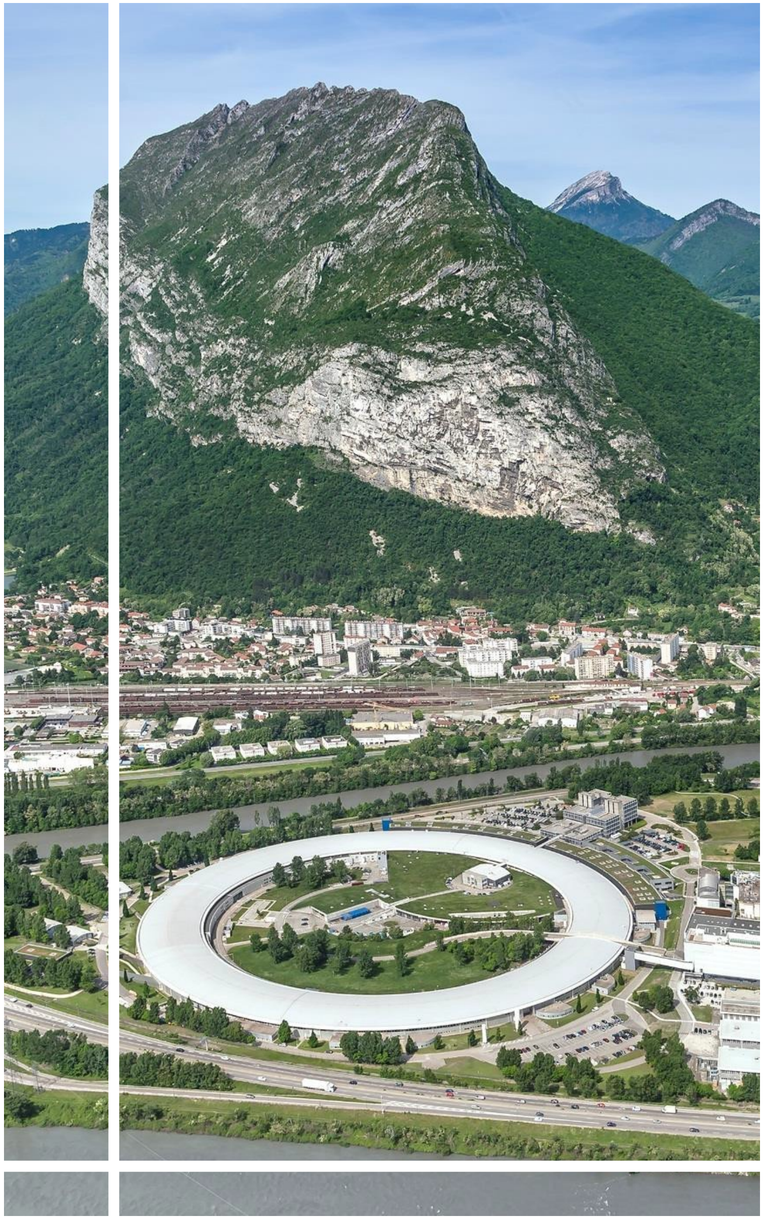
ESRF | The European Synchrotron



Ewoks

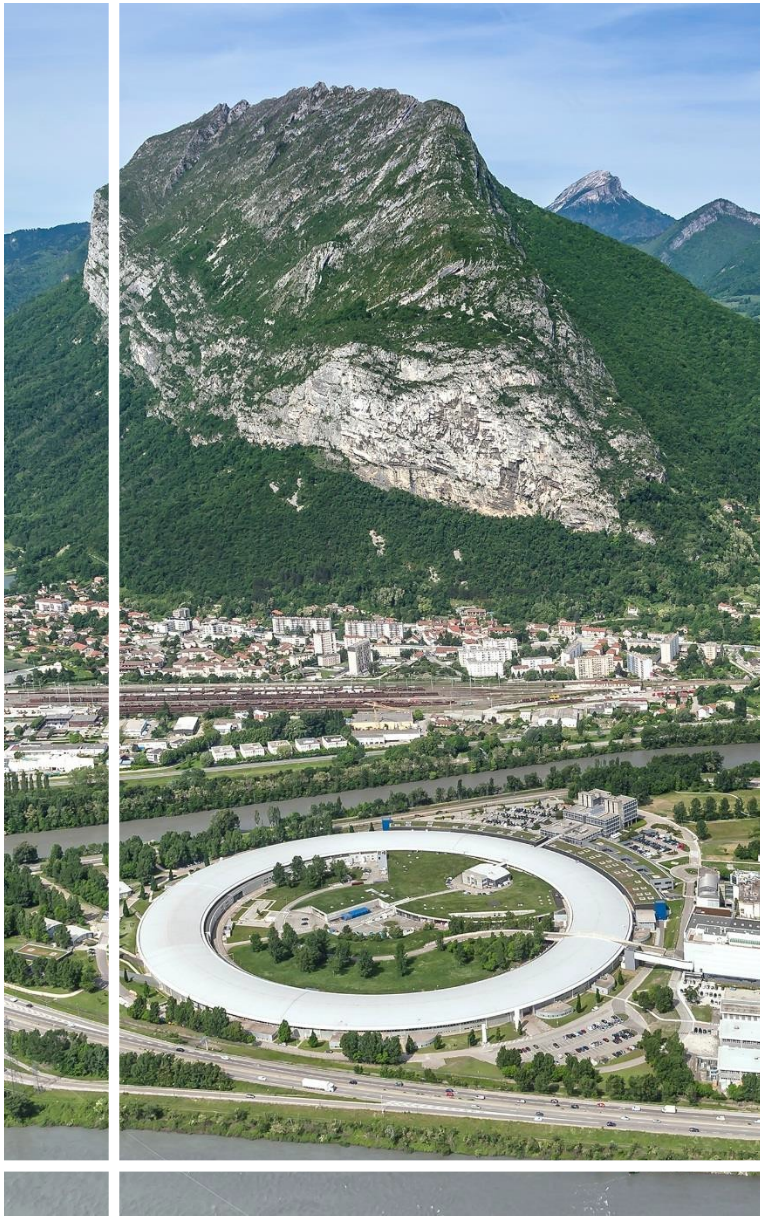
ESRF Workflow System

A meta-workflow system

W.De Nolf, I.Koumoutsos, O.Svensson, H.Payno,

J.Garriga Ferrer and A.Götz

STREAMLINE has received funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement No. 870313



Outline

1. What problem does EWOKS solve?
2. What does EWOKS provides?
   (how is it “meta”?)
3. Getting started with EWOKS
4. Examples

1. What problem does EWOKS solve?

What problem did we solve by developing EWOKS?

With the ESRF upgrade (**EBS**) and the introduction of the BeamLine Instrumentation Support Software (**BLISS**) more and more beamlines requested support for online data processing.

Solutions were developed independently in the past using various **workflow management systems** (WMS) like orange, passarelle, scipion, ... or **custom tooling**.

1. What problem does EWOKS solve?

Why workflows?

A **workflow** is a **recipe** that fully describes the steps to be taken in a **data processing** pipeline or the decisions to be made by an **automated data acquisition** system.

- **Reproducibility**: results can be regenerated from the recipe
- **Reuseability**: process many datasets with the same recipe
- **Traceability**: workflows describe the history of a piece of data
- **Human and machine**: workflows can be create and executed by both
- **Collaborative data processing**: experts from different fields can create and execute the same workflow

1. What problem does EWOKS solve?

Design requirements

We needed a way to share developer efforts across beamlines and techniques while still using the different features provided by existing workflow management systems.

- A **workflow** created in one system can also be executed in another
- A workflow **task** created in one system can also be executed in another
- **Changes** in technology should not affect users (creating workflows) and developers (implementing workflow tasks).
- Works **light-weight** (PC or laptop) and at **scale** (Beamline or Synchrotron).

1. What problem does EWOKS solve?

EWOKS is **NOT** a workflow management system.

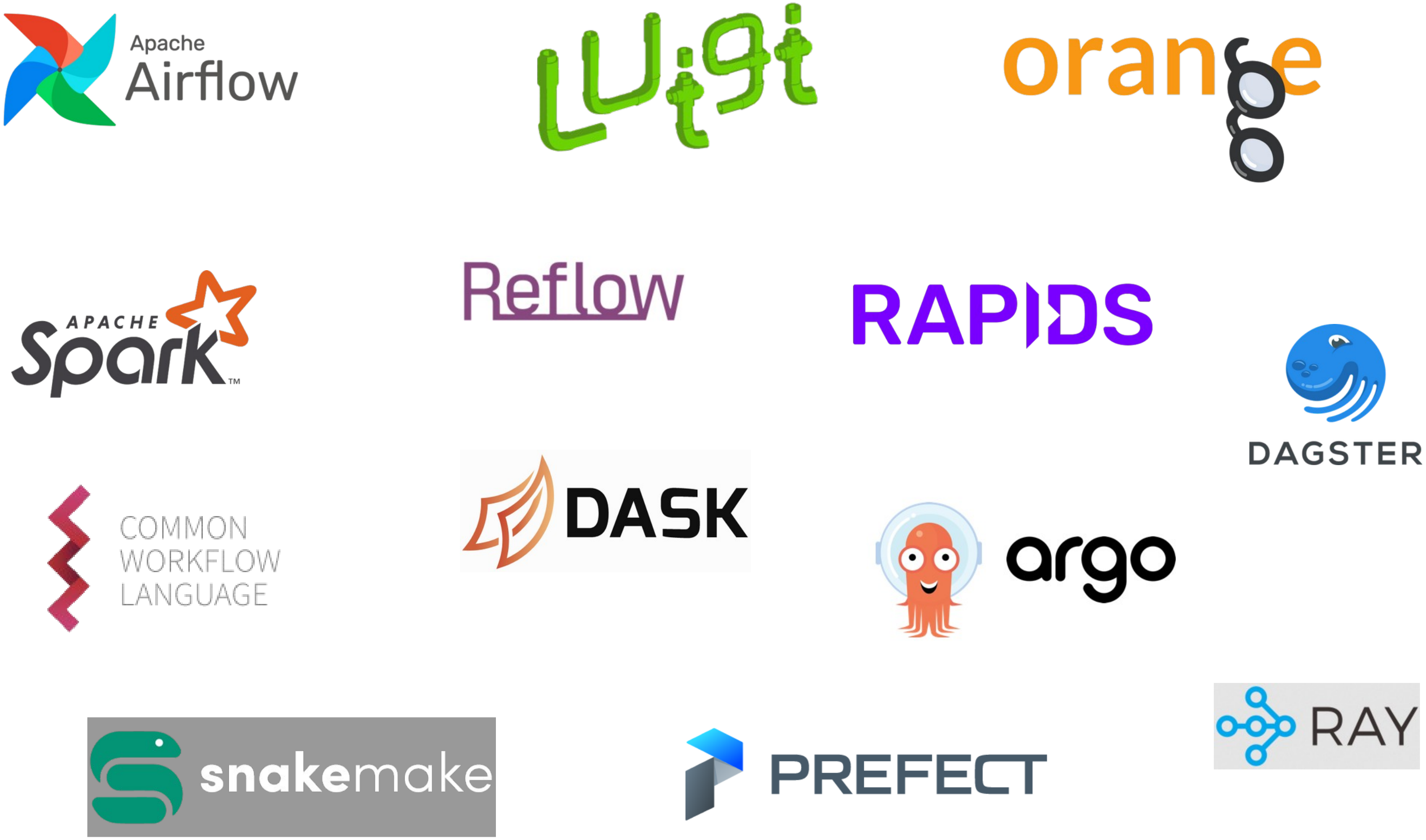
EWOKS is a **meta**-workflow system implemented in python.

It is “meta” in the sense that it **decouples** the workflow definition and task implementation from the WMS.

- EWOKS convention for workflow definition and task implementation
- Bindings for existing workflow management systems

2. What does EWOKS provide?

Many workflow management systems exist



2. What does EWOKS provide?

Current EWOKS bindings for existing workflow management systems

	WMS	Features
orange	Orange	Desktop GUI to create and execute workflows
DASK	Dask	Parallelize task execution (locally or on a cluster)
	Pypushflow	Cyclic workflows with conditional links
Celery	Celery	Schedule workflows (locally or on a cluster)

2. What does EWOKS provide?

What ewoks provides

- Common workflow definition (NetworkX + mapping for each WMS)
- Common task implementation (+ mapping for each WMS)
- Serialize workflows (NetworkX): json, yaml, …
- Execution events for monitoring: python logging facility
- Persist/cache task results: parts of a workflow need to execute only once
- Web support: frontend (React) and backend (REST) to create, execute and monitor workflows

2. What does EWOKS provide?

What ewoks does NOT provide

- Execution engine (except for a basic serial executor for DAGs)
- Data transfer between tasks (although the binding could use the persistence mechanism mentioned above)
- Worker caching
- Desktop GUI

4. Getting started with EWOKS

Install EWOKS

```
pip install ewoks
```

Create a test workflow (“acyclic1” is an test workflow that ships with ewoks)

```
ewoks convert acyclic1 --test test.json -s indent=2
```

Execute the test workflow and print output

```
ewoks execute test.json --output all
```

For more information

https://ewoks.readthedocs.io

4. Getting started with EWOKS

Install EWOKS with the orange binding (desktop GUI)

```
pip install ewoks[orange]
```

Open workflow in the orange GUI

```
ewoks execute test.json --binding orange
```

Install EWOKS for web

```
pip install ewoksserver[frontend]
```

Start the EWOKS web server

```
ewoks-server
```

For more information

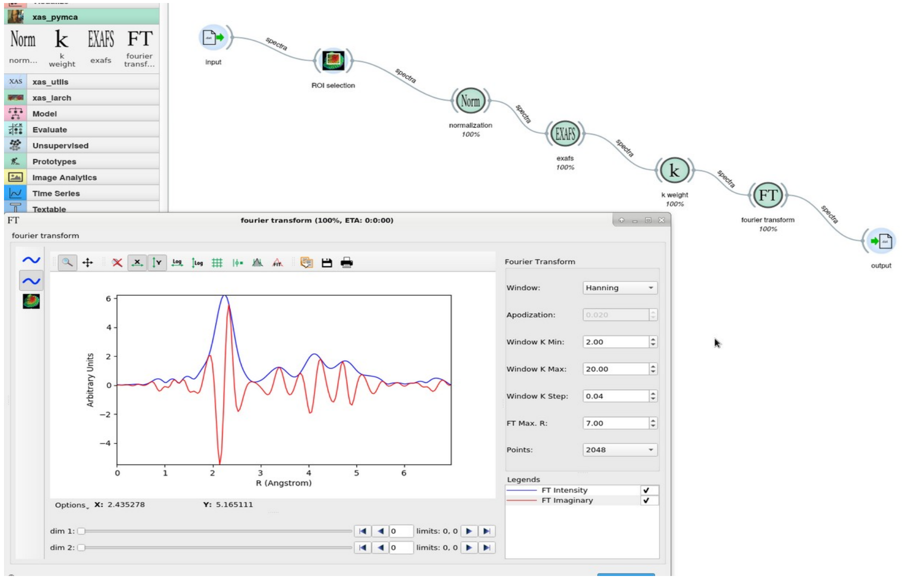
https://ewoks.readthedocs.io

4. Getting started with EWOKS

GUI to create, execute and monitor workflows

Desktop orange

Web React { REST:API }



Options X: 2.435278 Y: 5.165111

dim 1: 0 limits: 0, 0

dim 2: 0 limits: 0, 0

Fourier Transform

Window: Hanning

Apodization: 0.020

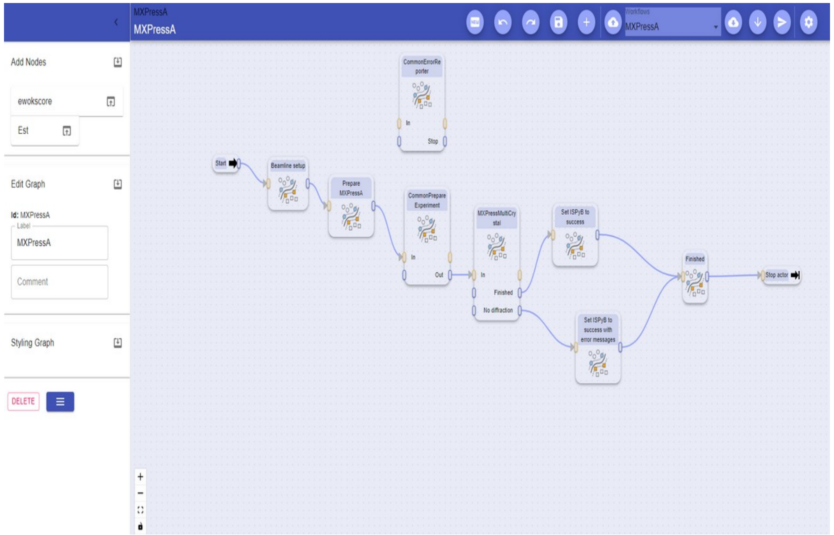
Window K Min: 2.00

Window K Max: 20.00

Window K Step: 0.04

FT Max. R: 7.00

Points: 2048



Id: MXPressA

Label: MXPressA

Comment

4. Getting started with EWOKS

Implement a workflow task

```
from ewokscore import Task

class SumTask(
    Task,
    input_names=["a"],
    optional_input_names=["b"],
    output_names=["result"]
):
    def run(self):
        result = self.inputs.a
        if self.inputs.b:
            result += self.inputs.b
        self.outputs.result = result
```

Execute a workflow

```
from ewoks import execute_graph

inputs = [{"id": "task1",
           "name": "a",
           "value": 10}]

# Optionally persist all task outputs
varinfo = {"root_uri": "/tmp/myresults"}

result = execute_graph(workflow,
                       varinfo=varinfo,
                       inputs=inputs)
print(result)
```

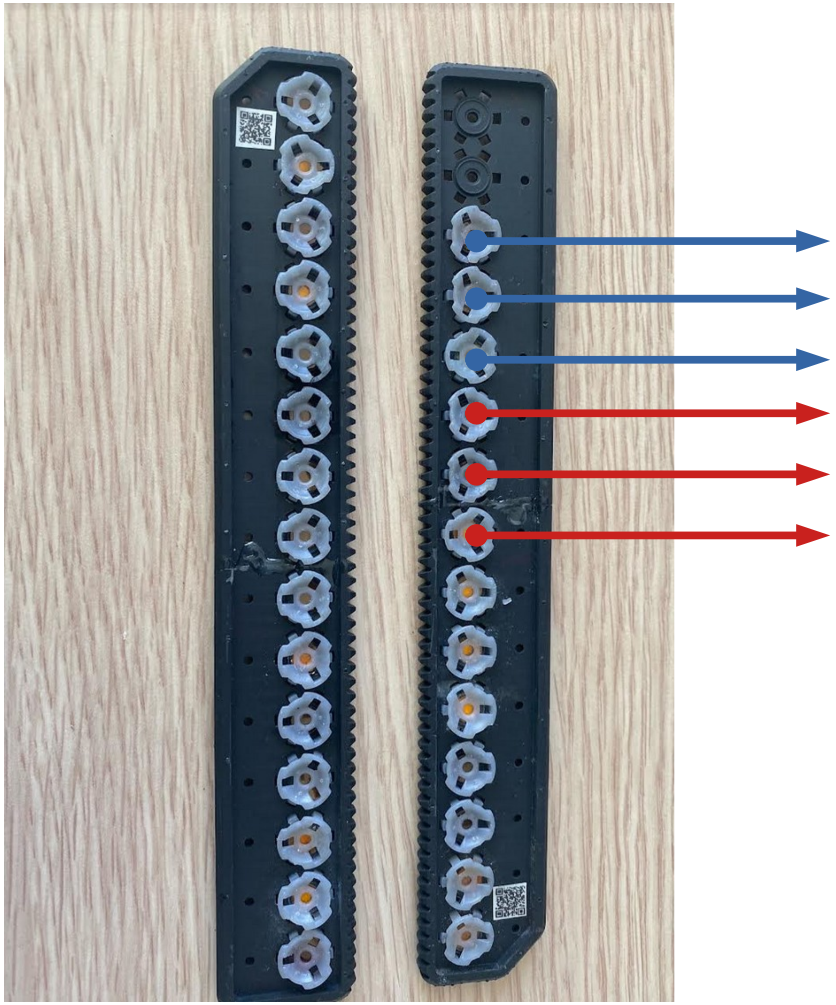
Define a workflow

```
nodes = [
    {
        "id": "task1",
        "task_type": "class",
        "task_identifier": "mypackage.SumTask",
        "default_inputs": [{"name": "a",
                            "value": 1}],
    },
    {
        "id": "task2",
        "task_type": "class",
        "task_identifier": "mypackage.SumTask",
        "default_inputs": [{"name": "b",
                            "value": 1}],
    },
]
links = [
    {
        "source": "task1",
        "target": "task2",
        "data_mapping": [{"source_output": "result",
                          "target_input": "a"}],
    },
]
workflow = {"graph": {"id": "testworkflow"},
            "nodes": nodes,
            "links": links}
```

For more information
https://ewoks.readthedocs.io

5. Examples: data reduction for X-ray powder diffraction

https://twitter.com/ESRF_Streamline



CeO_2

hole

capsule

sample_1

sample_2

...

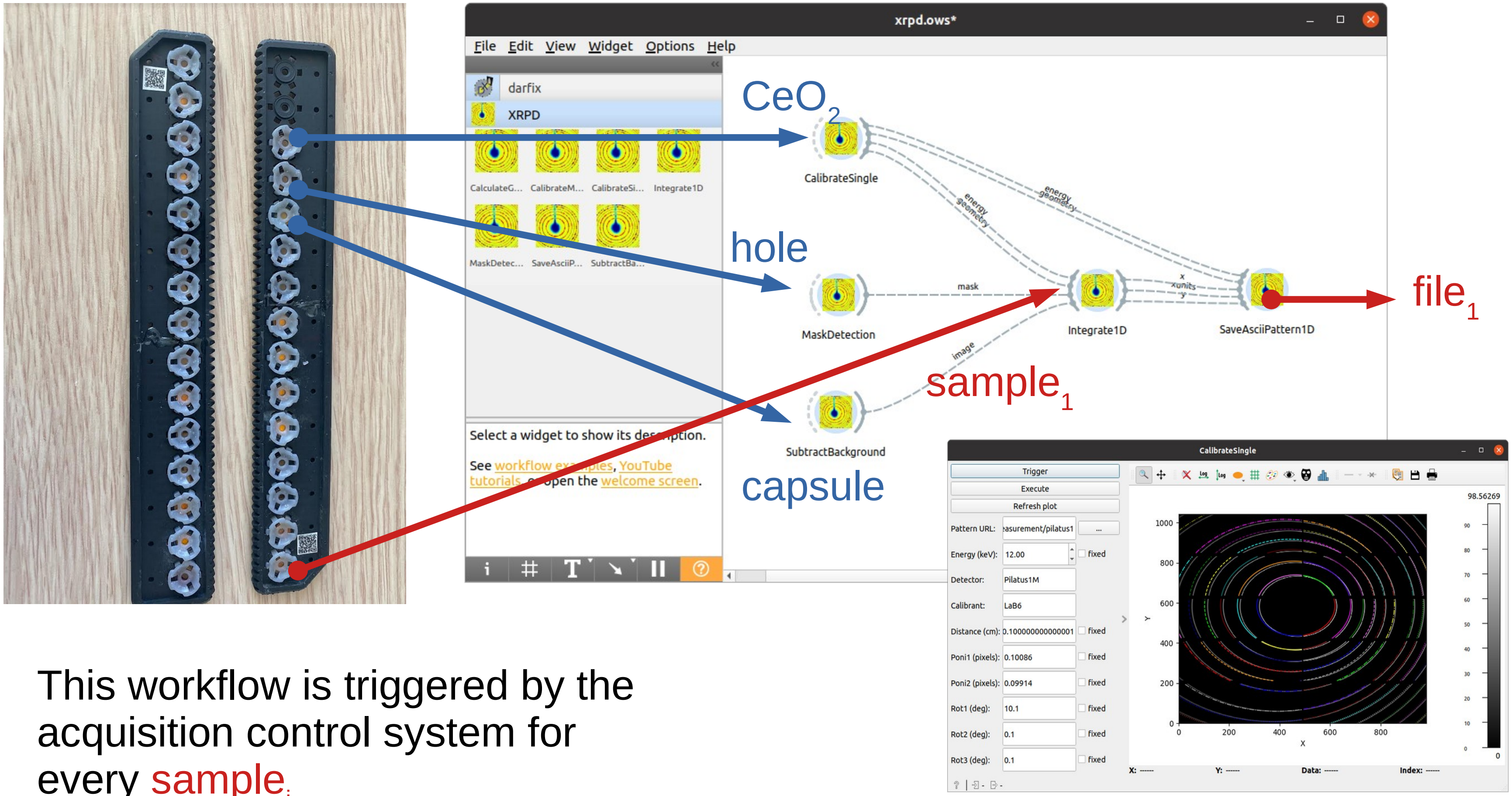
EWOKS workflow:

For calibration

For XRPD Data reduction for each sample

5. Examples: data reduction for X-ray powder diffraction

https://twitter.com/ESRF_Streamline



Select a widget to show its description.

See workflow examples, YouTube tutorials, or open the welcome screen.

CalibrateSingle

Pattern URL: :asurement/pilatus1

Energy (keV): 12.00 □ fixed

Detector: Pilatus1M

Calibrant: LaB6

Distance (cm): 0.100000000000001 □ fixed

Poni1 (pixels): 0.10086 □ fixed

Poni2 (pixels): 0.09914 □ fixed

Rot1 (deg): 10.1 □ fixed

Rot2 (deg): 0.1 □ fixed

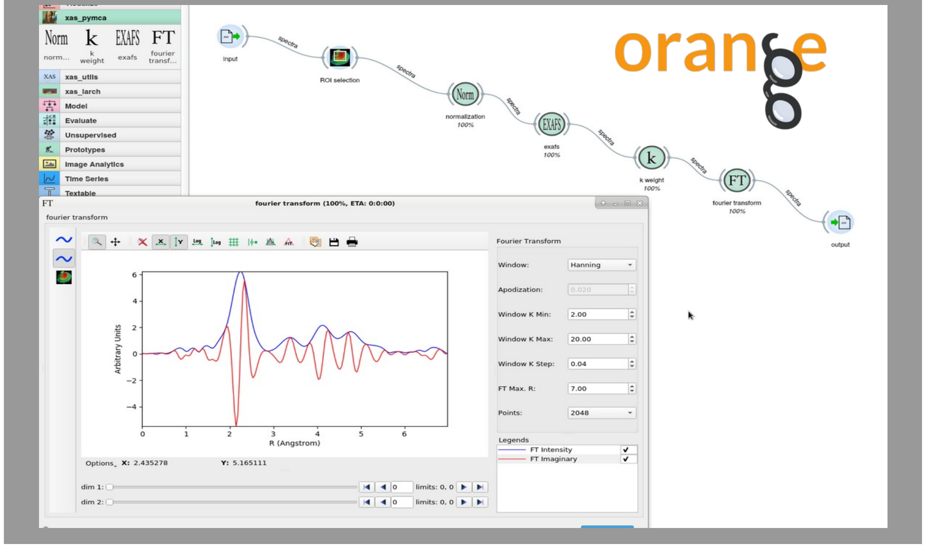
Rot3 (deg): 0.1 □ fixed

This workflow is triggered by the acquisition control system for every sample$_i$

5. Examples: online EXAFS plotting

EWOKS (GUI)

Create workflow + choose parameters



Fourier Transform

Window: Hanning

Apodization: 0.020

Window K Min: 2.00

Window K Max: 20.00

Window K Step: 0.04

FT Max. R: 7.00

Points: 2048

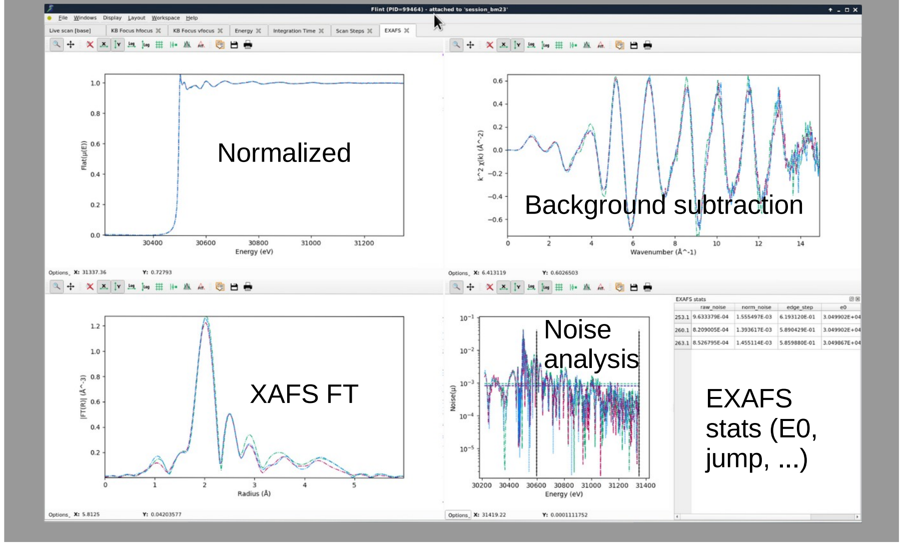
Legends

- FT Intensity ✓
- FT Imaginary ✓

X: 2.435278 Y: 5.165111

FLINT

Plot results



Normalized

Background subtraction

XAFS FT

Noise analysis

EXAFS stats (E0, jump, ...)

EXAFS stats

	raw_noise	norm_noise	edge_step	e0
253.1	9.633379E-04	1.555497E-03	6.193120E-01	3.049902E+04
260.1	8.209005E-04	1.393617E-03	5.890429E-01	3.049902E+04
263.1	8.526795E-04	1.455114E-03	5.859880E-01	3.049867E+04

Save workflow in a file

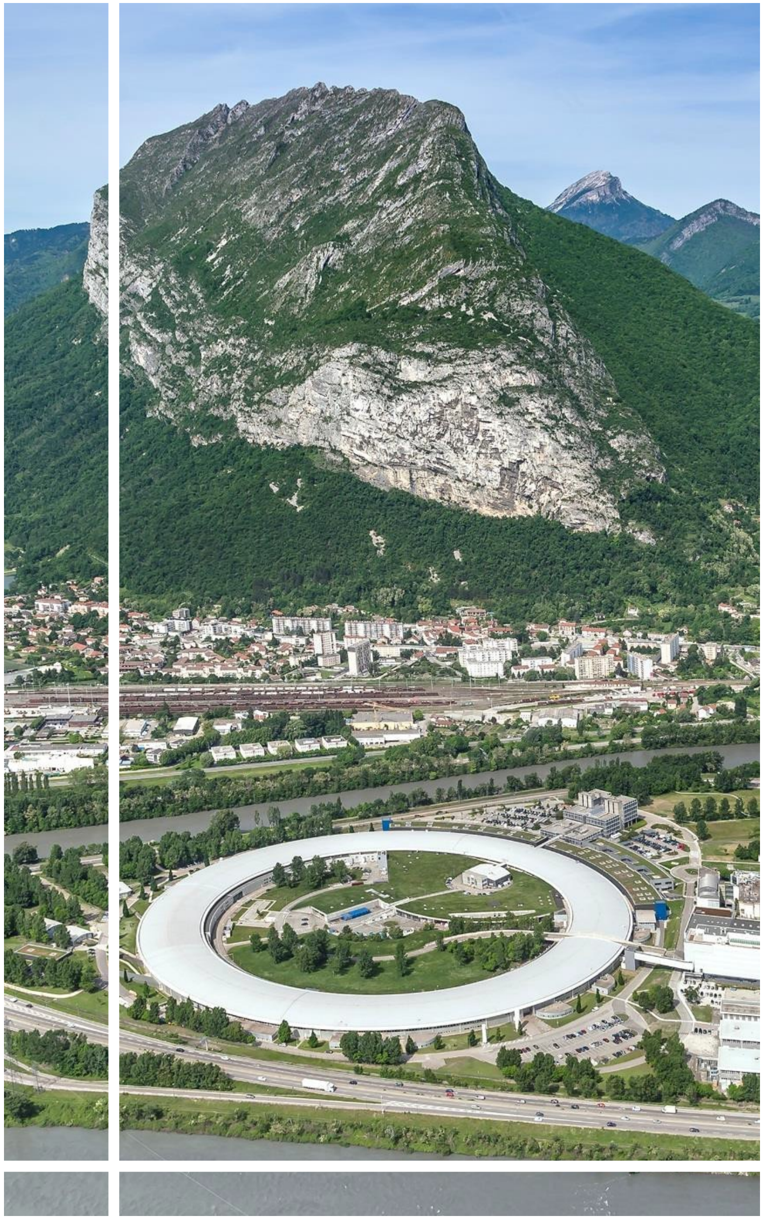
Every x seconds during scan: send workflow to worker and send result to Flint

EWOKS (worker)

Execute workflow

redis

Celery



Ewoks

May the force be with you

Documentation
https://ewoks.readthedocs.io

DOI
10.5281/zenodo.6075054

STREAMLINE has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No. 870313