UK Atomic Energy Authority

IMAS UDA Workshop

JSON Mappings

Adam Parker, Jonathan Hollocombe, Stephen Dixon

November 21, 2023 - ITER

UK Atomic Energy Authority

Agenda

1. Data mapping with JSON
2. Hands-on 1

3. Example mapping with MAST-U
4. Hands-on 2

5. Work-site visit

CCFE CULHAM CENTRE FOR FUSION ENERGY

Outline

UK Atomic Energy Authority

1) Introduction

- Example of End Goal
- Underlying Workflow
- Motivation and History
- Aims
- Infrastructure

2) Design, Structure, + Philosophy

- JSON mapping files
  - Directory structure
- Map types
  - VALUE
  - DIMENSION
  - PLUGIN
  - EXPR
  - CUSTOM
- Mapping File Validation

3) Techniques + Tools

- JSON
- Inja for templating
- External Libraries
  - Nlohmann
  - GSL
  - ExprTK
- Validation and testing

4) Implementation

- Random examples
- DRAFT examples
- Hands-on taster

CCFE
CULHAM CENTRE FOR FUSION ENERGY

Goal : MAST-U Mappings

"Pick a card, any card"

UK Atomic Energy Authority

Select MAST-U shot

- MU01 : 43306 – 45513
- MU02 : 45514 – 47174
- MU03 : 47175 – Present

Select IDS to map

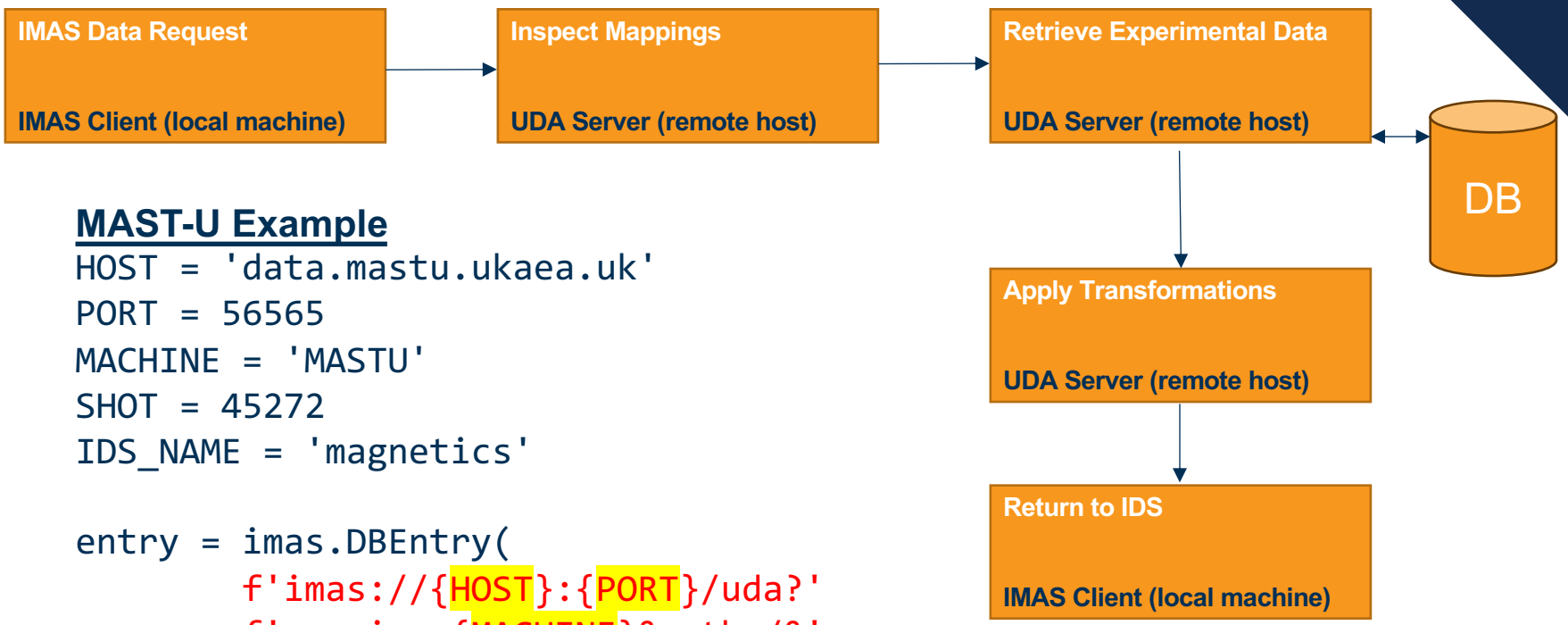
- **magnetics**
- **pf_active**
- **pf_passive**
- *wall*
- *tf*
- *pulse_schedule*

Live Demo

Wish me luck..

CCFE CULHAM CENTRE FOR FUSION ENERGY

What Is Actually Happening?



MAST-U Example

```
HOST = 'data.mastu.ukaea.uk'
PORT = 56565
MACHINE = 'MASTU'
SHOT = 45272
IDS_NAME = 'magnetics'

entry = imas.DBEntry(
        f'imas://{HOST}:{PORT}/uda?'
        f'mapping={MACHINE}&path=/&'
        f'shot={SHOT}', 'r’
)
ids = entry.get(IDS_NAME)
```

(Access Layer 5.0.0 Syntax)

imas://data.mastu.ukaea.uk:56565/uda?mapping=MASTU&path=/&shot=45272

Requires IP whitelist to externally access data.mastu.ukaea.uk

On Second thought..
I want to map JET data

UK Atomic Energy Authority

Select JET pulse

- 99518 – 99530
  (not 99524 please)

Select IDS to map

- summary,
- summary,
- summary,
- *.. or summary*

Live Demo

Example - **imas://<jet.server.jdc.uk>:<56560>/uda?mapping=JET&path=/&shot=99518**

CCFE CULHAM CENTRE FOR FUSION ENERGY

More Adventurous Example

UK Atomic Energy Authority

DRAFT data : Definitely Real And Functioning Tokamak

Synthetic/dummy data created for the purpose of mapping (more in Hands-on)


```
],
"/APF/PF_COIL_0_tnp/z/data_type": "float",
"/APF/PF_COIL_0_tnp/z/data_rank": 1,
"/APF/PF_COIL_0_tnp/width/data": [
    0.1044773547119372,
    0.936980128743582,
    0.12232255455261487,
    0.33638005069029986,
    0.502203923367993,
    0.6029004582196089,
    0.663690150629852,
    0.6505158717454598,
    0.24603096961743443,
    0.13362911260516797
],
"/APF/PF_COIL_0_tnp/width/data_type": "float",
"/APF/PF_COIL_0_tnp/width/data_rank": 1,
"/APF/PF_COIL_0_tnp/height/data": [
    0.8199911891860594,
    0.3913354569236497,
    0.4495548442164511,
    0.03338401464628249,
    0.911362444782734,
    0.9422272465629742,
    0.3299225594094807,
    0.5155881664835099,
    0.3054102722331802,
    0.3390917152215591
],
"/APF/PF_COIL_0_tnp/height/data_type": "float",
"/APF/PF_COIL_0_tnp/height/data_rank": 1,
"/APF/PF_COIL_1_tgr/current/data": [
```


Available Shots
45460, 48067, 48079
IDS to map
magnetics, pf_active, ….

Live Demo

Motivation: Mapping in the Past

UK Atomic Energy Authority

JET Mappings

JET PPF >> CPO >> IDS

Limited number of mappings (mapped by JH)

Mappings defined in XML files in two stages

— uses EXP2IMAS tool

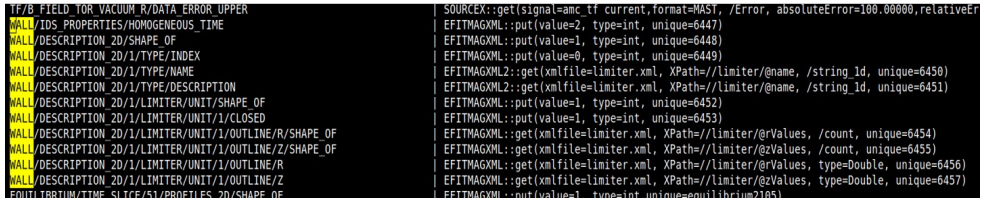
MAST Mappings

Previously used the IDAM database for mappings

IMAS IDS path mapped to an entry in the database

— Corresponds to a function call

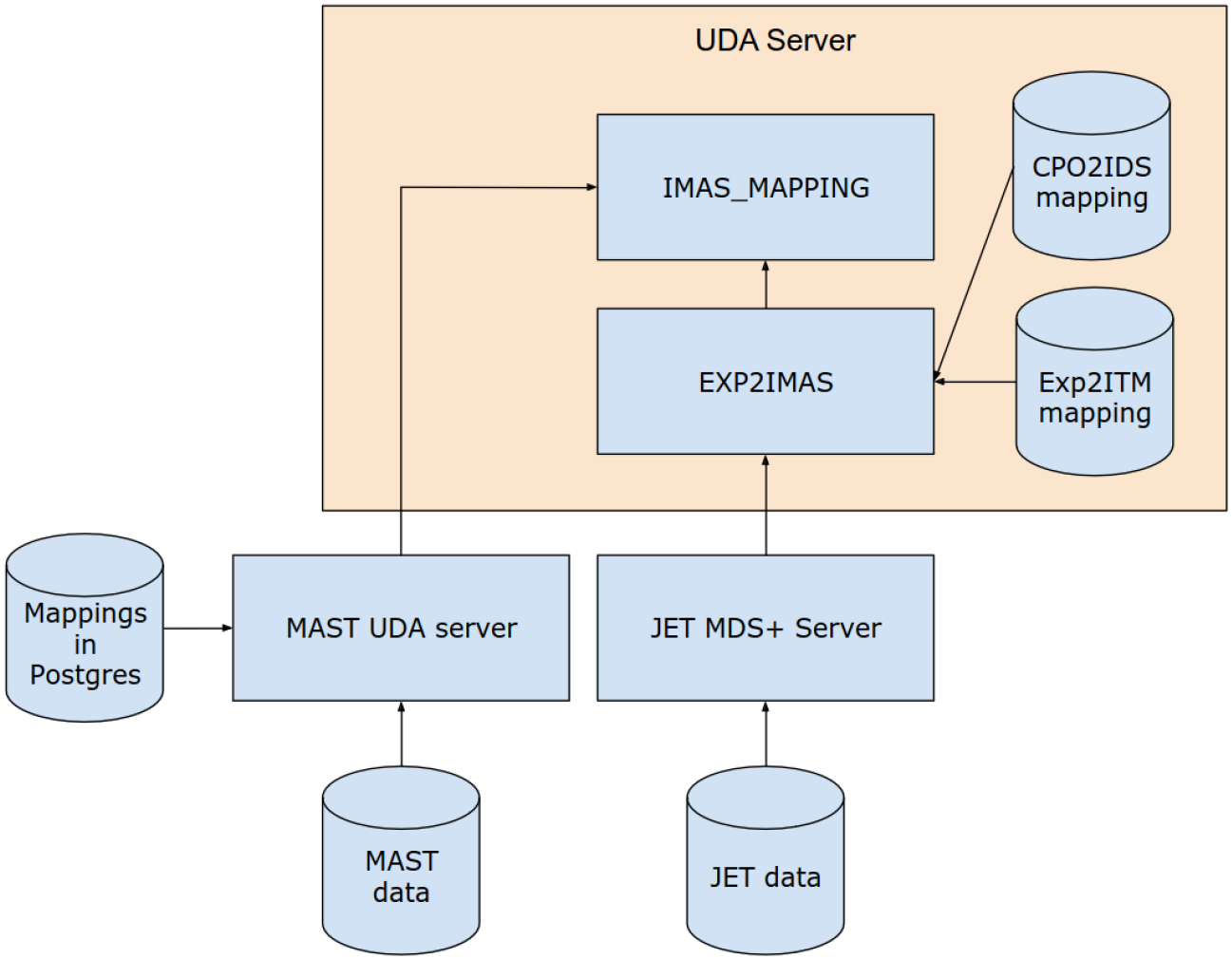
1) slow,
2) hard to maintain and update,
3) lacking features,
4) opaque



+ numerous other secret techniques people kept quiet

CCFE CULHAM CENTRE FOR FUSION ENERGY

Motivation: Mapping in the Past



Motivation: What Do We Need?

UK Atomic Energy Authority

1) **Speed!**
   Dynamic mappings on the fly, reducing storage, map in bulk

2) **Portable**
   Setup server and client easily, in theory on any machine or architecture

3) **Human Readable**
   Transparent, manageable by ROs and diagnosticians

4) **Reproducible and Self-Describing**
   Easily updateable, version controlled, available metadata and provenance, can handle iterations to the DD

5) **<u>New Development:</u> Machine Agnostic**
   Framework and functionality not specific to MAST-U

CCFE
CULHAM CENTRE FOR FUSION ENERGY

Infrastructure/Design Ideas (MAST-U)

UK Atomic Energy Authority

Backend/Plugin (Open Repo)

- **Reroutes IDS/IMAS to relevant point**
  Diverts/routes requests and forward to appropriate
- **Loads experiment mappings**
  Parse and manipulate JSON objects
- **Written in C++ (interface to legacy C)**
  Utilise UDA backend
- **Interface with UDA**
  Tool already used for MAST-U data access
- **Handle**:
  - **experimental data retrieval**,
    (NetCDF, geometry XMLs, images)
  - **type conversion,**
    (IMAS types *int, double, complex*)
  - **de-serialisation, tree traversal,**
    (data often stored in complex structures)
  - **and data output.**
    (appropriately place on the DataBlock)

Mapping Files (Machine Specific Repo)

- **Mappings from MAST-U signal to DD entry defined in mapping files per IDS**
- **Human readable (JSON)**
- **Managed by ROs and experts with no interface with the server or backend**
- **Handles multiple DD versions and updates**

...not complicated at all

CCFE CULHAM CENTRE FOR FUSION ENERGY

Mapping Files: JSON

JSON stands for **J**ava**S**cript **O**bject **N**otation

A lightweight format for storing and transporting data (used in web servers)

JSON is "self-describing" and easy to understand (compared to XML in my opinion)

```
{
    "name": "Adam Parker",
    "profession": "Data Manager",
    "age": 32,
    "address": {
        "city": "Oxford",
        "postal_code": "OX123456",
        "country": "England"
    },
    "languages": ["C++", "Python", "JSON"],
    "contact": [
        {
            "name": "Twitter",
            "link": "https://X.com/donthaveone"
        },
        {
            "name": "email",
            "link": "mailto:adam.parker@ukaea.uk"
        }
    ]
}
```

Data is stored in key-value pairs

Multiple data types:

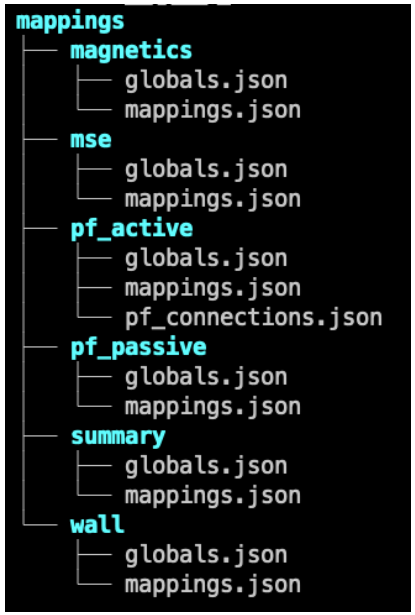
- strings,
- ints,
- floats,
- arrays,
- objects,
- etc.

Mapping Files: Directory Structure

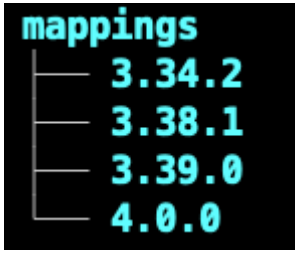
UK Atomic Energy Authority

Syntax / Naming / Helper file
<ids name>/globals.json
+
JSON Mapping File
<ids name>/mappings.json

Note, a top-level non-ids specific globals.json is included to avoid duplication



Mappings are separated by DD version
Version specified by the IMAS client requesting the given IDS decides which mappings are used



Example DD 3.X.X → 4.0.0:
- `global_quantities/li` → `li_3`,
- `local/magnetic_axis/b_field` → `b_field_tor`
- `distribution/global_quantities/thermalisation` → `thermalization`

CCFE CULHAM CENTRE FOR FUSION ENERGY

Mapping Files: *Top-level* globals.json

UK Atomic Energy Authority

```
{
    ...
    ”UNIT_SF”: 1000.
    “COMMENT”: ”Random comment description about the
mappings file of this machine, if necessary”,
    “DEG2RAD”: 0.0174532925199.
    "PLUGIN_ARGS": {
        “DRAFT_JSON”: {
            "source": "{{ shot }}",

        },
        "UDA": {
            "source": "{{ shot }}",
            "host": "uda2.hpc.l",
            "port": "56565"
        }
    }
    ...
}
```

PLUGIN_ARGS keyword object used to define extra required fields for each plugin, these are forwarded on to the server

Variables in here can be used for templated by any IDS

CCFE CULHAM CENTRE FOR FUSION ENERGY

Mapping Files: *Per IDS* globals.json

UK Atomic Energy Authority

```
{
    ...
    "COIL_NAMES_GEOM": [
        "d1-upper", "d2-upper", "d3-upper", "d5-upper", "d6-upper", "d7-upper", "dp-upper",
        "p4-upper", "p5-upper", "p6-upper", "px-upper", "d1-lower", "d2-lower", "d3-lower",
        "d5-lower", "d6-lower", "d7-lower", "dp-lower", "p4-lower", "p5-lower", "p6-lower",
        "px-lower", "p1-inner", "p1-outer", "pc"
    ],
    "COIL_NAMES": [
        "D1U", "D2U", "D3U", "D5U", "D6U", "D7U", "DPU", "P4U",
        "P5U", "P6U", "PXU", "D1L", "D2L", "D3L", "D5L", "D6L",
        "D7L", "DPL", "P4L", "P5L", "P6L", "PXL"
    ],
    "UNIT_SF": 1000.0,
    ...
}
```

Global keys, values, and names unique to each experiment/IDS
(– Used in templating and expression evaluation)

MAST-U real example
COIL_NAMES used to access signal : ***/AMC/FLUX_LOOPS/D1U***

MAST-U currents are also in kAmps so are scaled by **UNIT_SF** for the IDS

CCFE CULHAM CENTRE FOR FUSION ENERGY

Mapping Files: Map Types

UK Atomic Energy Authority

Strategy

Mapping type / transformations: (~90% of cases)

- Value mapping
- Direct unmodified mapping
- Simple mathematical operation (+/× by float)
- Slice/Concatenate/Transpose
- Slightly more complicated mathematical expression

... and then 10% of *'cross that bridge when we get to it'*

Aims of strategy

- common framework,
- reduce code duplication,
- stop unnecessary machine specific operations

"Fine line between mapping, post-processing, and analysis"

- Wise Data Manager, 2023
Beginners Guide to IMAS Mapping

CCFE
CULHAM CENTRE FOR FUSION ENERGY

Mapping Types: VALUE

UK Atomic Energy Authority

```
{
    ...
    "ids_properties/homogeneous_time": {
        "MAP_TYPE": "VALUE",
        "VALUE": 0
    },
    ...
    "test_signal/value_map_float": {
        "MAP_TYPE": "VALUE",
        "VALUE": 4.5
    },
    ...
    "test_signal/value_map_string": {
        "MAP_TYPE": "VALUE",
        "VALUE": "Hello World"
    },
    ...
    "test_signal/value_map_array": {
        "MAP_TYPE": "VALUE",
        "VALUE": [1, 2, 3, 4, 5]
    },
    ...
}
```

Very simple, hard-coded value, retrieved from mapping file on request

JSON library handles implicit type deduction

(Please get in touch if type not supported)

CCFE CULHAM CENTRE FOR FUSION ENERGY

Mapping Types: DIMENSION

```
{
    ...
    "coil/element": {
        "MAP_TYPE": "DIMENSION",
        "DIM_PROBE": "_random/data"
    },
    ...
    "_random/data": {
        "MAP_TYPE": "VALUE",
        "VALUE": [1, 6, 3, 2, 10]
    },
    ...
}
```

Reference another mapping within the file (using DIM_PROBE)

and take the size of the returned data

In this simple case
coil/element would return ***5***

This mapping type is mostly used when the probe is actual data retrieval

E.g. on MAST-U
R positions of pf_active coil elements are stored as an array for the number of elements

Mapping Types: PLUGIN

```
{
    ...
    "b_field_pol_probe[#]/field": {
        "MAP_TYPE": "PLUGIN",
        "PLUGIN": "UDA",
        "ARGS": {
            "signal": "/AMB/PICKUP/{{ BPOL_NAME }}"
        }
    },
    ...
    "random_data/signal/current": {
        "MAP_TYPE": "PLUGIN",
        "PLUGIN": "MY_DATA_READER",
        "ARGS": {
            "name": "/MY_DATA/SIGNAL",
            "key": "current"
        }
    },
    ...
}
```

MAST-U - Example

IDS field

```
b_field_pol_probe[1]/field
```

MAST-U UDA plugin request

```
UDA::get(signal=/AMB/PICKUP/B_BL1_N04, ...)
```

(More on templating later)

Random Data - Example

IDS field

```
random_data/signal/current
```

Plugin request

```
MY_DATA_READER::get(
    name=/MY_DATA/SIGNAL,
    key=current,
    …
)
```

Map type to generate plugin requests to return relevant data
For MAST-U this could be UDA but also could be your data read plugin

Mapping Types: PLUGIN Extensions

```
{
    ...
    "b_field_pol_probe[#]/field": {
        "MAP_TYPE": "PLUGIN",
        "PLUGIN": "UDA",
        "ARGS": {
            "signal": "/AMB/PICKUP/{{ BPOL_NAME }}"
        },
        "OFFSET": 4.5
    },
    ...
     "ip[#]": {
        "MAP_TYPE": "PLUGIN",
        "PLUGIN": "UDA",
        "ARGS": {
            signal": "/AMC/PLASMA_CURRENT"
        },
        "SCALE": 1000.0
    },
    ...
}
```

MAST-U UDA plugin request

`UDA::get(signal=/AMB/PICKUP/B_BL1_N04, ...)`

Data will be offset by 4.5
Scalar signal: 2.0 - returns: 6.5
Array signal: [1.3, 4.5, 3.3] - returns [5.8, 9.0, 7.8]

MAST-U UDA plugin request

`UDA::get(signal=/AMC/PLASMA_CURRENT, ...)`

Data to be scaled by 1000
Works with both scalar and array data (1D)
MAST-U plasma current (kAmps) x 1000
--- IDS plasma current (Amps)

Simple mathematical operation extension to PLUGIN map type

Note: can be combined

Mapping Types: PLUGIN Extensions

```
{
    ...
    "b_field_pol_probe[#]/field": {
        "MAP_TYPE": "PLUGIN",
        "PLUGIN": "UDA",
        "ARGS": {
            "signal": "/AMB/PICKUP/{{ BPOL_NAME }}"
        },
        "SLICE": "[1]" (string type)
    },
    ...
     "adams_function_example/current": {
        "MAP_TYPE": "PLUGIN",
        "PLUGIN": "MY_PLUGIN",
        "ARGS": {
            signal": "/ADAM/PLASMA_CURRENT"
        },
        "FUNCTION": "my_func"
    },
    ...
}
```

SLICE – Work in Progress

```
UDA::get(signal=/AMB/PICKUP/B_BL1_N04, …)[1]
```

Slice functionality appended to request
Relies on UDA server-side slicing and user knowledge of requested data and dimensions (tests ongoing)

Slice to get 2nd element
Array signal: [1.3, 4.5, 3.3] - returns 4.5

FUNCTION

```
MY_PLUGIN::my_func(
   signal=/ADAM/PLASMA_CURRENT, …
)
```

Default PLUGIN function is *'get'*
FUNCTION variable if present described a separate function within your plugin to call

Simple mathematical operation extension to PLUGIN map type
Note: can be combined

Mapping Type: EXPR

```
{
    ...
    "b_field_pol_probe[#]/poloidal_angle": {
        "MAP_TYPE": "EXPR",
        "PARAMETERS": {
            "Z": "_pickup[#]/unit_vector/Z",
            "R": "_pickup[#]/unit_vector/R"
        },
        "EXPR": "2*PI-atan2(Z,R)"
    },
    ...
     "_pickup[#]/unit_vector/Z": {
        "MAP_TYPE": "PLUGIN",
        ___
     },
     "_pickup[#]/unit_vector/R": {
        "MAP_TYPE": "PLUGIN",
        ___
     },
    ...
}
```

EXPR
`"b_field_pol_probe[4]/poloidal_angle"`

Retrieves _pickup[4]/unit_vector/Z assigns to Z

Retrieves _pickup[4]/unit_vector/R assigns to R

Evaluates 2*PI – atan2(Z/R)

Returns result to data_block and IMAS

Implementation of third-party C++ Mathematical Expression Toolkit
Very powerful manipulation of data, many preset functions available

Mapping Types: CUSTOM

```
{
    ...
    "custom/concat_example": {
        "MAP_TYPE": "CUSTOM",
        "CUSTOM_TYPE": "CONCAT",
        "ARGS": {
            "example/unit_vector/A",,
            "example/unit_vector/B",
            "example/unit_vector/C"

        }
    },
    ...
     "example/unit_vector/A": {
        "MAP_TYPE": "PLUGIN",
        ——
     },
     "example/unit_vector/B": {
        "MAP_TYPE": "PLUGIN",
        ——
     },
     "example/unit_vector/C": {
        "MAP_TYPE": "PLUGIN",
        ——
     },
    ...
}
```

CUSTOM::CONCAT – Work in Progress

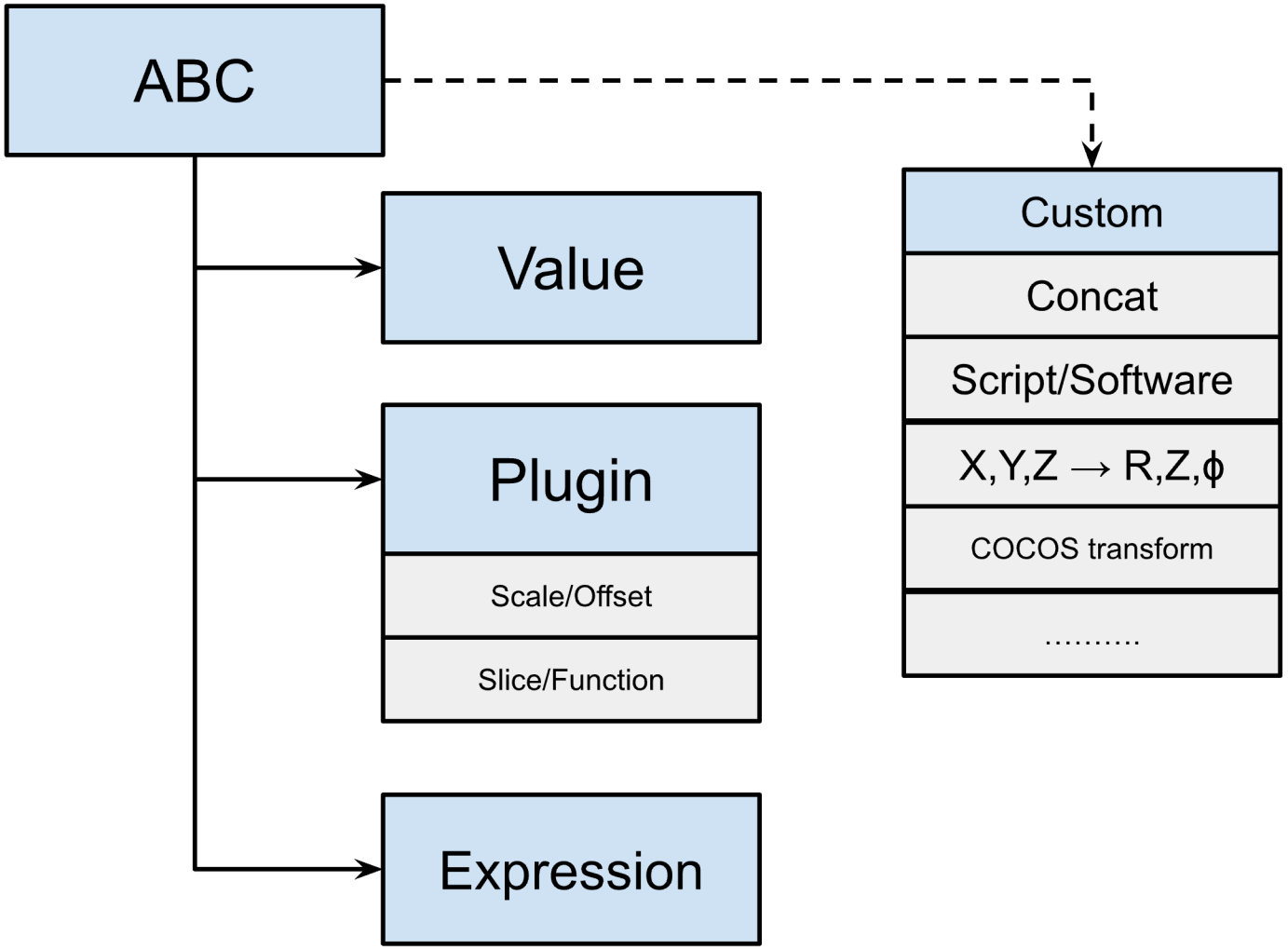
`"custom/concat_example"`

A: [1, 2, 3]
B: [4, 5, 6]
C: [7, 8, 9]

Returns to caller
[1, 2, 3, 4, 5, 6, 7, 8, 9]

Custom Function Library for all machines?

Custom Function Library



Tools: *Inja* for Dynamic Templating

UK Atomic Energy Authority

Inja is a template engine for modern C++, loosely inspired by **jinja** for python (syntax also commonly used in static web development)

Simple Example

```
name = "World";
post_inja_string = inja::render("Hello {{ name }}!", name);
-> "Hello World!"
```

JSON Mapping Example

```
"SPECIAL_COIL": "P1",
"COIL_NAMES": ["D1U", "D2U", "D3U"],
...
"signal": "/AMC/PF_COIL/{{ SPECIAL_COIL }}",
-> "/AMC/PF_COIL/P1"
…
"signal_2": "/AMC/PF_COIL/{{ at(COIL_NAMES, 2) }}"
-> "/AMC/PF_COIL/D3U"

"VALUE": "{{ length(COIL_NAMES) }}"
-> "3" -> can be converted to Integer within plugin
```

Plus, many more complicated examples

Used extensively in MAST-U Mappings
Able to dynamically substitute strings

CCFE CULHAM CENTRE FOR FUSION ENERGY

Useful Links

C++ Resources (including C++17)

Nlohmann JSON for Modern C++
Documentation - Repository

Inja - Template engine for modern C++, loosely inspired by jinja
Documentation - Repository

ExprTk - The C++ Mathematical Expression Toolkit Library
Documentation - Repository

GSL - Guidelines Support Library
Many implementations, Microsoft/GSL or header-only gsl-lite

Boost - Boost C++ Libraries
Documentation

Hidden Mappings and Comments

UK Atomic Energy Authority

Hidden Mappings

Not every entry within the mapping file has to be an IDS path to be requested

Any entry can be added, used for intermediate calculations or *'turned off'*

The accepted convention is to prefix keys by an underscore '_'

Comments

JSON by default does not allow comments

The file would be invalidated as this is not valid JSON syntax

Within the mapping file, each map type is allowed a COMMENT field.

This is purely for the mapper, the field is ignored when the mappings are ingested

```
{
    ...
    "_b_field_pol_probe[#]/poloidal_angle": {
        "MAP_TYPE": "EXPR",
        "PARAMETERS": {
            "Z": "_pickup[#]/unit_vector/Z",
            "R": "_pickup[#]/unit_vector/R"
        },
        "EXPR": "2*PI-atan2(Z,R)"
    },
    ...
     "_hidden_signal": {
        "MAP_TYPE": "PLUGIN",
        "PLUGIN": "UDA",
        "ARGS": {
            signal": "/AMC/PLASMA_CURRENT"
        }
    },
     "_pickup[#]/unit_vector/R": {
        "MAP_TYPE": "VALUE",
        "VALUE": 23.4,
        "COMMENT": "This is a random comment"
     },
    ...
}
```


CCFE CULHAM CENTRE FOR FUSION ENERGY

Structure Size & Number of Elements

UK Atomic Energy Authority

Within the DD, there exists a ‘struct_array’, where it expands into a structure (below) distinguished by `i1`

When requesting an IDS through IMAS, the imas_plugin will request the toplevel path to get the number of elements

E.g. when getting the `flux_loop(i1)` below, IMAS will request `/magnetics/flux_loop` for example..

Then it will know the number of subsequent requests to iterate over
`/magnetics/flux_loop[0]/name`
`/magnetics/flux_loop[1]/name`
`/magnetics/flux_loop[2]/name`

This can be mapped and is where DIMENSION mapping type comes in handy, or you could template using the length from globals.json
`VALUE”{{ length(my_array) }}”`

▶ flux_loop(i1)	Flux loops; partial flux loops can be described	struct_array [max_size=200 (limited in MDS+ backend only)]	1- 1...N

CCFE CULHAM CENTRE FOR FUSION ENERGY

Mapping Schema and Validation

UK Atomic Energy Authority

Testing and validation step available through pytest (and CI)

1) **JSON Verification**
   Test that all JSON globals.json and mappings.json are valid JSON syntax and can be read / parsed

2) **JSON Schema Validation**
   Validate all mappings.json against the top-level schema defining map types and required fields

3) **Inja Templating Render**
   Using globals.json, verify that all Inja templating strings complete and render to a valid string

4) **EXPR Map Type Validation**
   Test EXPR strings, substitute random floats to the parameters, check the output is sensible and of float type

CCFE
CULHAM CENTRE FOR FUSION ENERGY

Mapping Schema and Validation

UK Atomic Energy Authority

Map Type	Required Fields	Type	Optional Fields	Type
VALUE	**MAP_TYPE**	enum	COMMENT	string
	VALUE	string		
DIMENSION	**MAP_TYPE**	enum	COMMENT	string
	DIM_PROBE	string		
PLUGIN	**MAP_TYPE**	enum	COMMENT	string
	PLUGIN	string	OFFSET	float
	ARGS	object (string)	SCALE	float
			SLICE	string
			FUNCTION	string
EXPR	**MAP_TYPE**	enum	COMMENT	string
	PARAMETERS	object (string)		
	EXPR	string		
CUSTOM	**MAP_TYPE**	enum	COMMENT	string
	CUSTOM_TYPE	string	TBD	TBD

CCFE CULHAM CENTRE FOR FUSION ENERGY

Mapping Schema and Validation

```
tests/test_expr_dependents.py::test_cexprtk_wrapper PASSED
tests/test_expr_dependents.py::test_cexprtk_throw[(A+B)*C-dependents0-expected0] PASSED
tests/test_expr_dependents.py::test_cexprtk_throw[(A+B)*C-dependents1-expected1] PASSED
tests/test_expr_dependents.py::test_cexprtk_throw[-dependents2-expected2] PASSED
tests/test_expr_dependents.py::test_json_expr_strings_valid[retrieve_expr_and_dependents0] PASSED
tests/test_expr_dependents.py::test_json_expr_strings_valid[retrieve_expr_and_dependents1] PASSED
tests/test_expr_dependents.py::test_json_expr_strings_valid[retrieve_expr_and_dependents2] PASSED
tests/test_expr_dependents.py::test_json_expr_strings_valid[retrieve_expr_and_dependents3] PASSED
tests/test_expr_dependents.py::test_json_expr_strings_valid[retrieve_expr_and_dependents4] PASSED
tests/test_expr_dependents.py::test_json_expr_strings_valid[retrieve_expr_and_dependents5] PASSED
tests/test_expr_dependents.py::test_json_expr_strings_valid[retrieve_expr_and_dependents6] PASSED
tests/test_expr_dependents.py::test_json_expr_strings_valid[retrieve_expr_and_dependents7] PASSED
tests/test_expr_dependents.py::test_json_expr_strings_valid[retrieve_expr_and_dependents8] PASSED
tests/test_expr_dependents.py::test_json_expr_strings_valid[retrieve_expr_and_dependents9] PASSED
tests/test_schema.py::test_valid_globals[map_globals0] PASSED
tests/test_schema.py::test_valid_globals[map_globals1] PASSED
tests/test_schema.py::test_valid_globals[map_globals2] PASSED
tests/test_schema.py::test_valid_globals[map_globals3] PASSED
tests/test_schema.py::test_valid_globals[map_globals4] PASSED
tests/test_schema.py::test_valid_globals[map_globals5] PASSED
tests/test_schema.py::test_valid_globals[map_globals6] PASSED
tests/test_schema.py::test_valid_globals[map_globals7] PASSED
tests/test_schema.py::test_valid_globals[map_globals8] PASSED
tests/test_schema.py::test_valid_globals[map_globals9] PASSED
tests/test_schema.py::test_valid_structure[map_schema0] FAILED
tests/test_schema.py::test_valid_structure[map_schema1] PASSED
tests/test_schema.py::test_valid_structure[map_schema2] PASSED
tests/test_schema.py::test_valid_structure[map_schema3] PASSED
tests/test_schema.py::test_valid_structure[map_schema4] PASSED
tests/test_schema.py::test_valid_structure[map_schema5] PASSED
tests/test_schema.py::test_valid_structure[map_schema6] PASSED
tests/test_schema.py::test_valid_structure[map_schema7] PASSED
tests/test_schema.py::test_valid_structure[map_schema8] PASSED
tests/test_schema.py::test_valid_structure[map_schema9] PASSED
tests/test_templating_inja.py::test_cython_include PASSED
tests/test_templating_inja.py::test_template_syntax_fail[/magnetics/pfcoil/{{indices.1/test-expected0] PASSED
tests/test_templating_inja.py::test_template_syntax_fail[/magnetics/pfcoil/{{unknown}}/test-expected1] PASSED
tests/test_templating_inja.py::test_templating_in_json[get_keys_and_globals0] FAILED
tests/test_templating_inja.py::test_templating_in_json[get_keys_and_globals1] PASSED
tests/test_templating_inja.py::test_templating_in_json[get_keys_and_globals2] PASSED
tests/test_templating_inja.py::test_templating_in_json[get_keys_and_globals3] PASSED
tests/test_templating_inja.py::test_templating_in_json[get_keys_and_globals4] PASSED
tests/test_templating_inja.py::test_templating_in_json[get_keys_and_globals5] PASSED
tests/test_templating_inja.py::test_templating_in_json[get_keys_and_globals6] PASSED
tests/test_templating_inja.py::test_templating_in_json[get_keys_and_globals7] PASSED
tests/test_templating_inja.py::test_templating_in_json[get_keys_and_globals8] PASSED
tests/test_templating_inja.py::test_templating_in_json[get_keys_and_globals9] PASSED
tests/test_validator.py::test_json_open PASSED
tests/test_validator.py::test_json_open_fail PASSED
tests/test_validator.py::test_json_schema_pass PASSED
tests/test_validator.py::test_json_schema_fail PASSED
tests/test_validator.py::test_json_schema_comp PASSED
```

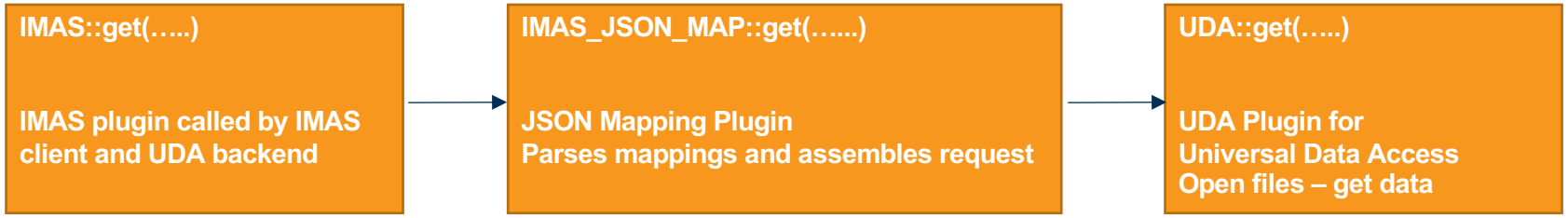
JSON Mapping Plugin

UK Atomic Energy Authority

Nothing special or revolutionary happening within the JSON Mapping Plugin and the surrounding framework

Majority of situations, requests are just being redirected to the relevant data access plugin with the useful information needed for the data retrieval (+ then apply small corrections)

Majority of MAST-U Mappings



CCFE CULHAM CENTRE FOR FUSION ENERGY

Cheat Sheet

```
{
    ...
    "ids_properties/homogeneous_time": {
        "MAP_TYPE": "VALUE",
        "VALUE": 0
    },
    ...
    "test_signal/value_map_float": {
        "MAP_TYPE": "VALUE",
        "VALUE": 4.5
    },
    ...
    "test_signal/value_map_string": {
        "MAP_TYPE": "VALUE",
        "VALUE": "Hello World"
    },
    ...
    "test_signal/value_map_array": {
        "MAP_TYPE": "VALUE",
        "VALUE": [1, 2, 3, 4, 5]
    },
    ...
}
```

```
{
    ...
    "b_field_pol_probe[#]/field": {
        "MAP_TYPE": "PLUGIN",
        "PLUGIN": "UDA",
        "ARGS": {
            "signal": "/AMB/PICKUP/{{ BPOL_NAME }}"
        },
        "SLICE": "[1]”,
        “FUNCTION”: “my_function”,
        “OFFSET”: 4.5,
        “SCALE”: 2.0
    },
    ...

     "b_field_pol_probe[#]/poloidal_angle": {
        "MAP_TYPE": "EXPR",
        "PARAMETERS": {
            ”X": "_pickup[#]/unit_vector/Z",
            ”Y": "_pickup[#]/unit_vector/R"
        },
        "EXPR": "2*X + Y"
    },
     ...
    "coil/element": {
        "MAP_TYPE": "DIMENSION",
        "DIM_PROBE": "_random/data"
    },

}
```

Thanks for listening
Questions / Comments?

Start Day 2 Hands-on 1