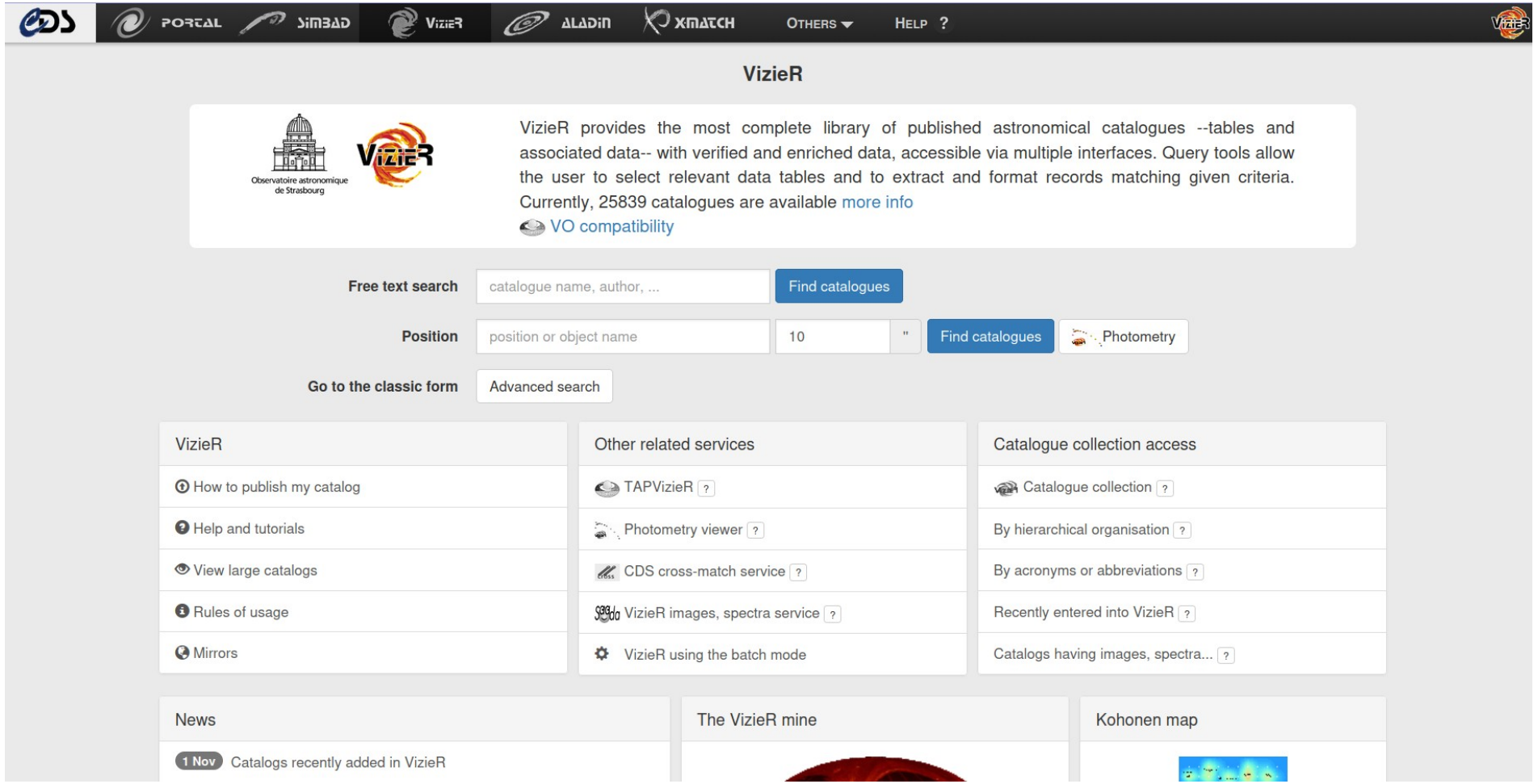
CENTRE DE DONNÉES
ASTRONOMIQUES DE STRASBOURG

ObsCore Builder : a tool to automatically generate ObsCore tables

Nathan Barloy, Gilles Landais

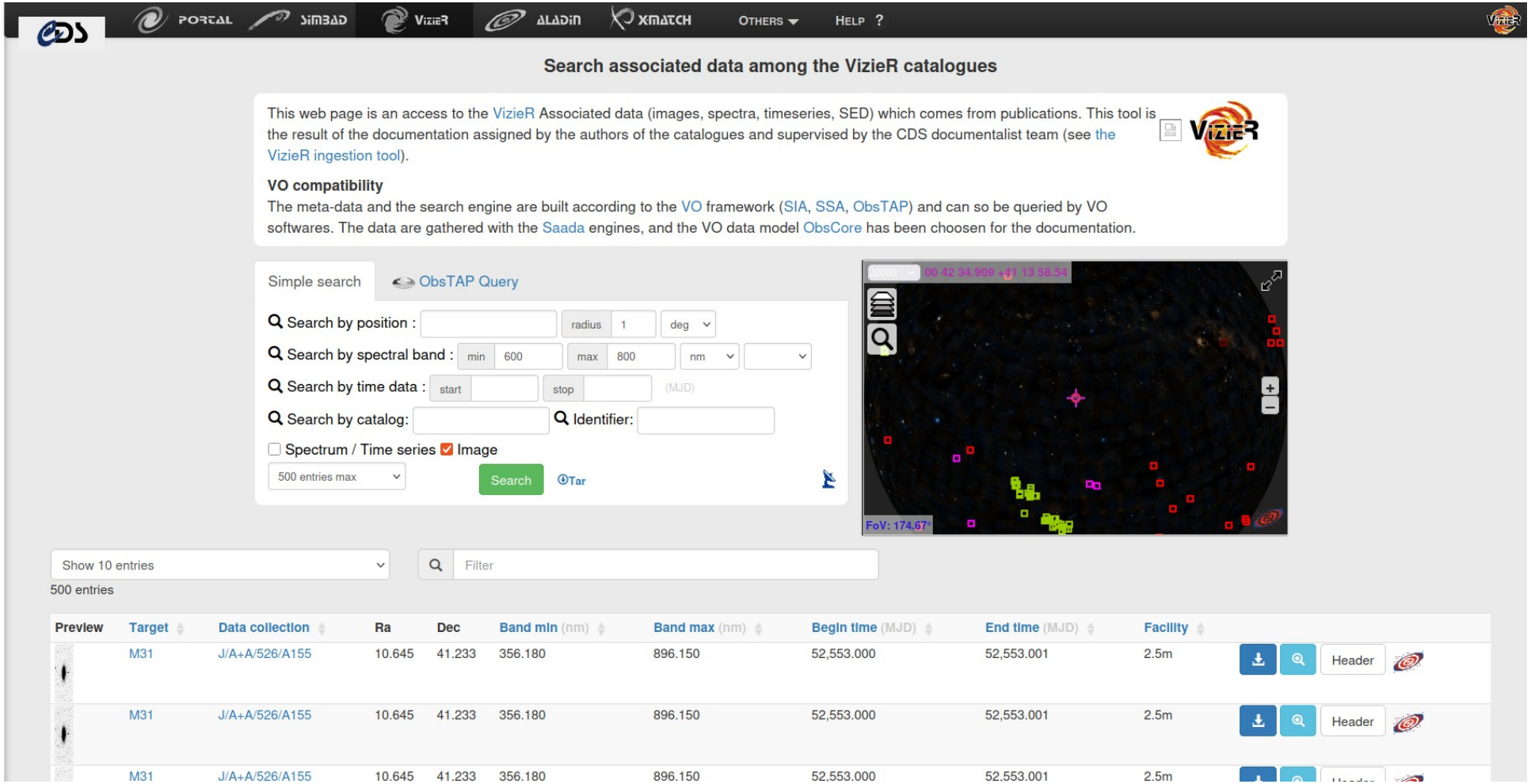
1. Context

Vizier: a catalog database



1. Context

VizieR catalogs indexing with an ObsCore database (called associated data)



1. Context

- Associated data (ObsCore) ingestion is currently done with Saada

1. Context

- Associated data (ObsCore) ingestion is currently done with Saada
- For now, only 3% of the catalogs are indexed in the assocdata database.

1. Context

- Associated data (ObsCore) ingestion is currently done with Saada
- For now, only 3% of the catalogs are indexed in the assocdata database.
- We want to add more data in those tables and add more functionnalities, and this can be difficult with Saada.

1. Context

- Associated data (ObsCore) ingestion is currently done with Saada

- For now, only 3% of the catalogs are indexed in the assocdata database.
- We want to add more data in those tables and add more functionnalities, and this can be difficult with Saada.
- This is why we want to create a new tool, in the form of a python library that can integrated in lots of forms.

1. Context

CDS – Open Data pour l'ASStronomie (CDS-ODAS)

Un renouvellement majeur du service de données associées CDS VizieR.

https://cdsarc.cds.unistra.fr/assocdata/

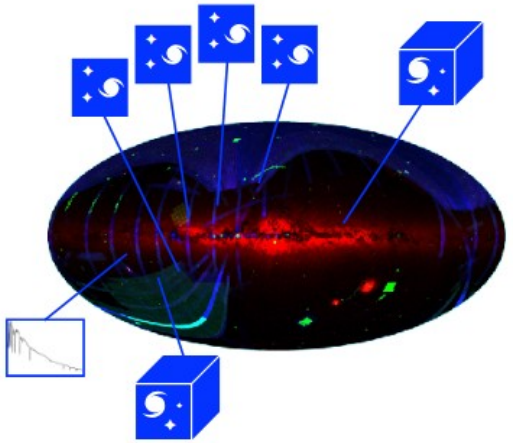
https://vizier.cds.unistra.fr/

1. Renouvellement du workflow pour le dépôt des données associées aux publications de revues à comité de lecture: images, spectres/séries temporelles, cubes de données

2. Renouvellement du mécanisme d'extraction des métadonnées des cubes / images / séries temporelles au modèle de données standard IVOA.

Études de faisabilité sur différents systèmes d'extraction de métadonnées et d'ingestion des données. (y compris Dataverse)

3. Intégration avec les interfaces de service VizieR rendant les données FAIR via des protocoles d'accès standardisés.



4. Documentation et diffusion à la communauté astronomique - mise en avant des nouvelles capacités de publication des données pour soutenir la science ouverte.

2. Objectives

- To have a simple and easy to use internal tool, that allows the ingestion of the associated data → a python library

2. Objectives

- To have a simple and easy to use internal tool, that allows the ingestion of the associated data → a python library

- To allow for an easy access to the associated data and the catalogs via standard protocols like TAP, SIA, SSA...

2. Objectives

- To have a simple and easy to use internal tool, that allows the ingestion of the associated data → a python library
- To allow for an easy access to the associated data and the catalogs via standard protocols like TAP, SIA, SSA...
- To create a web page for catalogs submission in a new workflow, allowing the user to see the ObsCore table and correct it if needed

3. How does it work ?

The core of this library is to give functions allowing to generate an ObsCore table from a FITS file collection.

The three main steps for the ObsCore generation are the following:

1) Automated extraction of data from the FITS files

2) Data curation with external services

3) Manual addition of data to the table

3. How does it work ?

1) Automated extraction of data from the FITS files

- The library is mainly based on Astropy

3. How does it work ?

1) Automated extraction of data from the FITS files

- The library is mainly based on Astropy
- Only the headers of the FITS files are used

3. How does it work ?

1) Automated extraction of data from the FITS files

- The library is mainly based on Astropy
- Only the headers of the FITS files are used
- Some data are simply retrieved through some common keywords

3. How does it work ?

1) Automated extraction of data from the FITS files

- The library is mainly based on Astropy
- Only the headers of the FITS files are used
- Some data are simply retrieved through some common keywords
- The WCS is solved with Astropy (mainly)

3. How does it work ?

1) Automated extraction of data from the FITS files

- The library is mainly based on Astropy
- Only the headers of the FITS files are used
- Some data are simply retrieved through some common keywords
- The WCS is solved with Astropy (mainly)
- What flexibility should we have with the given FITS ?

3. How does it work ?

What flexibility should we have with the given FITS ?

We would like to accept everything we can, but...

3. How does it work ?

What flexibility should we have with the given FITS ?

We would like to accept everything we can, but...

Sometimes, some non-standard things are used in multiple FITS

CTYPE1 = ‘LINEAR’

We can add a new rule for this

3. How does it work ?

What flexibility should we have with the given FITS ?

We would like to accept everything we can, but...

Sometimes, some non-standard things are used in multiple FITS

CTYPE1 = ‘LINEAR’

We can add a new rule for this

Sometimes, we could infer some information, but shouldn’t

CTYPE1 = ‘WAVE’
but no CUNIT1 given

The unit can be given in the comments, if not the default is ‘m’

3. How does it work ?

What flexibility should we have with the given FITS ?

We would like to accept everything we can, but...

Sometimes, some non-standard things are used in multiple FITS

CTYPE1 = ‘LINEAR’

We can add a new rule for this

Sometimes, we could infer some information, but shouldn’t

CTYPE1 = ‘WAVE’
but no CUNIT1 given

The unit can be given in the comments, if not the default is ‘m’

In general, we to accept non-standard rules, as long as they don’t contradict with the standards.

3. How does it work ?

2) Data curation

- We can fetch missing data in external services like Simbad, for example if an object name is given, but not the coordinates.

3. How does it work ?

2) Data curation

- We can fetch missing data in external services like Simbad, for example if an object name is given, but not the coordinates.

- If the wavebands are missing, we can try to get them with the name of the observatory, the instrument or the filter.
  From public data that we found, we tried to create a local database of filters

IRS filters:

Filter ID	λ_{ref}	λ_{mean}	λ_{eff}	λ_{min}	λ_{max}	W_{eff}	ZP_{ν}	ZP_{λ}	Obs. Facility	Instrument	Description
IRS/MOS.band408	4074.78	4075.25	4070.38	3979.53	4220.00	100.94	4010.04	7.24e-9	IRS	MOS	IRS-P3 MOS band408 RSR
IRS/MOS.band443	4429.16	4429.55	4429.38	4333.52	4509.48	108.11	4369.58	6.68e-9	IRS	MOS	IRS-P3 MOS band443 RSR
IRS/MOS.band485	4837.20	4837.55	4833.11	4745.51	4909.89	105.65	3470.45	4.45e-9	IRS	MOS	IRS-P3 MOS band485 RSR
IRS/MOS.band520	5197.30	5197.61	5197.04	5110.21	5271.82	103.57	3793.18	4.21e-9	IRS	MOS	IRS-P3 MOS band520 RSR
IRS/MOS.band570	5694.41	5694.68	5694.17	5613.92	5768.27	103.59	3484.89	3.22e-9	IRS	MOS	IRS-P3 MOS band570 RSR
IRS/MOS.band615	6148.07	6148.33	6147.82	6072.10	6219.89	107.31	3210.30	2.55e-9	IRS	MOS	IRS-P3 MOS band615 RSR
IRS/MOS.band650	6496.60	6496.85	6495.15	6422.05	6577.34	108.54	2929.60	2.08e-9	IRS	MOS	IRS-P3 MOS band650 RSR

3. How does it work ?

3) Manual addition of data to the table

- Motivated by the fact that we can’t directly modify the FITS we receive. Furthermore, some catalogs sometimes share a format.

3. How does it work ?

3) Manual addition of data to the table

- Motivated by the fact that we can’t directly modify the FITS we receive. Furthermore, some catalogs sometimes share a format.
- This can be done with a Mapping file, that allows the user to give some additional information.

3. How does it work ?

3) Manual addition of data to the table

- Motivated by the fact that we can’t directly modify the FITS we receive. Furthermore, some catalogs sometimes share a format.
- This can be done with a Mapping file, that allows the user to give some additional information.
- This file can be given when extracting an ObsCore table from FITS file, or directly applied on an existing table.

3. How does it work ?

3) Manual addition of data to the table

- Motivated by the fact that we can’t directly modify the FITS we receive. Furthermore, some catalogs sometimes share a format.
- This can be done with a Mapping file, that allows the user to give some additional information.
- This file can be given when extracting an ObsCore table from FITS file, or directly applied on an existing table.
- Different mapping can be used for different files.

3. How does it work ?

3) Manual addition of data to the table

```
CUNIT1: 'Angstrom'

facility_name:= 'LAMOST'
instrument_name: TELESCOP
target_name: split($FILENAME, '_')[2]
```

3. How does it work ?

3) Manual addition of data to the table

Add a default value
to a FITS keyword

```
CUNIT1: 'Angstrom'

facility_name:= 'LAMOST'
instrument_name: TELESCOP
target_name: split($FILENAME, '_')[2]
```

3. How does it work ?

3) Manual addition of data to the table

Add a default value to a FITS keyword

Force a value in an ObsCore field

```
CUNIT1: 'Angstrom'

facility_name:= 'LAMOST'
instrument_name: TELESCOP
target_name: split($FILENAME, '_')[2]
```

3. How does it work ?

3) Manual addition of data to the table

Add a default value to a FITS keyword

Force a value in an ObsCore field

Fill an ObsCore field with a FITS keyword

```
CUNIT1: 'Angstrom'

facility_name:= 'LAMOST'
instrument_name: TELESCOP
target_name: split($FILENAME, '_')[2]
```

3. How does it work ?

3) Manual addition of data to the table

Add a default value to a FITS keyword

Force a value in an ObsCore field

Fill an ObsCore field with a FITS keyword

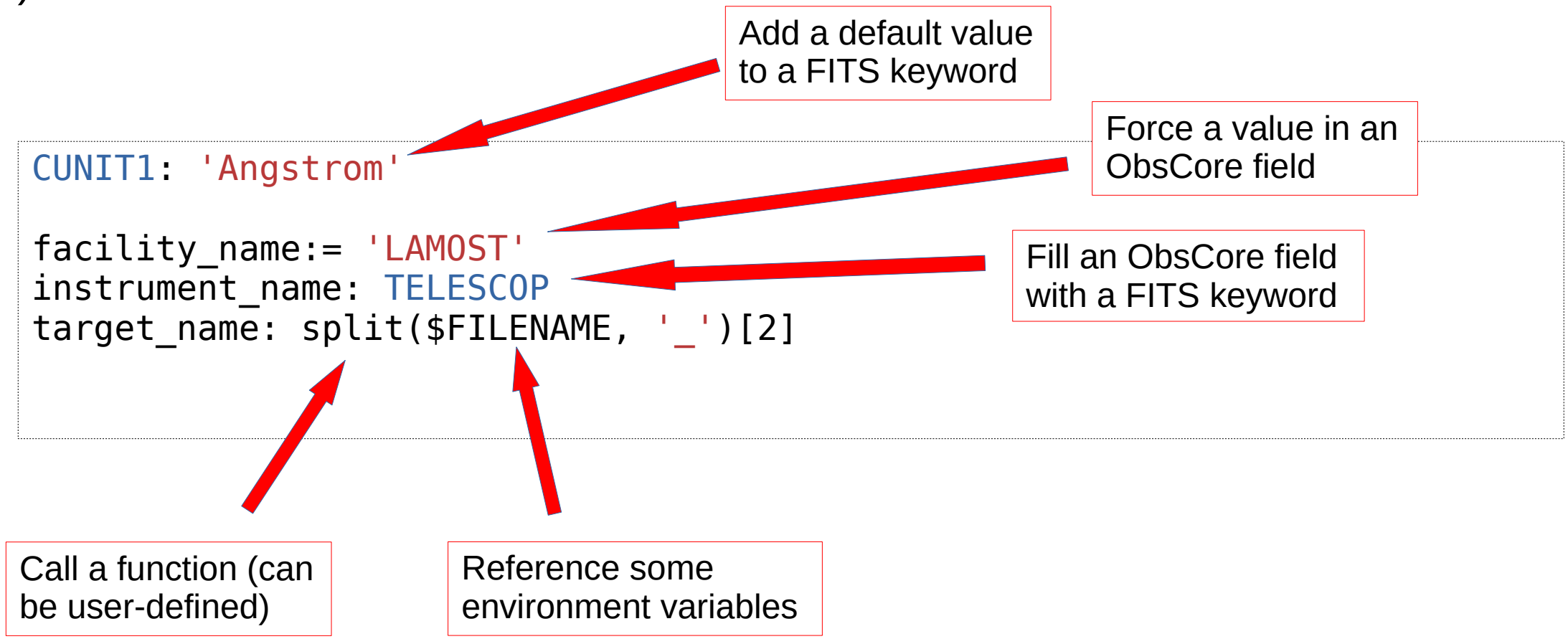
```
CUNIT1: 'Angstrom'

facility_name:= 'LAMOST'
instrument_name: TELESCOP
target_name: split($FILENAME, '_')[2]
```

Call a function (can be user-defined)

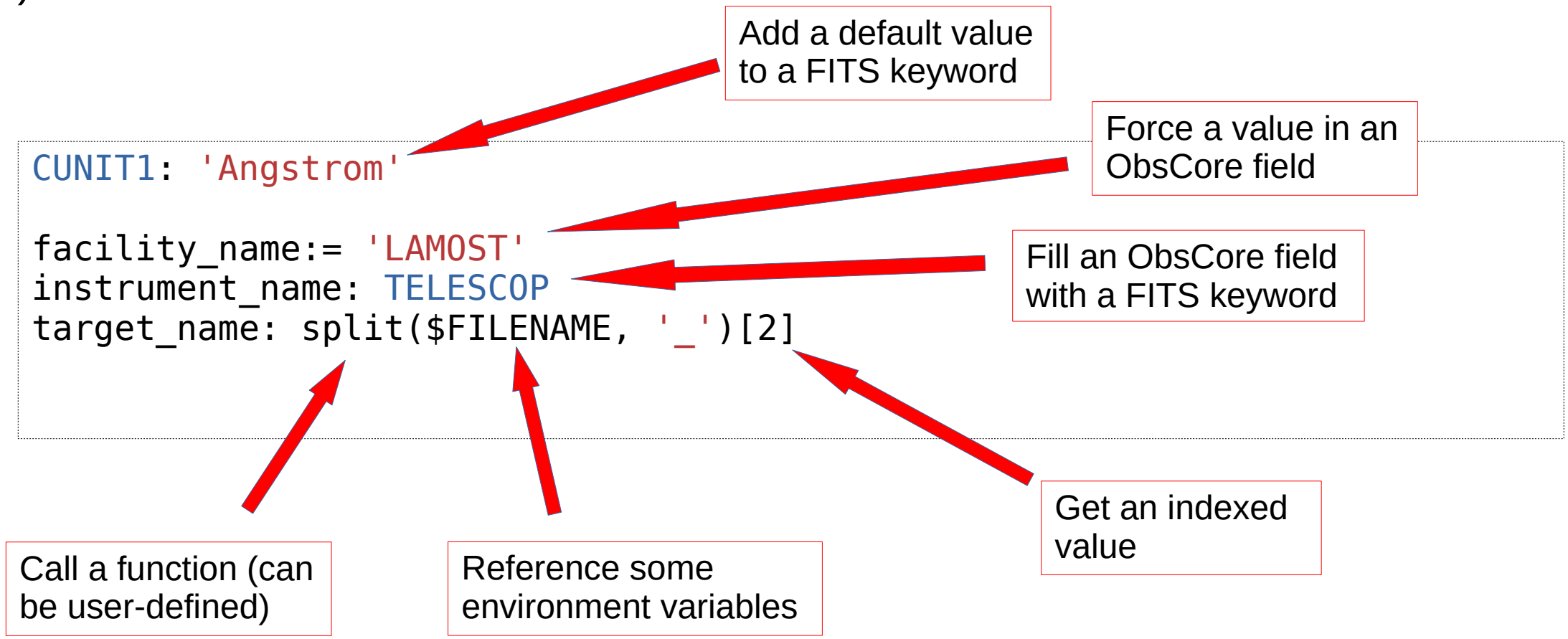
3. How does it work ?

3) Manual addition of data to the table



3. How does it work ?

3) Manual addition of data to the table




```
CUNIT1: 'Angstrom'

facility_name:= 'LAMOST'
instrument_name: TELESCOP
target_name: split($FILENAME, '_')[2]
```

4. Usage

The python usage is really simple

```
from assocdata import extract, apply_mapping

catalog_directory = '~/Documents/catalog1.fits'
response = extract(catalog_directory)
response.curate()
obscore_table = response.to_table()

# With a mapping
response = extract(catalog_directory, './path/to/mapping_file.map')

# We can also apply a mapping afterwards
apply_mapping(obscore_table, './path/to/mapping_file.map')
```

What’s next ?

- Support more non-standard FITS format

What’s next ?

- Support more non-standard FITS format
- Make a CLI tool that can be easily installed by our staff to test this library

What’s next ?

- Support more non-standard FITS format
- Make a CLI tool that can be easily installed by our staff to test this library
- Create of new webpage for data submission, to have a workflow that will use this tool

What’s next ?

- Support more non-standard FITS format
- Make a CLI tool that can be easily installed by our staff to test this library
- Create of new webpage for data submission, to have a workflow that will use this tool
- Make part of the library open and available on PyPi

Conclusion

This tool should allow for an easier data ingestion process for everyone: the CDS staff and the users.
As a consequence, the number of catalogs indexed in our ObsCore database should increase.

Our data will be more easily findable using a VO standard, so we are more FAIR compliant.

This tool is still in development, so a lot of things can still change.