00255012244933... | 3381 53252 |
| 0.00265204833499... | 3381 53252 |
| 0.00275397422065... | 3381 53252 |
| 0.00285629516788... | 3381 53252 |
| 0.00295822105354... | 3381 53252 |

Mea_00001_ - sm_00001

| Name | Unit |
|--|------|
| <input checked="" type="checkbox"/> MainFunctionTask2_102400ns | - |
| <input checked="" type="checkbox"/> test_ods_ram1 | rpm |
| <input checked="" type="checkbox"/> test_ods_ram2 | °C |
| <input checked="" type="checkbox"/> test_ods_flag1 | - |

Generation Parameters

| Name | GenerationParameters |
|------------------|----------------------|
| MainFunctionT... | |
| test_ods_ram1 | |
| test_ods_ram2 | |
| test_ods_flag1 | |

Options

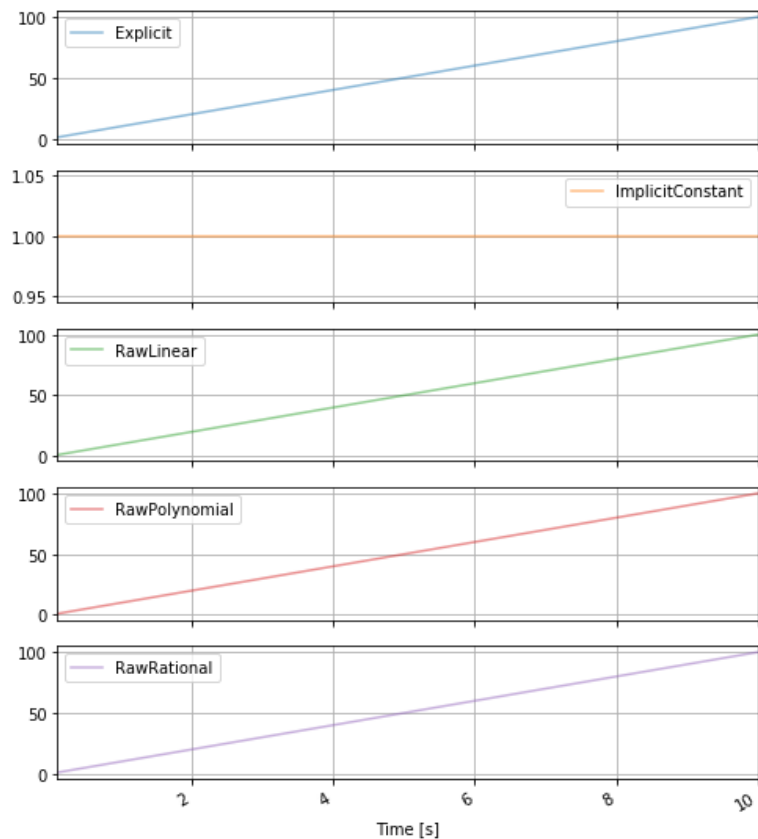
ModeEnum: MO_CALCULATED

Load(/json) Load(Protobuf) Generation Parameters Save Close

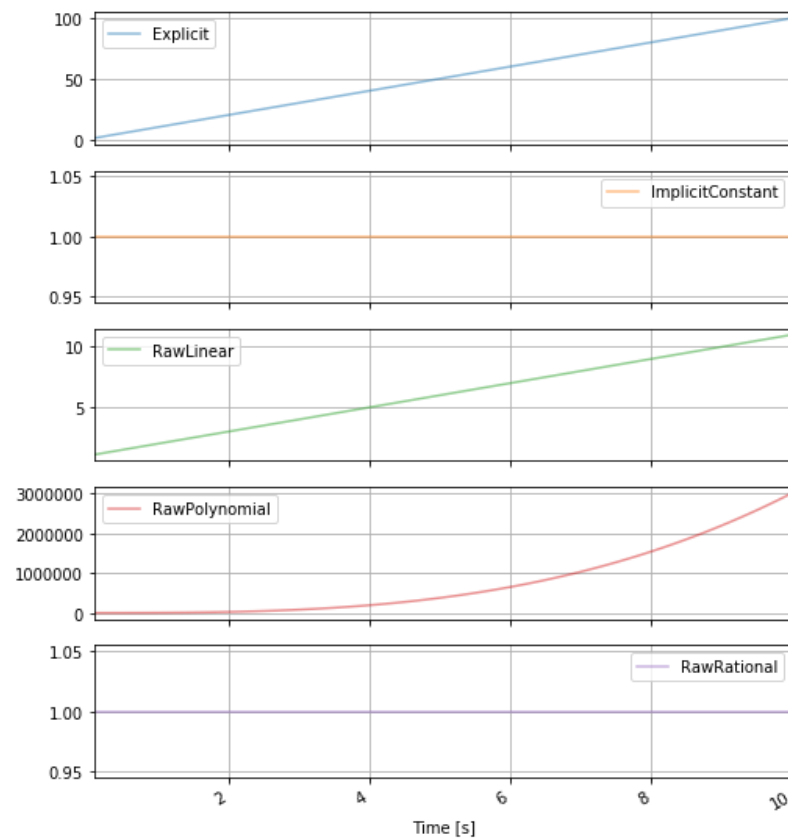
| Data | Chart |
|-----------------------|--|
| Main Function Task... | test_ods_ram1 test_ods_ram2 test_ods_flag1 |
| 0 | 2641.406 90.00977 |
| 0.00010192588565... | 2641.406 90.00977 |
| 0.00020424683289... | 2641.406 90.00977 |
| 0.00030617271854... | 2641.406 90.00977 |
| 0.00040809860420... | 2641.406 90.00977 |
| 0.00051002448986... | 2641.406 90.00977 |
| 0.00061195037552... | 2641.406 90.00977 |
| 0.00071427132275... | 2641.406 90.00977 |
| 0.00081619720841... | 2641.406 90.00977 |
| 0.00091812309407... | 2641.406 90.00977 |
| 0.00102004897973... | 2641.406 90.00977 |
| 0.00112197486539... | 2641.406 90.00977 |
| 0.00122429581262... | 2641.406 90.00977 |
| 0.00132622169828... | 2641.406 90.00977 |
| 0.00142814758394... | 2641.406 90.00977 |
| 0.00153007346960... | 2641.406 90.00977 |
| 0.00163199935525... | 2641.406 90.00977 |
| 0.00173392524091... | 2641.406 90.00977 |
| 0.00183624618814... | 2641.406 90.00977 |
| 0.00193817207380... | 2641.406 90.00977 |
| 0.00204009795946... | 2641.406 90.00977 |
| 0.00214202384512... | 2641.406 90.00977 |
| 0.00224394973078... | 2641.406 90.00977 |
| 0.00234627067801... | 2641.406 90.00977 |
| 0.00244819656367... | 2641.406 90.00977 |
| 0.00255012244933... | 2641.406 90.00977 |
| 0.00265204833499... | 2641.406 90.00977 |
| 0.00275397422065... | 2641.406 90.00977 |
| 0.00285629516788... | 2641.406 90.00977 |
| 0.00295822105354... | 2641.406 90.00977 |

変換関数への対応 デモ (ATFX)

BRIX AoSubMatrix Plot SEQREP1 - MO_STORAGE (valuematrix-read)



BRIX AoSubmatrix Plot SEQREP1 - MO_CALCULATED (valuematrix-read)



ODS 6.0 REST API データ取得時に出てきた課題

- ◆ 欧州ASAMでのODS 6.0 Evaluation中のためODSサーバ間の細かな相違があった
 - ODS仕様書を基準として各ODSサーバベンダと連携して仕様書に明記されている部分に関してのサーバ修正を行いほぼ共通のPythonスクリプトにてデータ取得できるようになりました。
- ◆ 各ODS対応を順番に行ったが、SelectStatements対応(異なるaid指定可否)の差異で調整が必要となった。
 - ODSサーバ内部設計などにより制約はあると思うので、最低限この条件での検索などはできることなどの定義が必要と思います。
- ◆ JSON形式のクライアントを先に作成したため、ProtobufのJSONシリアライズ仕様による差異を解釈するのに時間がかかりました。
 - attribute名がprotoファイルのsnake case表記(string_arrays)がProtobufのJSONシリアライズにてcamel case表記(stringArrays)に読み替える必要があるなど
- ◆ base attribute namesとderived attribute nameのどちらを使えばよいか迷う場所が多かった。
 - ほとんどがderived attribute nameを使うようにODS仕様には明記されてる。
 - この部分の各ODSサーバの解釈の差異が多かった。