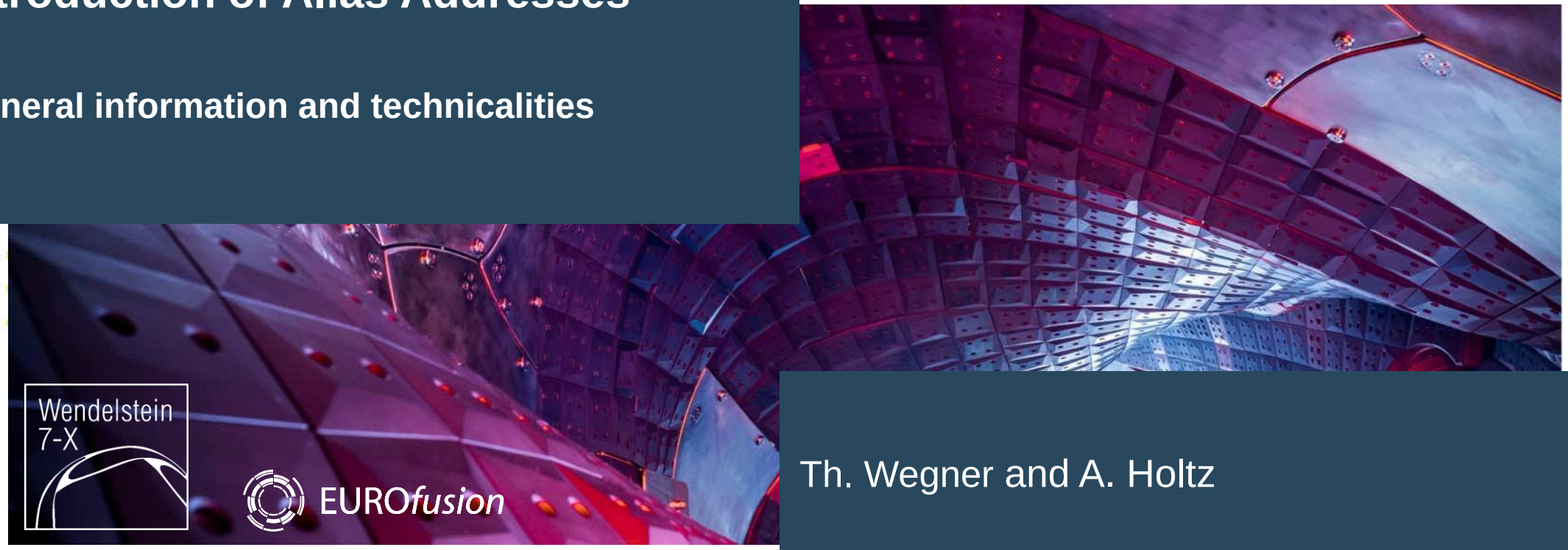
MAX-PLANCK-INSTITUT
FÜR PLASMAPHYSIK

Introduction of Alias Addresses

General information and technicalities



Th. Wegner and A. Holtz

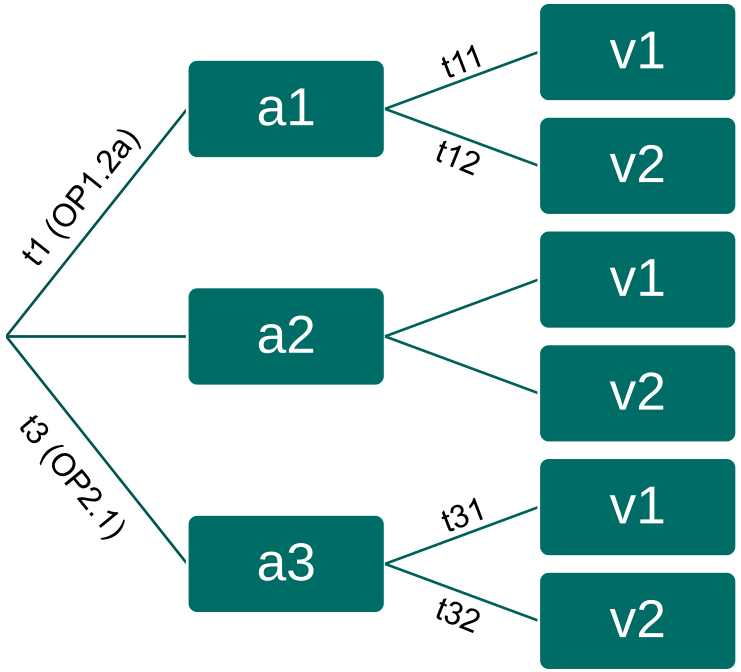
This work has been carried out within the framework of the EUROfusion Consortium, funded by the European Union via the Euratom Research and Training Programme (Grant Agreement No 101052200 — EUROfusion). Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Commission. Neither the European Union nor the European Commission can be held responsible for them.

Initial Situation and Objectives

- complicated and not intuitive archive structure/addresses
  e.g. http://archive-webapi.ipp-hgw.mpg.de/ArchiveDB/codac/W7X/ControlStation.70301/CONTROL-0_PARLOG/laser
- changing addresses between campaigns and even days
- versioning of evaluated data dependent on time, e.g. v2 for PID ≤ 20180919.48 but v3 for PID > 20180919.48
- similar data under completely different addresses
- non conclusive error handling of archive, e.g. when address is not correct or a version is missing

➢ one has to know the:
  - right address **a**
  - valid time interval **t**
  - the right version **v**

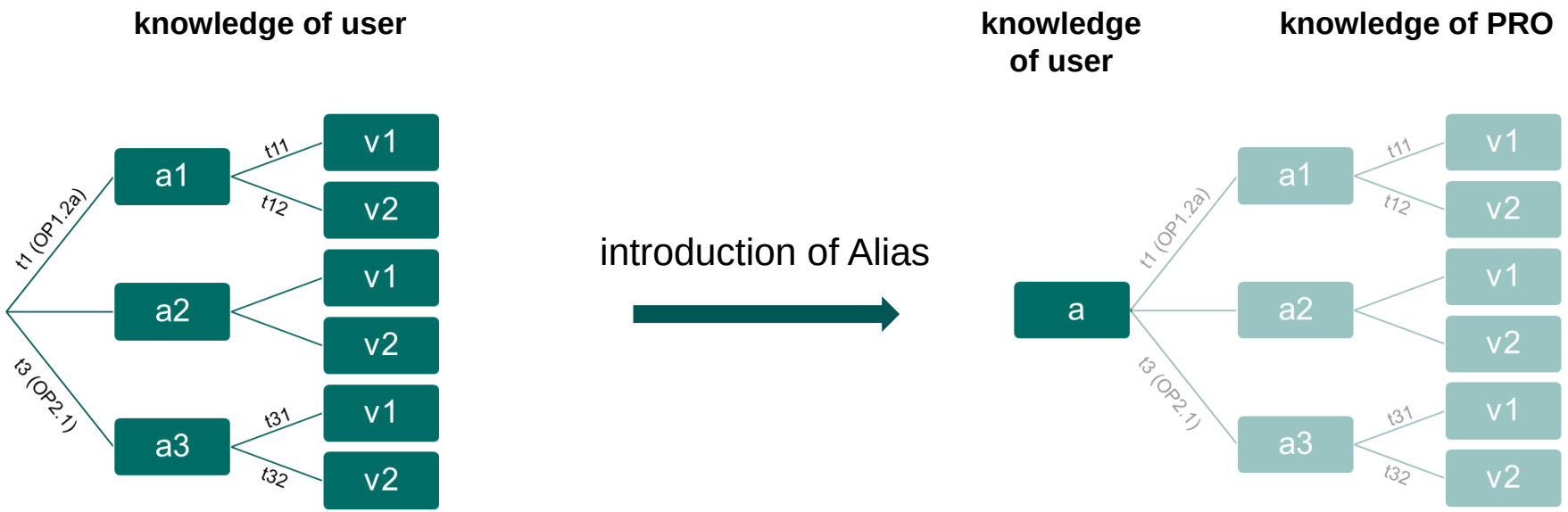
- simplify the way to get data
- reducing the chance to get wrong/not valid data
- allowing the possibility to easily update data w/o knowing meta data (**t**,**v**)



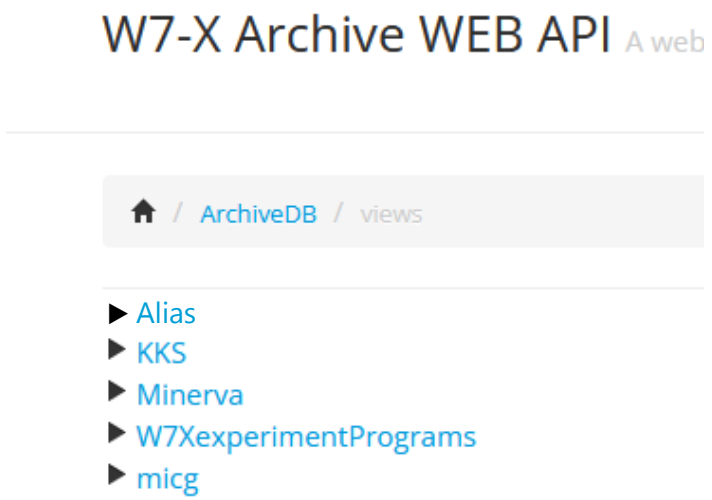
Implementation of Alias addresses

Advantages of Alias Addresses

- meta data (time interval, versions etc.) not relevant for user
- definition of alias addresses by PRO pointing to the correct address and version for each time interval
- permanent assignment of alias addresses (if nothing changes, nothing has to be done)



Structure of Alias Addresses



- location: *ArchiveDB/views/**Alias***
- structure: <***description***>_<***KKS***>_<***auto/eval***>
  - **description** short description of Data, physics quantity
  - **KKS** specification of the data source
  - **auto/eval** level data quality
- example: ArchiveDB/views/Alias/**Wdia**_**QXG**_**auto**
- similar data groups alphabetically with same description, e.g., *ECRH_CBG*, *CoilCurrents, Emission_Lines, Invasive*
- further properties have to be assigned in sub path nodes, e.g., **Wdia**_**QXG**_**auto**/1kHz

- listing of alias addresses on the logbook components page
- list of pre-defined alias addresses (will be) stored in attachment (working file, not complete)
- complete list of productive alias addresses: http://archive-webapi.ipp-hgw.mpg.de/alias.html

Some Examples

description	*ArchiveDB/views/**Alias**/*
ECRH (total) power	*ECRH_CBG_auto / Total_Power* **or** *ECRH_CBG_auto / A1_Power*
NBI total power	*NBI_CBD_eval / Total_Power* **or** *NBI_CBD_eval / S4_Power*
line integrated density	*Ndl_QMJ_auto*
(central) electron temperature	*Te_QTB_eval / Center* **and** *Te_QME_eval / Center*
Te - Profiles	*Te_QTB_eval / Profile / reff* **and** *Te_QTB_eval / Profile / value*
LBO injection time	*Invasive_LBO_QSL_auto / TimeInjection*
Pellet injection time	*Invasive_Pellets_CHE_auto / TimeInjection*
spectroscopic line	*EmissionLines_<KKS> / <ion>_<cwl>* **e.g.** *EmissionLines_QSD_eval / FeIX_10nm*
non-planar coil current	*CoilCurrents_AAE_auto / NPC_5*
gas flow rate div gas inlet	*GasInlet_QSQ_eval / FlowRate_V4*
Visible Cameras	*VisVideo_QSV_auto / AEQ10*

reff for profiles often not yet available

Path to Implementation & Action Items

- consolidation of pre-defined Alias addresses
- start with “important” plants with relevant data

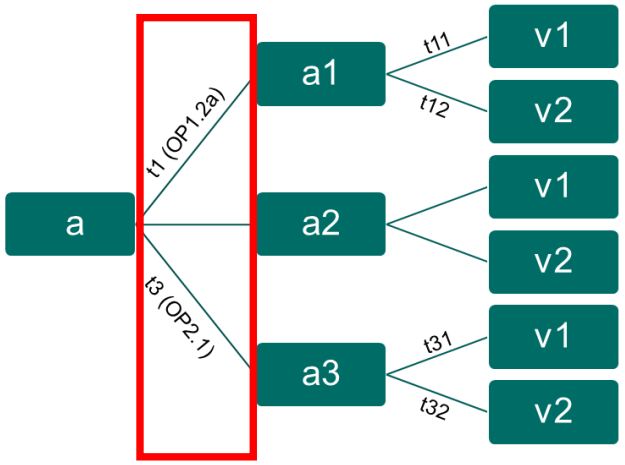
- stepwise creation of Alias Addresses also with help of A. Holtz (via ticket)
  - ☐ write ticket and/or contact ThW with tuple (address, time interval, target) information
- some data still in test archive
  - ☐ movement from test archive to productive archive (ArchiveDB) (via ticket)
- inconsistency of data shape (1D vs 2D)
  - ☐ partial re-structuring of datastreams (aim to have independent reff and Te values, different datastreams)
- traceability of data access: Which version is used for data and publications?
  - ☐ temporary solution: storage of timestamp of data access

General Concept

- **consists of**
  - **name**
    - /KKS/AAE_MainCoils/Currents_1Hz, /Minerva/Diagnostics/ECE/Calibrationvalues
  - **validity interval – time period for which the Alias is valid**
    - can be very broad (all time) or very specific (e.g. single experiment)
  - **target – any address within the same archive (Test or ArchiveDB)**
    - targets up to signal, not within a signal, e.g. 3rd element of profile can not be addressed

Usecase 1:
Adressing different streams/signals over time under the same name

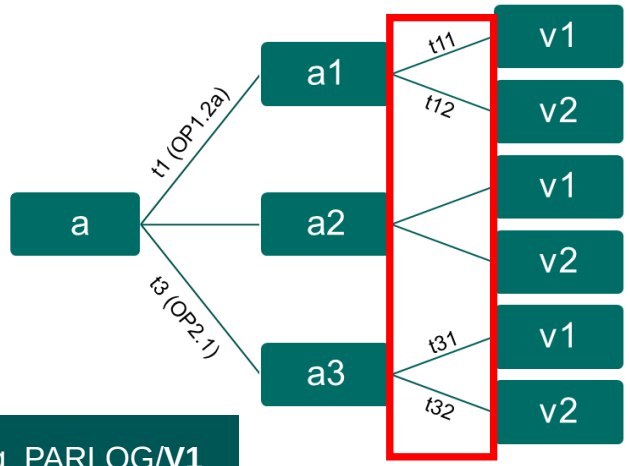
- **multiple Alias can have same name, but different targets and validity intervals (e.g. OP1.2 or OP2.1)**
- **when reading time interval appropriate Alias is selected**



/KKS/AAE_MainCoils/Currents_1Hz	**1970-01-01 00:00:00 – 2017-07-04 07:14:25**	/codac/W7X/CoDaStationDesc.84/DataReductionProcessDesc.30_DATASTREAM/
/KKS/AAE_MainCoils/Currents_1Hz	**2017-07-04 07:14:25 – 2021-04-19 00:00:00**	/codac/W7X/CoDaStationDesc.84/DataModuleDesc.21643_DATASTREAM/
/KKS/AAE_MainCoils/Currents_1Hz	**2021-04-19 00:00:00 – 2021-09-06 12:00:00**	/codac/W7X/CoDaStationDesc.84/FeedBackProcessDesc.29601_DATASTREAM/
/KKS/AAE_MainCoils/Currents_1Hz	**2021-09-06 12:00:00 – 2262-04-11 23:47:16**	/codac/W7X/ControlStation.2131/AAE_Current_1Hz_MagneticFieldDirection_DATASTREAM/

Usecase 2:
Changing Aliases

- **changing version, model or calibration data for specific time interval**
- **new Aliases have higher priority than older ones and are merged into existing ones**



Minerva/Equilibrium/ReferenceEquilibriumID --- **1970-01-01 00:00:00 – 2262-04-11 23:47:16** ---> /raw/W7XAnalysis/Equilibrium/RefEq_PARLOG/**V1**

- use RefEq_PARLOG/V2 for 2021-04-19

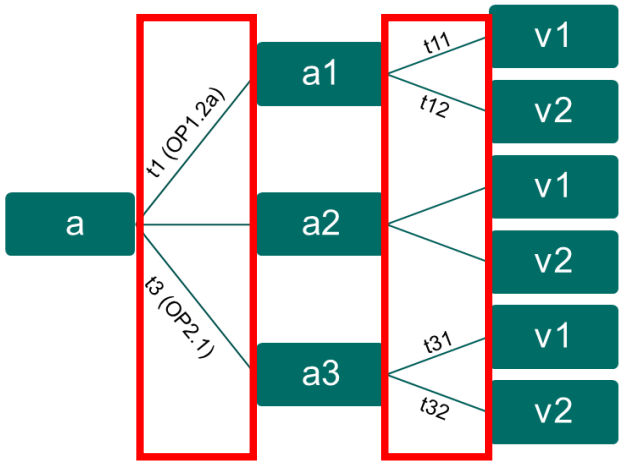
Minerva/Equilibrium/ReferenceEquilibriumID --- **1970-01-01 00:00:00 – 2021-04-19 00:00:00** ---> /raw/W7XAnalysis/Equilibrium/RefEq_PARLOG/**V1**

Minerva/Equilibrium/ReferenceEquilibriumID --- **2021-04-19 00:00:00 – 2021-04-20 00:00:00** ---> /raw/W7XAnalysis/Equilibrium/RefEq_PARLOG/**V2**

Minerva/Equilibrium/ReferenceEquilibriumID --- **2021-04-20 00:00:00 – 2262-04-11 23:47:16** ---> /raw/W7XAnalysis/Equilibrium/RefEq_PARLOG/**V1**

Usecase 3:
Deleting Aliases

- **Aliases become obsolete or mistakes occured**
- **completely delete Alias by setting target to null for full validity interval**
- **partly delete Alias by setting target to null for specific validity interval**



Minerva/Equilibrium/ReferenceEquilibriumID — **1970-01-01 00:00:00 – 2262-04-11 23:47:16** →

RefEq_PARLOG/V2 for 2021-04-19 is wrong, but V1 is also **not** correct

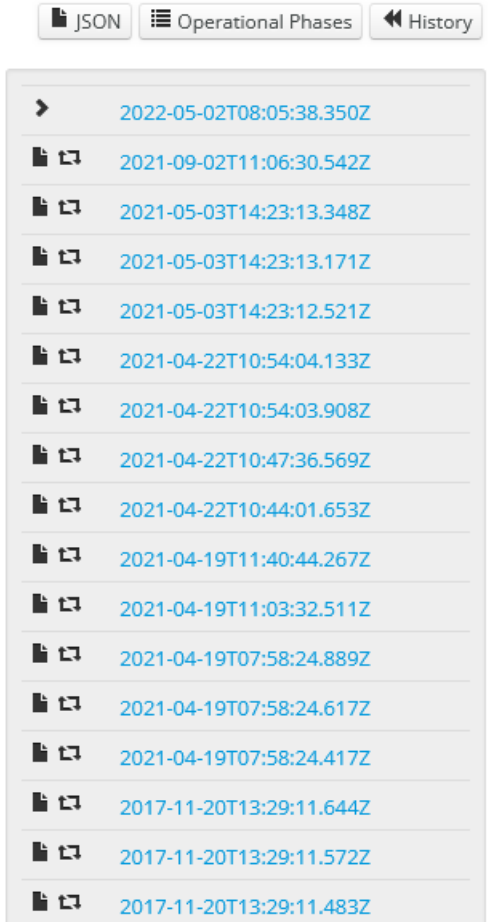
Minerva/Equilibrium/ReferenceEquilibriumID — **1970-01-01 00:00:00 – 2021-04-19 00:00:00** → /raw/W7XAnalysis/Equilibrium/RefEq_PARLOG/V1

Minerva/Equilibrium/ReferenceEquilibriumID — **2021-04-19 00:00:00 – 2021-04-20 00:00:00** →

Minerva/Equilibrium/ReferenceEquilibriumID — **2021-04-20 00:00:00 – 2262-04-11 23:47:16** → /raw/W7XAnalysis/Equilibrium/RefEq_PARLOG/V1

History

- **Usecase: what was valid at time X?**
- **all Aliases are stored and nothing is deleted**
- **with a given timestamp it is possible to find the Alias valid at that time**
- **currently no proper API support**
- **„History“ for manual browsing is accessible via http://archive-webapi.ipp-hgw.mpg.de/alias.html**



User Interfaces I – manual updates & Java

- **manually via http://archive-webapi.ipp-hgw.mpg.de/alias.html**
  - add, delete and resolve Aliases
  - view history and additional information
  - still work in progress

- **Java (part of SignalAccess)**
  - use archive.inserter.AliasInserter for creating and deleting Aliases
  - TreeLister and SignalReader transparently resolve Aliases

User Interfaces II – Python

- **send POST to http://archive-webapi.ipp-hgw.mpg.de/alias.html**
  - for details see http://archive-webapi.ipp-hgw.mpg.de/#alias

```
import urllib.request, json

def write_alias_to_sandbox():
    url = 'http://archive-webapi.ipp-hgw.mpg.de/alias.html'
    data_dict = {
        "database": "Sandbox", "alias": "AliasTest/Test/Test666",
        "target": "raw/W7X/CoDaStationDesc.10082/DataModuleDesc.10084_DATASTREAM/39/AAQ11_ActVal_Ramp",
        "from": 1707523200000000000, "upto": 1708387200000000000,
        "user": "holtza"
    }
    data = json.dumps(data_dict).encode("utf-8")
    r = urllib.request.Request(url, data, headers={"Content-type": "application/json"})
    response = urllib.request.urlopen(r)
```

- **use w7xarchive.alias from W7X_APIpy (ask K.J. Brunner for details)**
  - create and delete Alias, resolve target of given address

```
import w7xarchive.alias as alias

def write_alias_to_sandbox_usingW7X_APIPY():
    alias.create_alias(
        database="sandbox", alias="AliasTest/Test/Test888",
        target="raw/W7X/CoDaStationDesc.10082/DataModuleDesc.10084_DATASTREAM/39/AAQ11_ActVal_Ramp",
        time_from="2024-02-10 00:00:00.0", time_to="2024-02-20 00:00:00.0")
```

Action Items

- stepwise creation of Alias Addresses also with help of A. Holtz (via ticket)
  - ☐ write ticket and/or contact ThW with tuple (address, time interval, target) information
- some data still in test archive
  - ☐ movement from test archive to productive archive (ArchiveDB) (via ticket)
- inconsistency of data shape (1D vs 2D)
  - ☐ partial re-structuring of datastreams (e.g. independent reff and Te values, different datastreams)
- traceability of data access: Which version is used for data and publications?
  - ☐ temporary solution: storage of time stamp of data access