RDM at project Apollo:

Margaret Hamilton, lead programmer of the Apollo Guidance Computer

Image: Wikimedia, https://commons.wikimedia.org/wiki/File:Margaret_Hamilton_-_restoration.jpg

Apollo 11 git

master

Go to file t

- R31.agc
- R60_62.agc
- RCS-CSM_DAP_EXECUTIVE_PROG...
- RCS-CSM_DIGITAL_AUTOPILOT.agc
- README.md
- REENTRY_CONTROL.agc
- RESTARTS_ROUTINE.agc
- RESTART_TABLES.agc
- RT8_OP_CODES.agc
- S-BAND_ANTENNA_FOR_CM.agc
- SERVICER207.agc
- SERVICE_ROUTINES.agc
- SINGLE_PRECISION_SUBROUTINE...
- STABLE_ORBIT.agc
- STAR_TABLES.agc
- SXTMARK.agc

Apollo-11 / Comanche055 / **REENTRY_CONTROL.agc** ↑ Top

Code | Blame | 1609 lines (1356 loc) · 30.8 KB | Code 55% faster with GitHub Copilot | Raw

```
25  #
26  #       Assemble revision 055 of AGC program Comanche by NASA
27  #       2021113-051.  10:28 APR. 1, 1969
28  #
29  #       This AGC program shall also be referred to as
30  #                       Colossus 2A
31
32  # Page 844
33  # ENTRY INITIALIZATION ROUTINE
34  # ---------------------------
35
36                  BANK    25
37                  SETLOC  REENTRY
38                  BANK
39
40                  COUNT*  $$/ENTRY
41                  EBANK=  RTINIT
42
43  EBENTRY         =       EBANK7
44  EBAOG           EQUALS  EBANK6
45  NTRYPRIO        EQUALS  PRIO20          # (SERVICER)
46  CM/FLAGS        EQUALS  STATE +6
47
48  STARTENT        EXIT                    # MM = 63
49
50                                          # COME HERE FROM CM/POSE.  RESTARTED IN CM/POSE.
51                  CS      ENTMASK         # INITIALIZE ALL SWITCHES TO ZERO
52                                          # EXCEPT LATSW, ENTRYDSP, AND GONEPAST.
```

Homer: crawl metadata

Marge: share data and work with colleagues

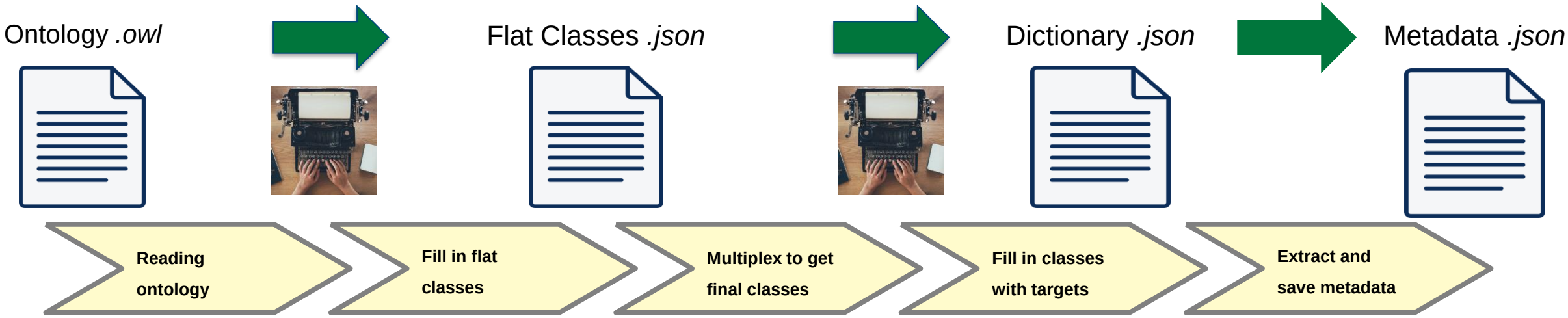
TUM Metadata Crawler: HOMER
HPMC tool for **O**ntology-based **M**etadata **E**xtraction and **R**e-use

* In order to enhance the FAIR principles (ensuring data is Findable, Accessible, Interoperable, and Reusable), researchers should effectively oversee the data they generate throughout the different stages of the research journey.
* A step in this direction is classifying produced data by producing metadata (i.e. data about data) to be attached each generated dataset, be it raw or post-processed
* If done manually, this job becomes easily burdensome and time-consuming with the increase of the produced data
* To relief scientists from this pain, a specific tool (for the moment called “Crawler”) is being developed at TUM-AER with the purpose of **automated extraction of metadata** from selected data files
* Currently, the Crawler is in a quite advanced phase and **almost ready for release**

TUM HOMER Crawler: Working principles

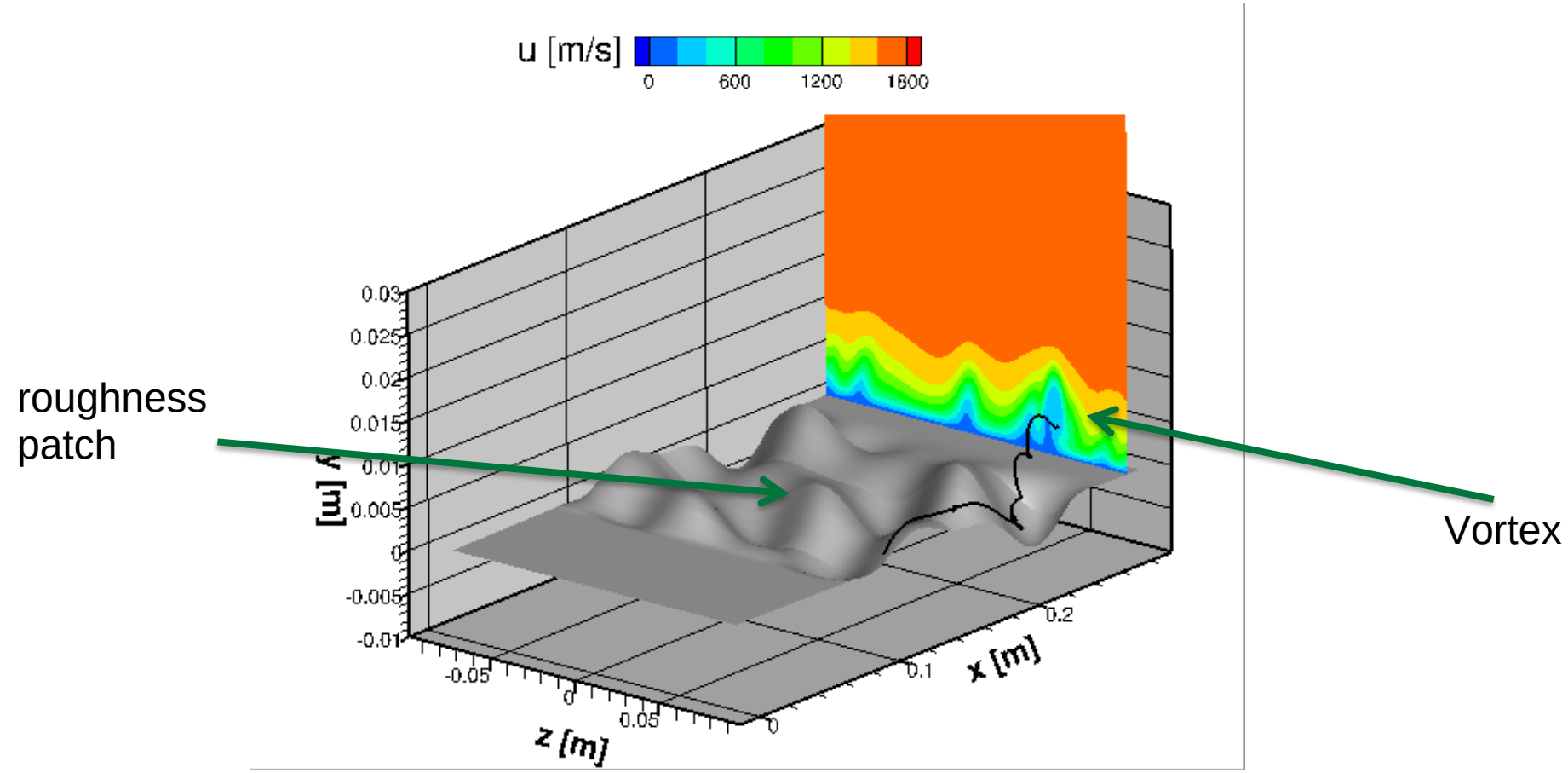
gitlab.lrz.de/nfdi4ing/crawler GitLab

* Python-based application already available via GitLab
* Reads ontologies and helps in extracting metadata from selected data files
* With limited user input, metadata files can be generated in an automated way
* The key-players of the application are Ontologies, Flat Classes, Dictionaries and Metadata. The crawler is executed in 5 steps



TUM HOMER Crawler: Example – CFD Simulation

* Hypersonic flow: Roughness patch on an Apollo-like heatshield at Mach 20



CFD Example – Steps 1 and 2

* **Step 1**: Read the ontology → Get the json file with empty flat classes
* **Step 2**: Manually fill in the flat classes AND adapt the ontology

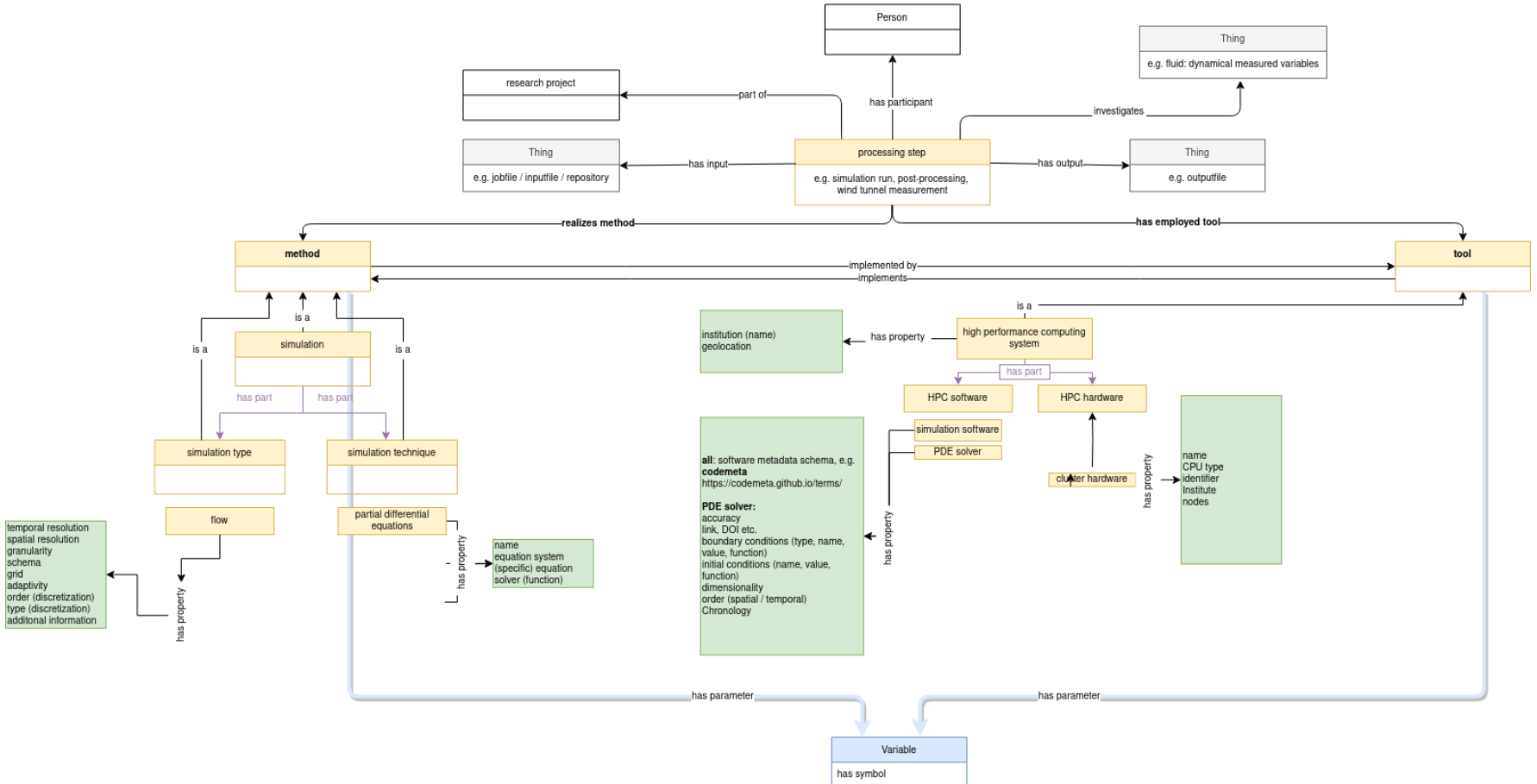
Ontology *.owl*

```
...
    "Person": {
        "__count__": 1,
        "__is_subclass_of__": [
            "Agent"
        ],
        "__restrictions__": [],
        "orcidId": [
            1
        ],
        "first_name": [
            1
        ],
        "last_name": [
            1
        ],
        "title": [
            1
```

```
...
"Person": {
        "__count__": 2,

"__is_subclass_of__": [
            "Agent"
        ],

"__restrictions__": [],
        "orcidId": [
            1
        ],
        "first_name": [
            1
        ],
        "last_name": [
            1
        ], .....
```

TUM Metadata Crawler: Adapt the NFDI4Ing Ontology

* Take some time to adapt the Ontology to your needs

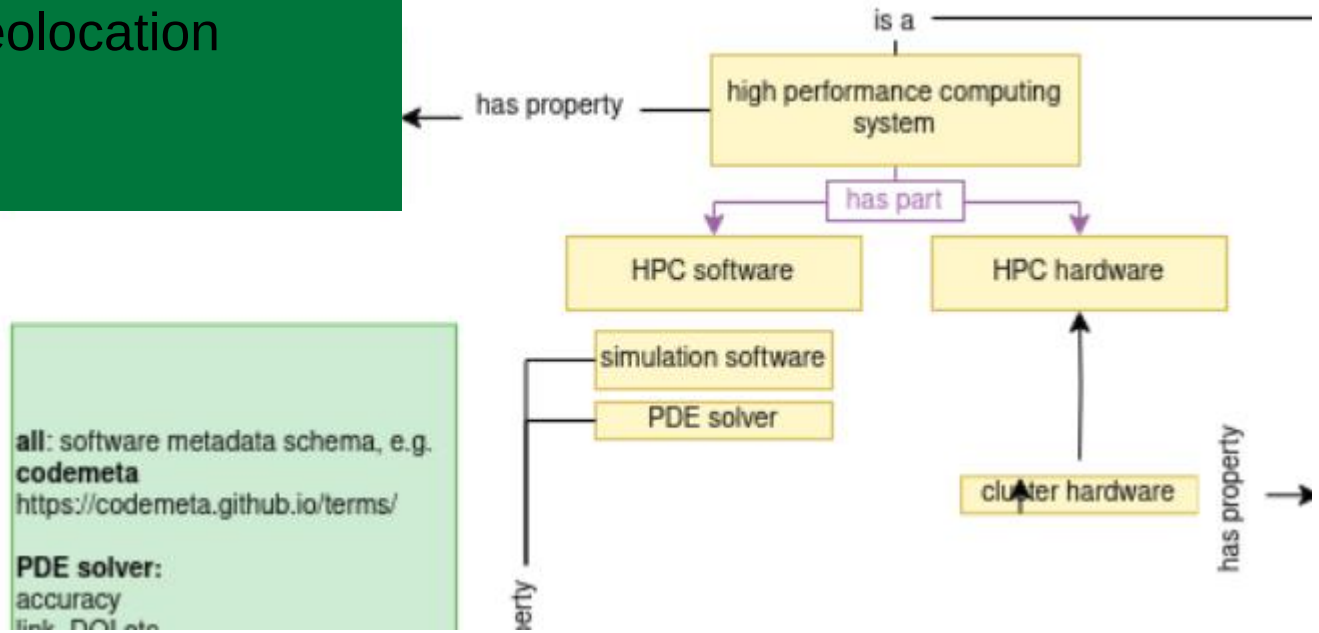


TUM Metadata Crawler: Adapt the NFDI4Ing Ontology

* Take some time to adapt the Ontology to your needs

How can the property *hpc_provider* be crawled?

hpc_provider
geolocation



CFD Example – Steps 3 and 4

* **Step 3**: Use the multiplexer to expand the classes count as needed
* **Step 4**: Use **filler.py** to fill in the extended dictionary

```
What is your main input path (file + path)? input.dat
```

relative path

```
What is your main input path (file + path)? /Documents/NFDI/Crawler/Test/input.dat
```

absolute path

CFD Example – Crawling Options

* **Step 4**: There are 4 different crawling options:

1. regex: (Python) **regular** expression inside a file
2. h5reader: get the information from an **h5 header**
3. **string**: Provide the information yourself
4. os: Use a linux **shell command** to retrieve the information (e.g.: current time, computer name ...)

CFD Example – Filler.py

```
 What is your main input path (file + path)? input.dat
Editing properties for entity: HPC_system
Do you want to use the first path value you wrote for the field: HPC_provider? (Type 'y' if yes, type any letter if not): y
Using main path value:  input.dat
Choose the type value for 'HPC_provider' from the following options:
1. regex
2. h5reader
3. string
4. os
Enter the number corresponding to your choice: 1
You selected: regex
Enter value for HPC_provider -> pattern: hpcfacility
```

```
"HPC_system": {
    "__restrictions__": [],
    "HPC_provider": {
        "path": "input.dat",
        "type": "Regex",
        "pattern": "hpcfacility",
        "postprocessor": ""
    },
```

Inputfile:
input.dat

```
15 # Name of person who makes calculation
16 my name      : Friedrich Ulrich
17 #
18 # HPC Facility
19 hpcfacility     : LRZ
20 #
21 # title of calculation
22 title        : 2D Testsimulation
```

CFD example – Step 5

* **Step 5**: Metadata extraction from target files to new .json or .yaml file

```
"HPC_system": {
    "HPC_provider": "LRZ",
    "geolocation": "Garching"
},
```

CFD example – Automate steps 4 and 5

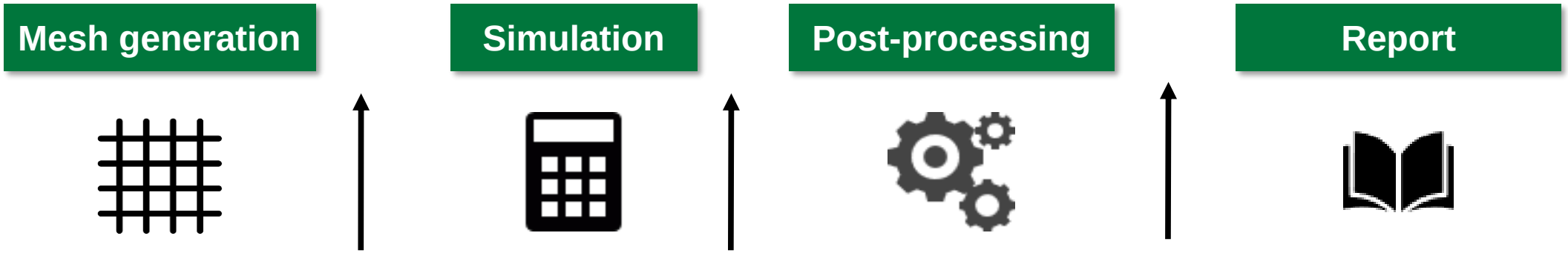
* Now, only steps 4 and 5 have to be re-run to extract similar data from different target files
* This can be easily automated (ex. via simple shell script)

```
"HPC_system": {
    "__restrictions__": [],
    "HPC_provider": {
        "path": "input.dat",
        "type": "Regex",
        "pattern": "hpcfacility",
        "postprocessor": ""
    },
```

```
"HPC_system": {
    "__restrictions__": [],
    "HPC_provider": {
        "path": "DifferentPath/input2.dat",
        "type": "Regex",
        "pattern": "hpcfacility",
        "postprocessor": ""
    },
```

TUM HOMER Crawler – Application in HPC workflow

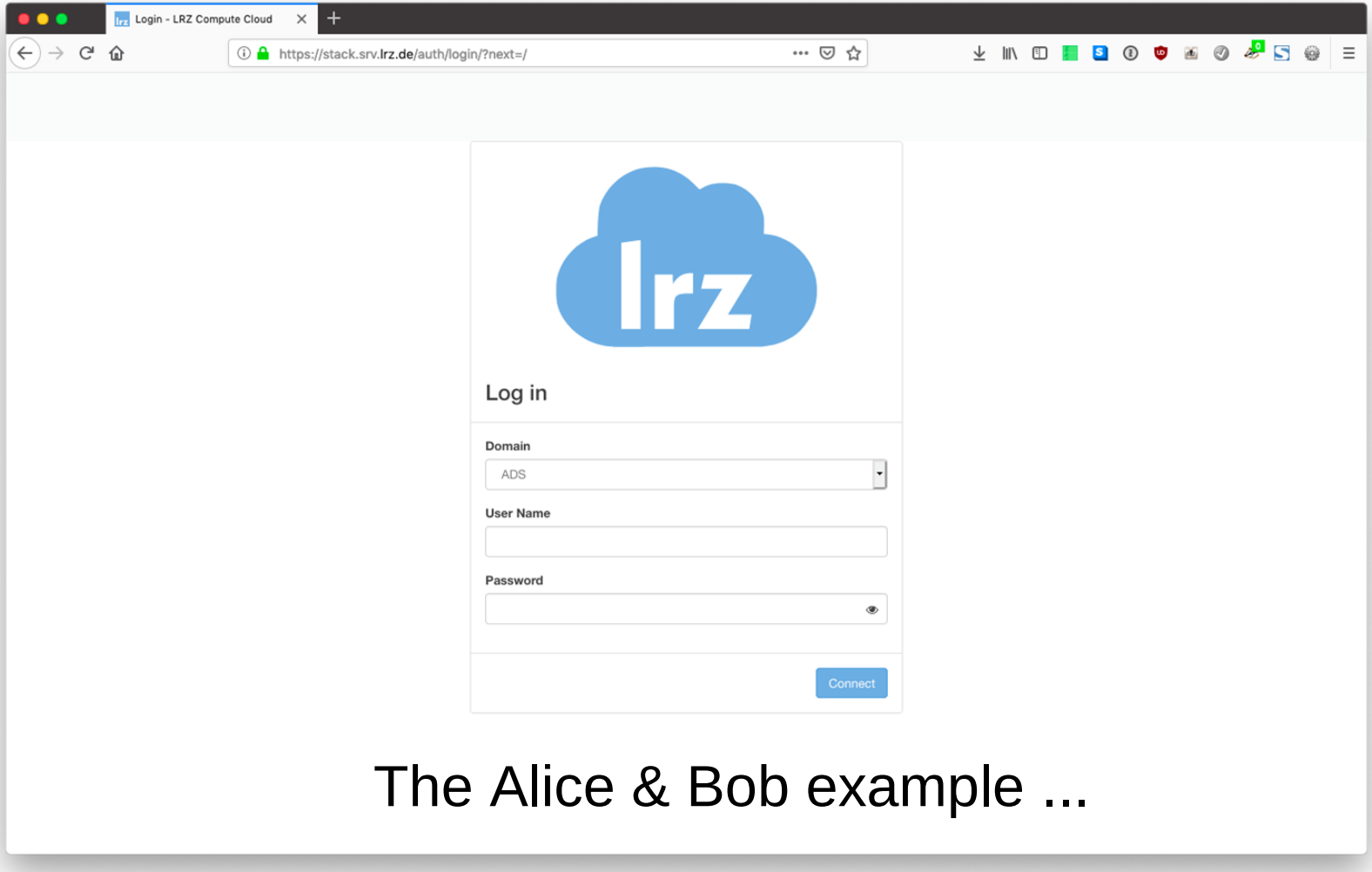
* Depending on the application, the crawler can be used at different steps within the workflow of CFD (or similar) applications
* Wherever data files are created, the crawler can be used to extract relevant metadata



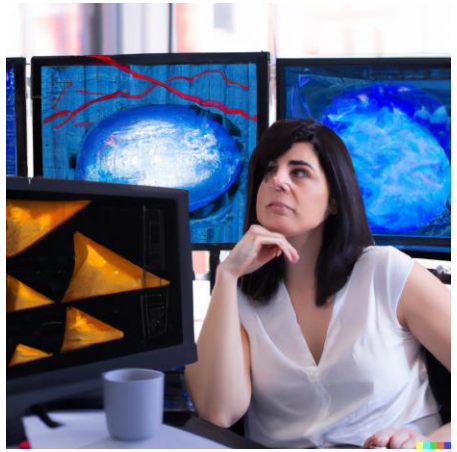
TUM HOMER Crawler – Next steps

* Increase the number of supported formats
* **Test** on a real world examples based on Metadata4Ing ontology
* recursive file crawling
* **improve user experience**
* …

Why should I use Marge?

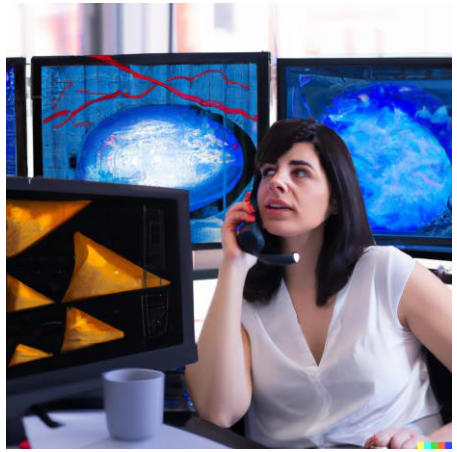


The Alice & Bob example ...



Alice works on the SuperMuc. Her data is stored on the DSS.

Bob wants to test his theory on Alice's data. He does not have LRZ access.



Alice dataset is 6.5 TB large.

Bob’s options:

Downloading several TB of data
and processing it with
his own machine...

or using the NFDI4Ing Cloud!

The NFDI4Ing Cloud:

LRZ USER

- share DSS data easily
- provide (visual) CFD results within the browser

(Non) LRZ USER

- analyze DSS data without LRZ account
- Full linux system: bring your own code
- no download of full dataset
- Look at results of colleagues within your browser

More data gets used, more colaborations, more citations ...

Sharedss: share data

```
ga25koj3@dsshare:~$ sharedss share BL
groupadd: group 'dss-access-ga25koj3' already exists
Created /sharedss/ga25koj3
Changed ownership to ga25koj3:dss-access-ga25koj3
Saved settings to /var/lib/sharedss/mounts/ga25koj3
Bound /dss/container/ga25koj3/BL subdirectory to /sharedss/ga25koj
3 with access allowed for all users in group dss-access-ga25koj3
Success
ga25koj3@dsshare:~$ sharedss unshare BL
Success
```

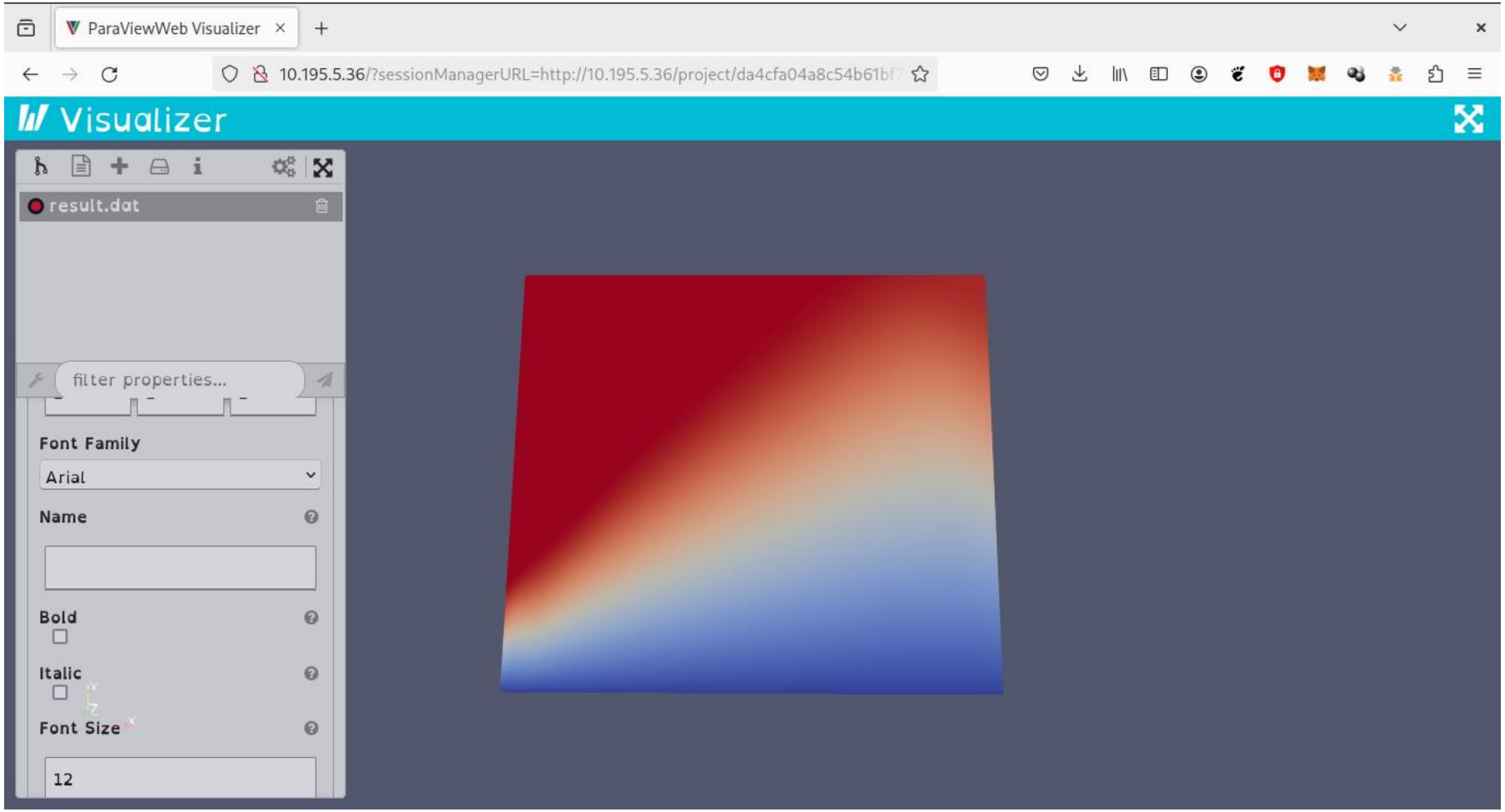
Sharedss: user mangement

```
ga25koj3@dsshare:~$ sharedss list

ga25koj3@dsshare:~$ sharedss allow fulrich
Adding user `fulrich' to group `dss-access-ga25koj3' ...
Adding user fulrich to group dss-access-ga25koj3
Done.
ga25koj3@dsshare:~$ sharedss list
fulrich
ga25koj3@dsshare:~$ sharedss revoke fulrich
Removing user `fulrich' from group `dss-access-ga25koj3' ...
Done.
ga25koj3@dsshare:~$ sharedss list

ga25koj3@dsshare:~$
```

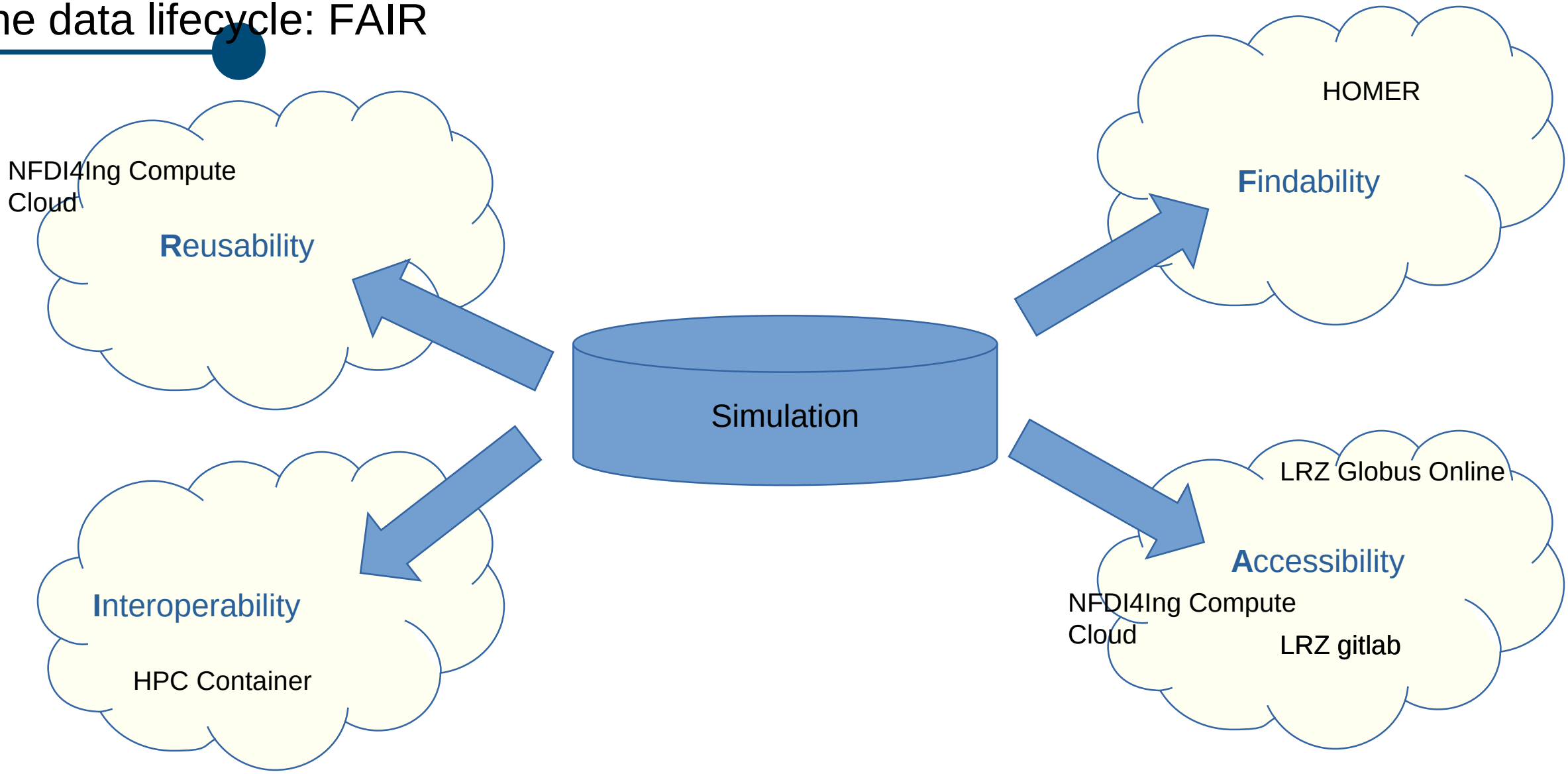
Sharedss: on the user side...

new folder

Visualizer: Demo



The data lifecycle: FAIR



What's next?

- Pilot users
- DFN web-based user mangement system
- Searchable database of shared metadata on the cloud