



# PaNOSC Closing Event

Paving the way towards the PaN FAIR Data Commons

29-30 November 2022

Grenoble - France

## VINYL Simulations – post-project plans

**Authors:** Carsten Fortmann-Grote, Juncheng E, Mads Bertelsen,  
Shervin Nourbakhsh, Aljosa Hafner, Manuel Sanchez del Rio,  
Mousumi Upadhyay Kahaly, Nagy Gregely, Nobert

**Affiliations:** European Spallation Source, European XFEL, Institute Laue-Langevin  
CERIC-ERIC, European Synchrotron Radiation Facility, ELI-ALPS

**29/11/2022**



PaNOSC has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement no. 823852

# Introduction

## Aim of WP5 - ViNYL

- Virtual Neutron and X-ray Laboratory
- Harmonize simulation codes for neutron and x-ray experiments
- Agree on dataformats
  - For transfer of signal between codes
  - For detector results
- Led by Carsten Fortmann-Grote
- Chose Python packages with OpenPMD and Nexus data

# Introduction

## Aim of WP5 - ViNYL

Solutions at the end of PaNOSC

Neutrons

Simulation software  
at start of PaNOSC

X-ray

McStas  
(C meta language)

SimEx  
(Python Framework)

OASYS  
(Python GUI application)  
(Framework)

McStasScript  
(Python API)

SimEx-lite  
(Python Framework)

Sim engines

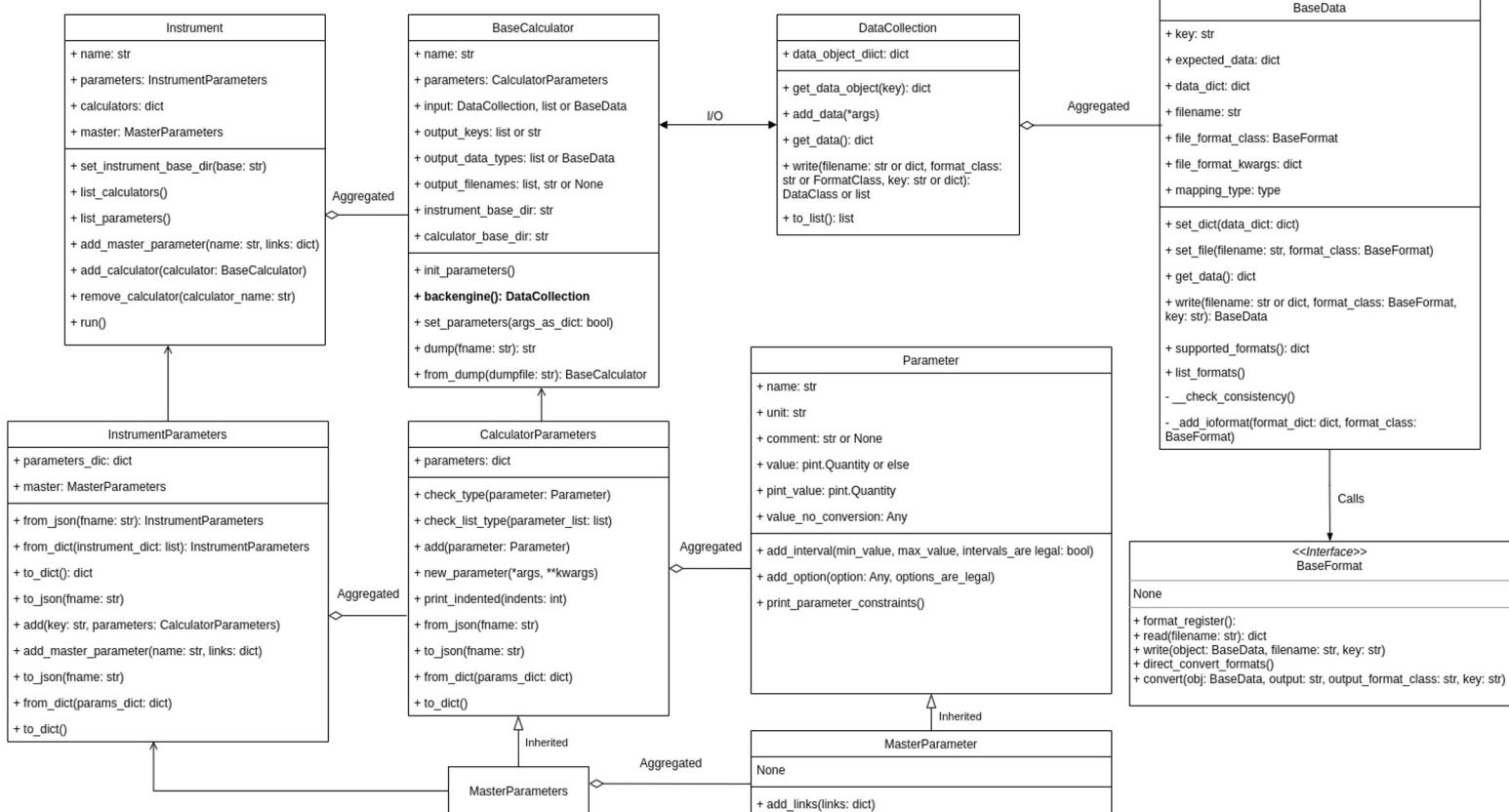
OASYS  
(GUI from docker)  
(Remote framework)

libpyviny

# libpyvinyl

Juncheng E, Shervin Nourbakhsh, Mads Bertelsen, Carsten Fortmann-Grote

## Libpyvinyl class overview



<https://github.com/PaNOSC-ViNYL/libpyvinyl>

A simple usage example:

<https://github.com/PaNOSC-ViNYL/libpyvinyl/tree/master/tests/integration/plusminus>

- Harmonize the user interfaces for **neutron** and **X-ray** simulation
- Base classes:
  - **BaseCalculator**
  - **BaseData**
  - **BaseFormat**
- Auxiliary classes:
  - **Parameter**
  - **CalculatorParameters**
  - **DataCollection**
  - **Instrument**
- It provides the developers of start-to-end simulation platforms in neutron and X-ray community with a framework to integrate various backengine software.
- It reduces the effort needed to integrate their software to a start-to-end simulation platform for simulation software developers.

# libpyvinyl

Juncheng E, Shervin Nourbakhsh, Mads Bertelsen, Carsten Fortmann-Grote

## *libpyvinyl is a dependency*

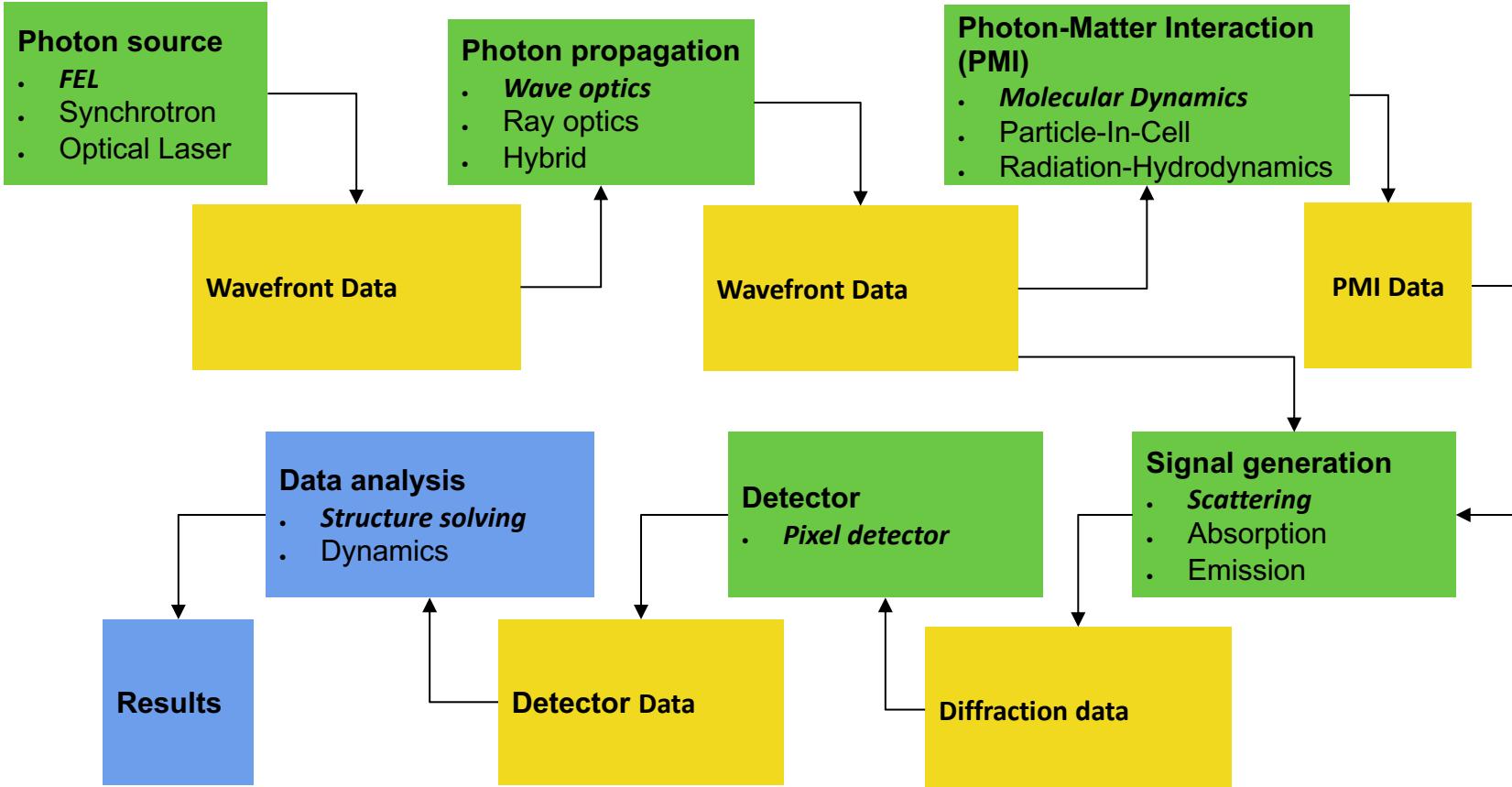
- libpyvinyl is a starting point for packages
- Provides base classes for
  - Parameters
  - Calculator
  - Data



# SimEx-lite

Juncheng E, Carsten Fortmann-Grote

## Package overview - Workflow



- It is the core package of the SIMEX platform providing the calculator interfaces and data APIs.
- It is built based on **libpyvinyl**. A **calculator** can be easily constructed within SimEx-Lite with the definition of the corresponding **data class** and the **format class**.
- Users can choose which backengine software to implement to make the installation minimal per needs.

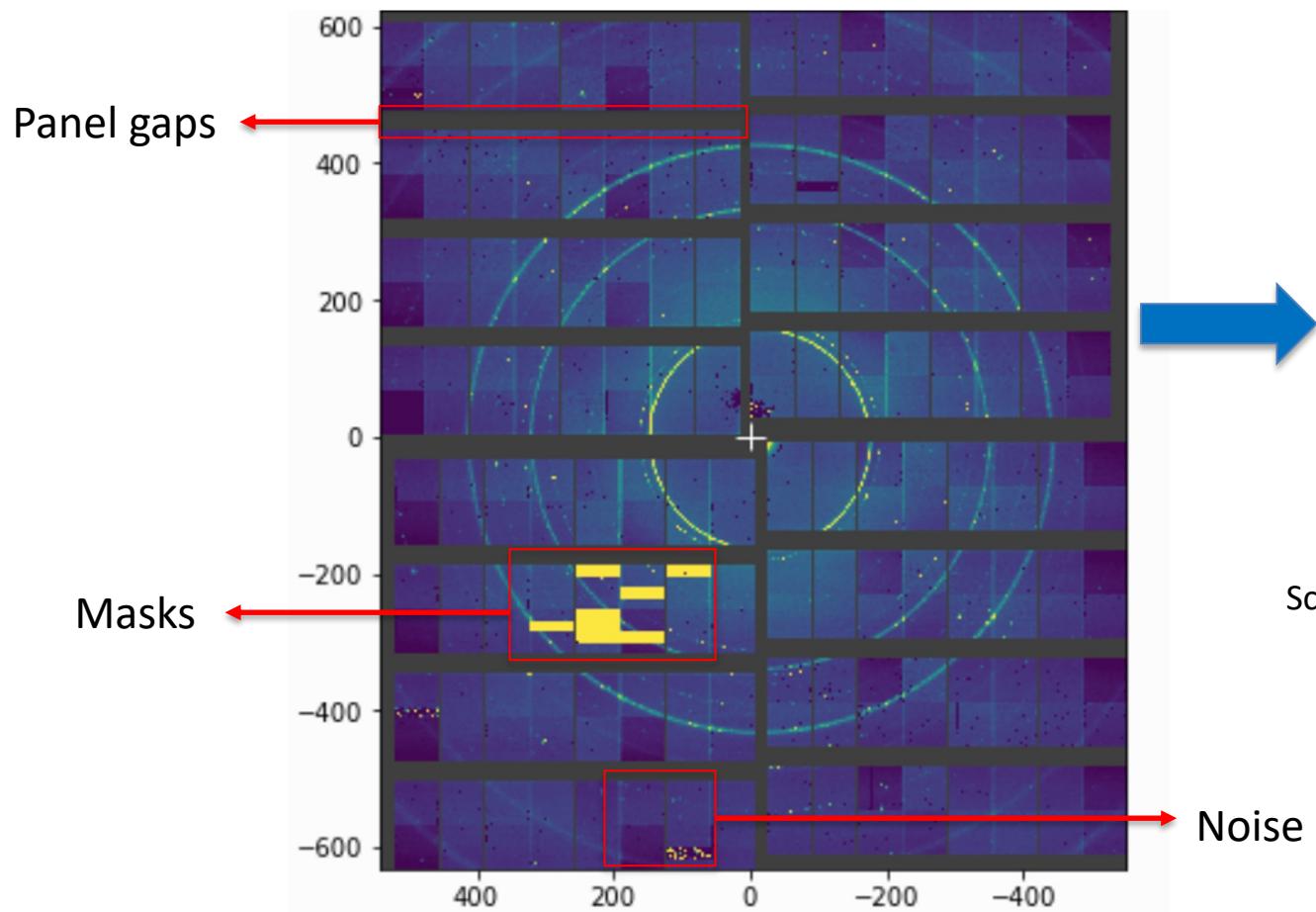
## Calculators

## Data interfaces

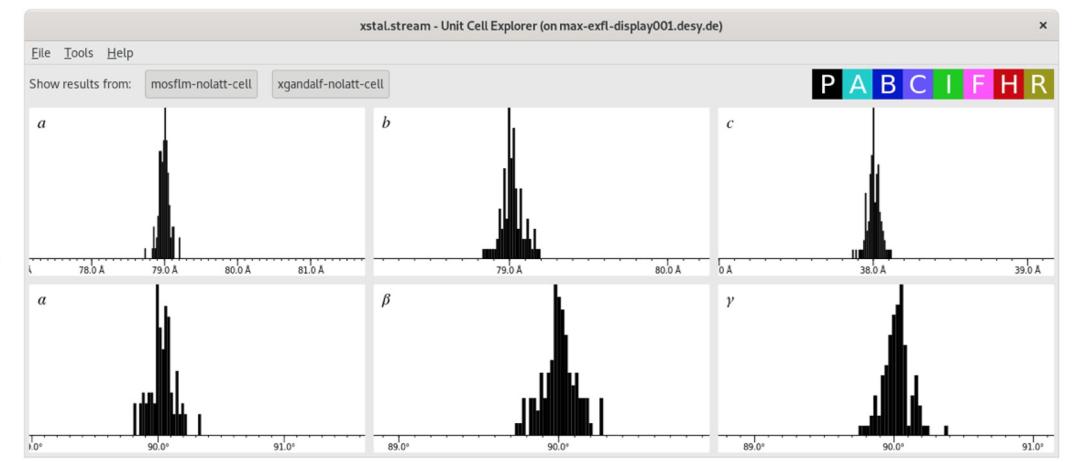
<https://github.com/PaNOSC-ViNYL/SimEx-Lite>

# SimEx-lite

## Serial Crystallography Simulation and Analysis



Demonstrate the effects of noise, panel gaps and masks on the results of crystallography analysis with SimEx simulation diffraction pattern results

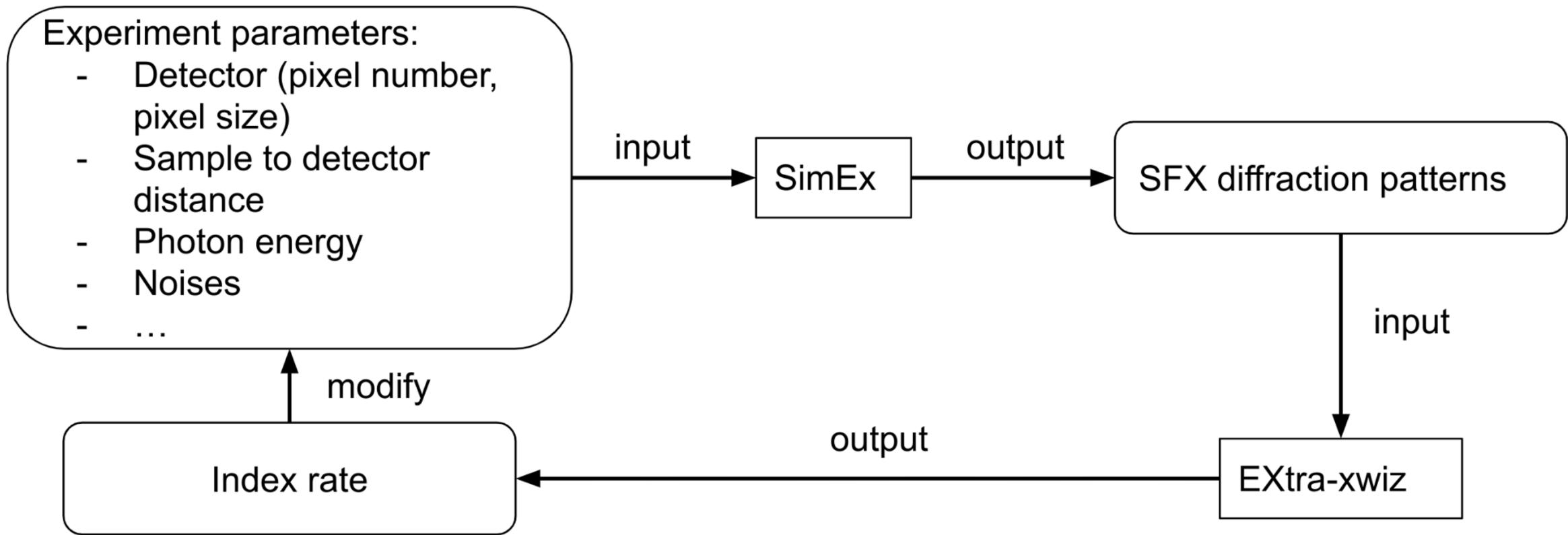


### Scenarios:

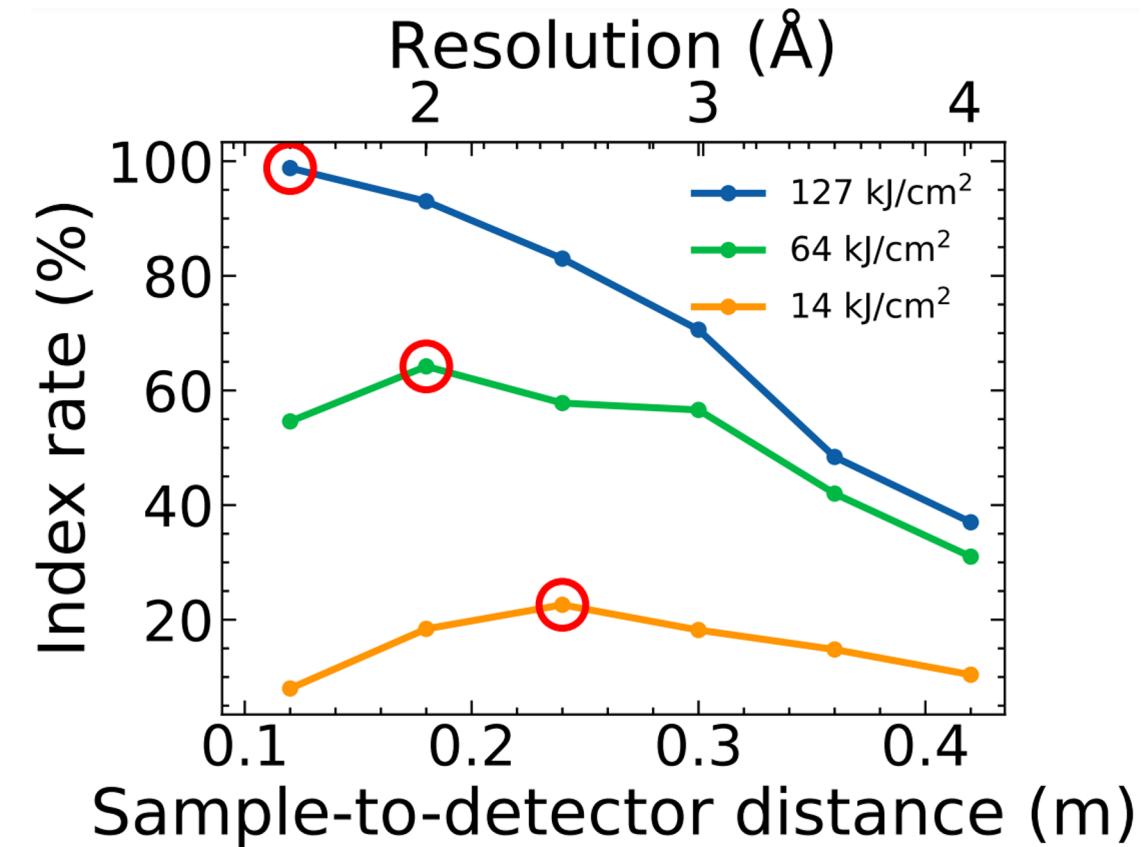
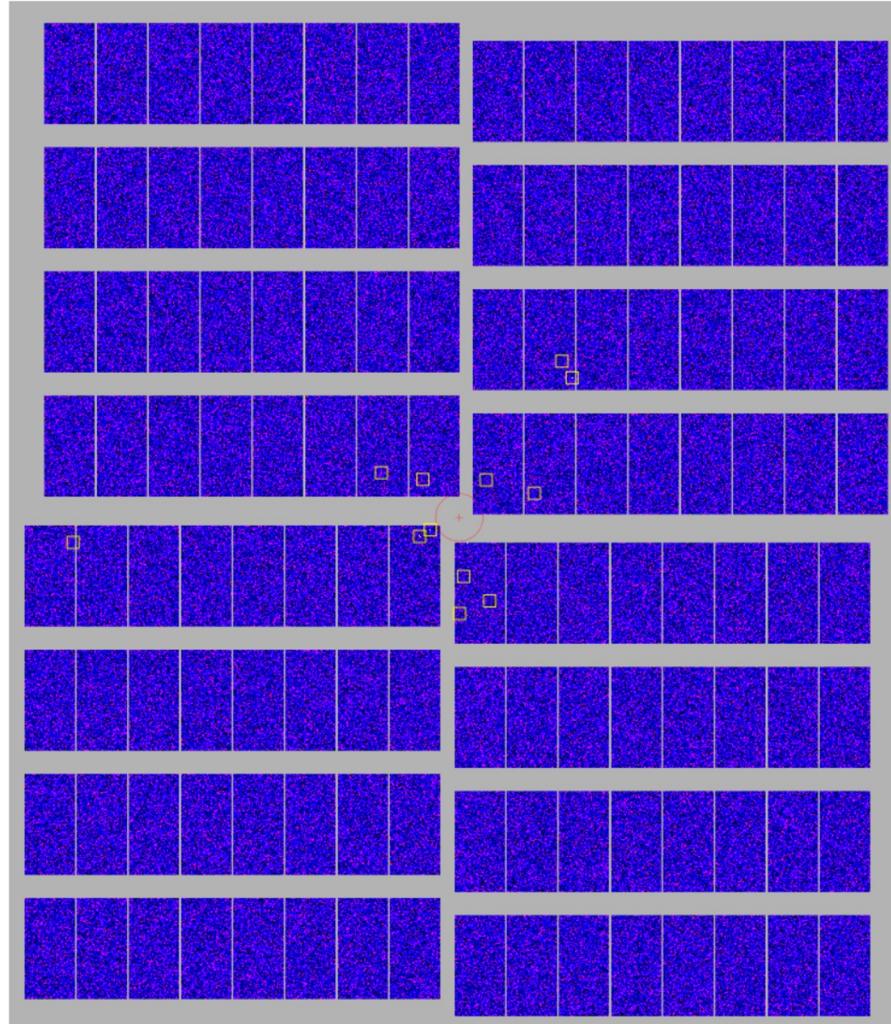
- New researchers want to get hands-on experience of data processing for serial crystallography.
- Starting from a working example of a simulation coupled to an analysis pipeline.
- Explore how different experimental conditions and different analysis options affect the results.

# SimEx-lite

## Optimize sample to detector distance in different energy densities



## Optimize sample to detector distance in different energy densities



## Scientific publications

Structural Dynamics      ARTICLE      scitation.org/journal/sdy

### Expected resolution limits of x-ray free-electron laser single-particle imaging for realistic source and detector properties ← Editor's pick

Cite as: Struct. Dyn. 9, 064101 (2022); doi:10.1063/4.0000169  
Submitted: 7 September 2022 · Accepted: 31 October 2022 ·  
Published Online: 16 November 2022



Juncheng E,<sup>1</sup> Y. Kim,<sup>1</sup> J. Bielecki,<sup>1</sup> M. Sikorski,<sup>1</sup> R. de Wijn,<sup>1</sup> C. Fortmann-Grote,<sup>1,2</sup> J. Sztuk-Dambietz,<sup>1</sup> J. C. P. Koliyadu,<sup>1</sup> R. Letrun,<sup>1</sup> H. J. Kirkwood,<sup>1</sup> T. Sato,<sup>1</sup> R. Bean,<sup>1</sup> A. P. Mancuso,<sup>1,3,a)</sup> and C. Kim<sup>1,b)</sup>

#### AFFILIATIONS

<sup>1</sup>European XFEL, Holzkoppel 4, 22869 Schenefeld, Germany

<sup>2</sup>Max Planck Institute for Evolutionary Biology, August-Thienemann-Straße 2, 24306 Plön, Germany

<sup>3</sup>Department of Chemistry and Physics, La Trobe Institute for Molecular Science, La Trobe University, Melbourne, Victoria 3086, Australia

<sup>a)</sup>Electronic mail: [adrian.mancuso@xfel.eu](mailto:adrian.mancuso@xfel.eu)

<sup>b)</sup>Author to whom correspondence should be addressed: [chan.kim@xfel.eu](mailto:chan.kim@xfel.eu)

The papers are based on the simulation of photon-matter interaction, scattering process and detector response within the framework of SIMEX.

## scientific reports



OPEN

### Effects of radiation damage and inelastic scattering on single-particle imaging of hydrated proteins with an X-ray Free-Electron Laser

Juncheng E<sup>1,2</sup>, Michal Stransky<sup>1,2</sup>, Zoltan Jurek<sup>3,4</sup>, Carsten Fortmann-Grote<sup>1,5</sup>, Libor Juha<sup>6,7</sup>, Robin Santra<sup>3,4,8</sup>, Beata Ziaja<sup>2,3</sup> & Adrian P. Mancuso<sup>1,9</sup>

# OASYS

Manuel Sanchez del Rio, Aljosa Hafner

## Project overview

- ✓ OASYS = OrAnge SYnchrotron Suite
- ✓ A common platform to build synchrotron-oriented User Interfaces ***that communicate***
- ✓ The upper layer of the application presented to the user
- ✓ Open Source & Python technology



Luca Rebuffi, Manuel Sanchez del Rio (2017)

**OASYS (OrAnge SYnchrotron Suite) : an open-source graphical environment for x-ray virtual experiments**

Proc.SPIE 10388: 10388-10388. <http://dx.doi.org/10.1117/12.2274263>

<https://oasys-kit.github.io/>



Elettra Sincrotrone Trieste



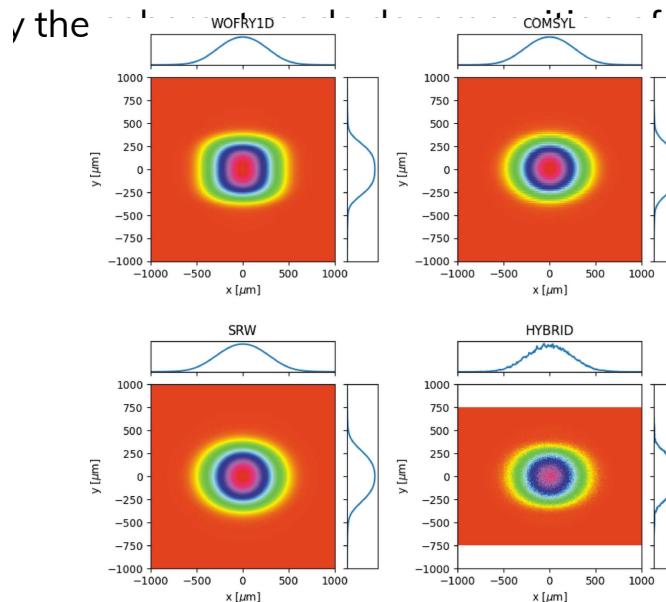
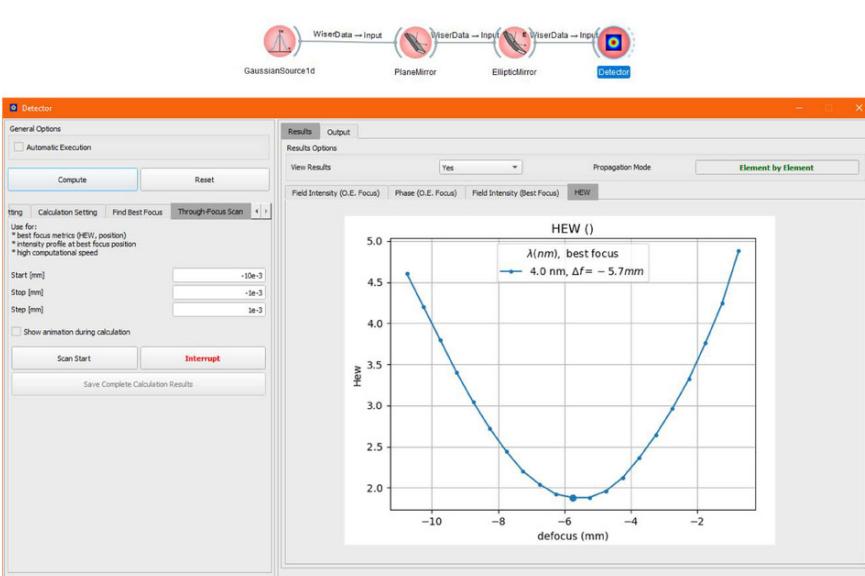
## Wiser and COMSYL

Main code developments - Require high-performance computing resources and benefit from remote deployments:

- **Wiser** code for numerical integration of wave propagation:

Allows for precise metrology simulations, correctly taking into account shape and roughness contributions at various length scales.

- **COMSYL** (COherent Modes for SYnchrotron Light):



y the indulator radiation in a storage ring.

- M. Manfredda et al. Advances in Computational Methods for X-Ray Optics V. Vol. 11493. International Society for Optics and Photonics. SPIE, 2020, 114930B. DOI: 10.1117/12.2568574
- M. Glass et al. EPL (Europhysics Letters) 119.3 (Aug. 2017), p. 34004. DOI: 10.1209/0295-5075/119/34004
- M. Sanchez del Rio et al. Journal of Synchrotron Radiation 29.6 (Nov. 2022), pp. 1354–1367. DOI: 10.1107/S1600577522008736.

# OASYS

## Remote usage

**Main challenge:** OASYS is a GUI program built in PyQt. How to run it in the browser? As a **Jupyter hub plug-in!**

- **Deployed easily as a Docker container** and can be used within an existing Jupyter hub instance
- **Remote repository workspace downloader has been developed as a widget**
- Using <https://hub.gke2.mybinder.org> for the first time was successful

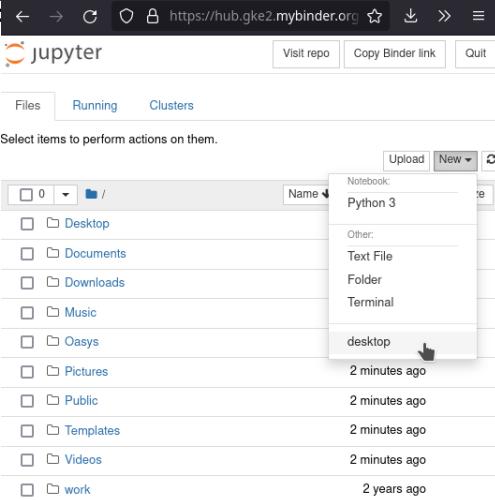


Figure: Select and run a desktop environment from within Jupyter hub.

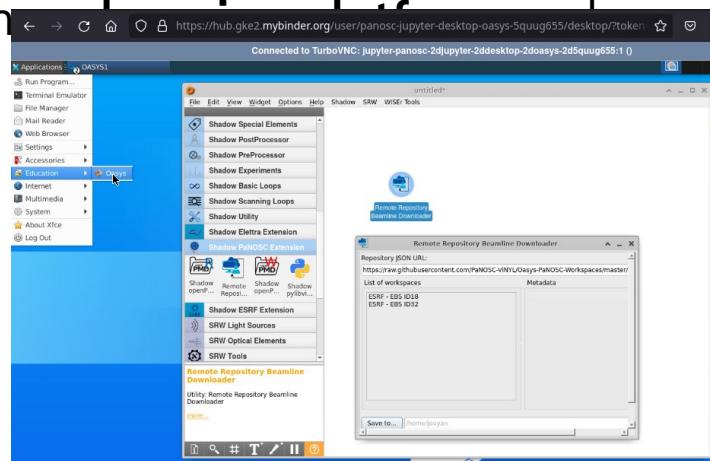


Figure: OASYS GUI environment inside a web browser.

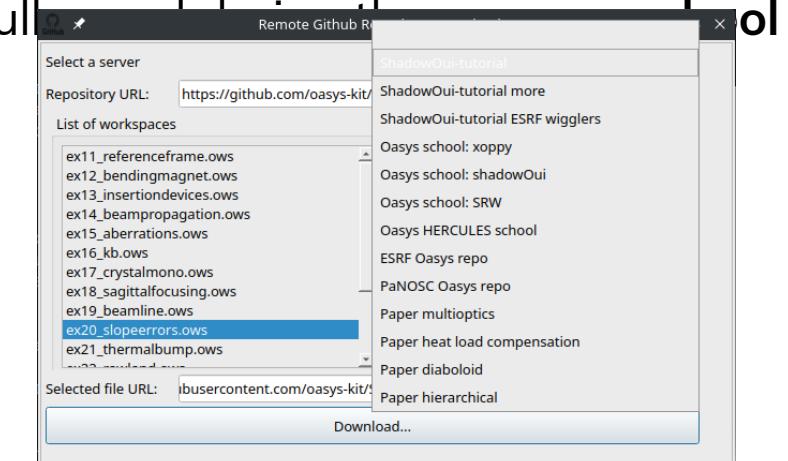


Figure: Remote repository downloader widget showing a selection of curated workspaces available.

# OASYS

## Other developments – openPMD and link to SimEx

- Deliverable D5.1: contributions to openPMD standard for ray-tracing
- Synergy of WP5 simulation codes: link between diffraction simulations and beamline optics
  - OASYS-SimEx Widget developed and available
  - OASYS PaNOSC toolbox contains all the tools developed in the project and is available for installation through the built-in add-on manager as OASYS1\_PaNOSC

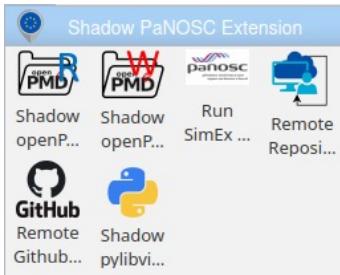


Figure: OASYS PaNOSC toolbox.

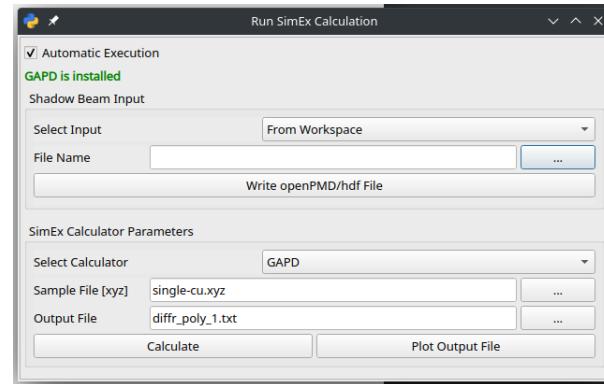


Figure: SimEx interaction widget showing the GAPD calculator.

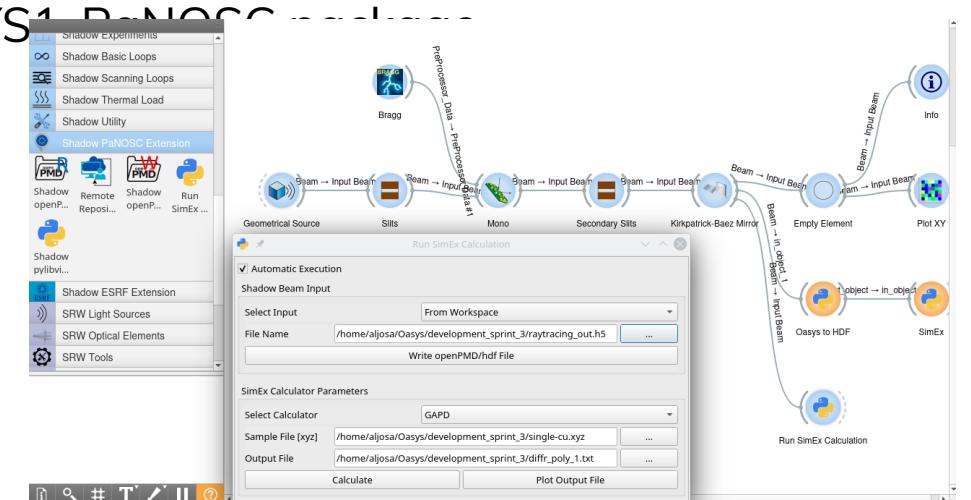
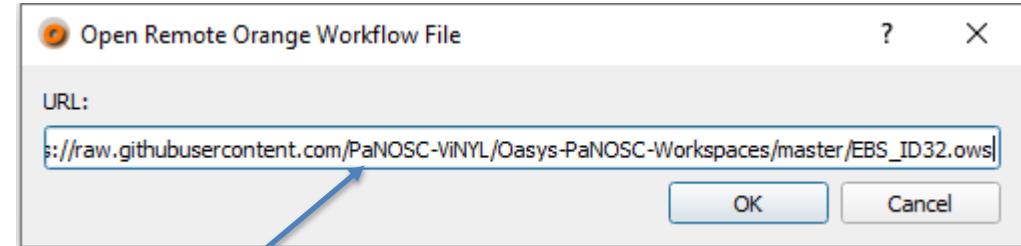
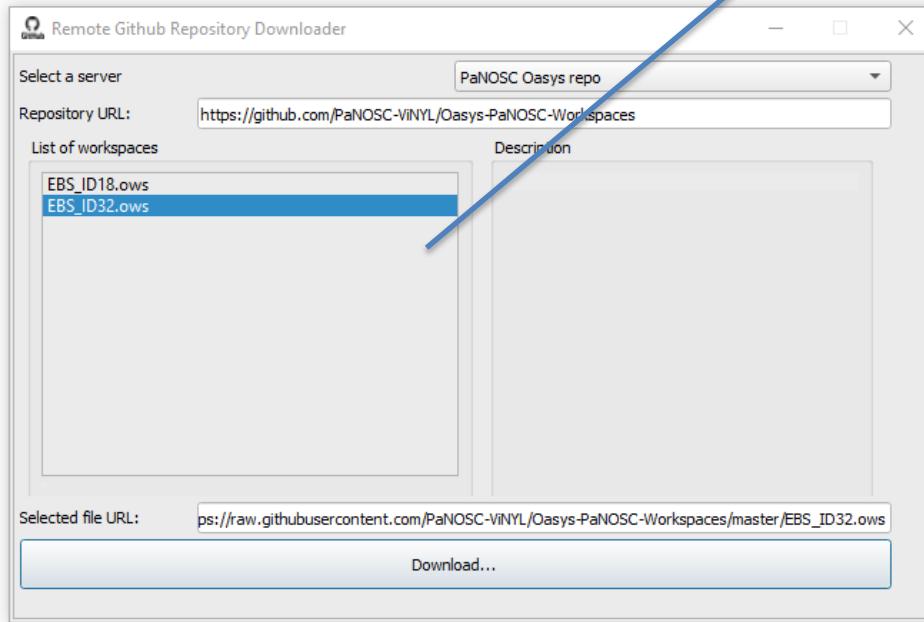
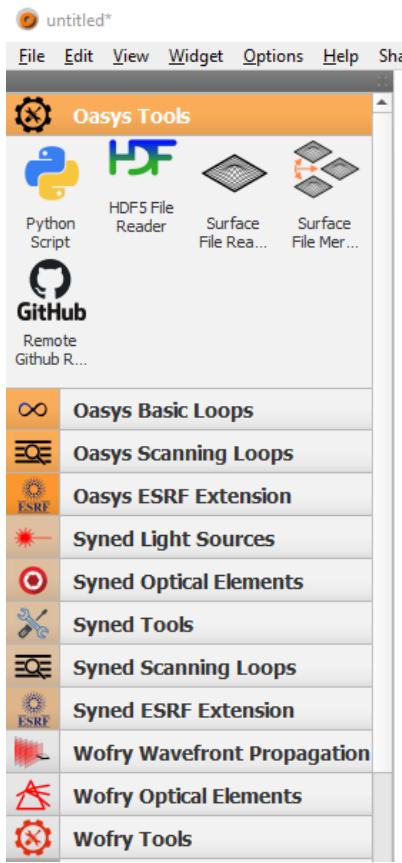


Figure: Seamless workflow from beamline optics to diffraction simulations within the OASYS GUI.

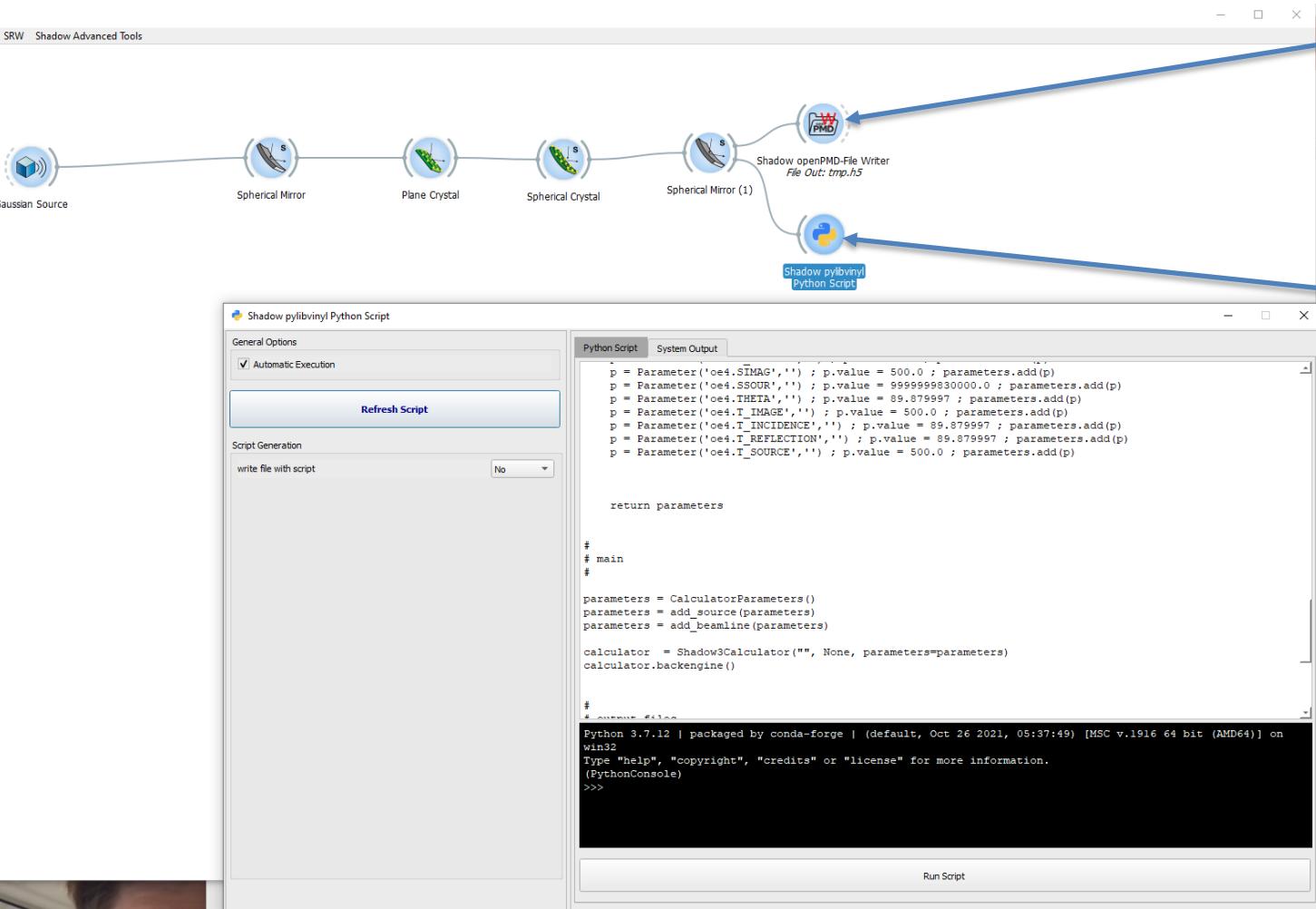
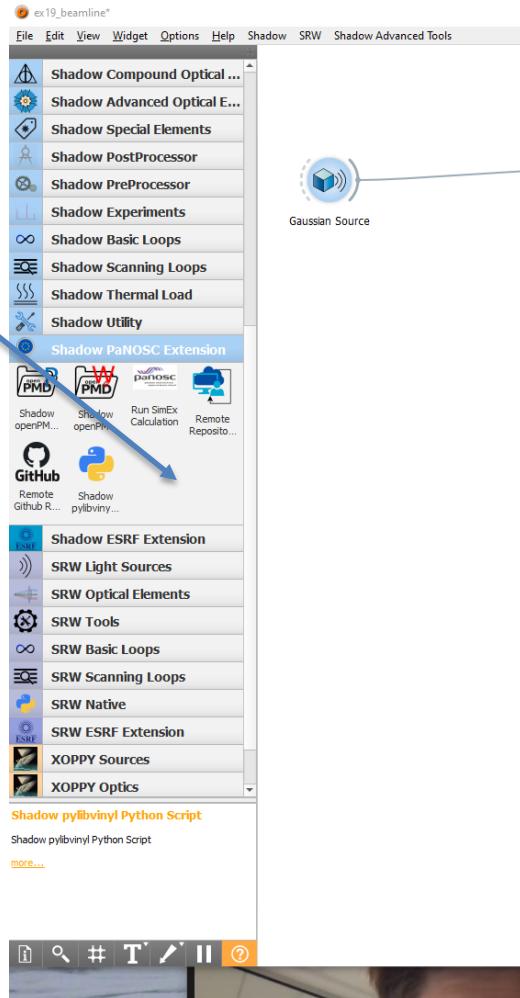
# OASYS

## Remote access of workspaces



## PaNOSC tools for SHADOW simulations

PaNOSC add-on



# OASYS

## E-learning course on pan-training.eu

The screenshot shows a Moodle-based e-learning course titled "Introduction to Oasys (Hercules 2022)". The course navigation menu on the left includes sections like Participants, Grades, Outline, Introduction, Power Transport, How to open remote OASYS-workflows, Photon Transport, Coherence Transport, Additional Material, Acknowledgements, e-Learning, Dashboard, Calendar, Content bank, Quiz Taster, Introduction to Muon Spin Spectroscopy, and Advanced Topics in. The main content area features a network diagram, the title "Introduction to Oasys (Hercules 2022)", and an "Outline" section. The outline lists the course structure: Welcome, Aim, Objectives, Sections (Introduction, Power Transport, Photon Transport, Coherence Transport), and a note about using OASYS in two ways. It also provides links for installing OASYS on Mac OS, Windows OS, Linux OS, and Docker.

Introduction to Oasys (Hercules 2022)

e-Learning | My courses | Introduction to Oasys (Hercules 2022)

### Outline

Welcome to the Hercules 2022 Oasys tutorial written by Manuel Sanchez del Rio and Juan Reyes Herrera.

The aim of this course is to learn the following:

- Calculate main characteristics of synchrotron source (Bending magnets and Insertion devices).
- Calculate the heat-load on different beamline components.
- Simulating beamline optics by ray-tracing to obtain main parameters of the beam, e. g., size and divergence, energy resolution, intensity/flux.
- Understand basic principles of X-ray optics: Mirrors and Crystals.
- Basic concepts about coherence.

And it is split into four sections:

1. Introduction
2. Power Transport
3. Photon Transport
4. Coherence Transport

Before going to the content, it is important to explain that you can use OASYS in two ways:

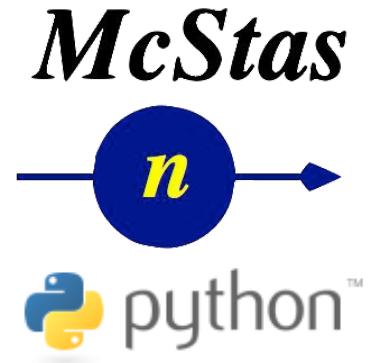
1. Install Oasys on your own laptop. Follow the instructions in the following links:  
Mac OS   Windows OS   Linux OS   Docker

This course needs the installation of the ESRF add-on in OASYS. Open OASYS and follow the "install as user" instructions here.

# McStasScript

Mads Bertelsen

## Overview



- API for popular instrument simulation tool McStas
  - Describe instrument to be simulated
  - Get help and overviews
  - Run simulations
  - View data
  - Get data as numpy arrays
  - Use widgets

```
guide = instr.add_component("guide", "Guide_gravity")
guide.set_parameters(w1=0.05, h1=0.05, m=2, G=-9.82, l=10)
guide.set_AT(2.0, RELATIVE=src)

slit = instr.add_component("slit", "Slit")
slit.radius = 0.03
slit.set_AT(guide.l + 0.2, RELATIVE=guide)

sample = instr.add_component("sample", "PowderN")
sample.set_AT(0.1, RELATIVE=slit)
sample.set_parameters(radius=0.015, yheight=0.05,
                      reflections='"Na2Ca3Al2F14.laz"')

data = instr.backengine()
```

# McStasScript

## Documentation

- Documentation
  - Install guide - pip
  - Configuration – set a few paths
  - Quick guide
  - Tutorials
  - Reference
- All as jupyter notebooks

### McStasScript documentation

Search the docs ...

#### GETTING STARTED

- Overview
- Installation
- Version history
- Quick start

#### USER GUIDE

- Instrument object
- Component object
- Parameters and variables
- Data
- Plotting
- Functions
- Widgets
- Instrument reader

#### MCSTASSCRIPT TUTORIAL

- McStasScript introduction
- Advanced McStas features: SPLIT
- Advanced McStas features: EXTEND and WHEN
- Advanced McStas features: JUMP
- Dynamic instrument cuts with MCPL bridges

#### MCSTAS UNION TUTORIAL

- The Union components
- Advanced geometry using the Union components
- Visualizing what happens in Union master
- Using conditional component to modify loggers

before="master")  
outer\_wall\_vac.set\_AT([0,0,0], RELATIVE=sample\_geometry)  
outer\_wall\_vac.yheight = 0.15 - 0.01  
outer\_wall\_vac.radius = 0.1 - 0.003  
outer\_wall\_vac.material\_string = "Vacuum"  
outer\_wall\_vac.priority = 61

### Adjusting the logger view to see the larger cryostat area

The loggers were only viewing a small area around the sample can, but this can be expanded as we still have access to the component objects.

```
logger_zx.set_parameters(D1_min=-0.12, D1_max=0.12, D2_min=-0.12, D2_max=0.12)  
logger_zy.set_parameters(D1_min=-0.12, D1_max=0.12, D2_min=-0.12, D2_max=0.12)  
logger_xy.set_parameters(D1_min=-0.12, D1_max=0.12, D2_min=-0.12, D2_max=0.12)
```

### Viewing the instrument

The instrument can be viewed with the `show_instrument` method. The mock cryostat and detector can be seen in a 3D view.

```
instrument.show_instrument()
```

/Users/madsbertelsen/PaNOSC/McStasScript/github/McStasScript/docs/source/tutorial/run\_folder/pythonTutorial.instr --no-output-files --trace --ncount=300 --dir=pythonTutorial\_mcdisplay\_16 wavelength=3.0

Previous Pause Next Ray index 74 / 299 Keep rays  Scatter Markers

Reset view: Home Side Top Show BB

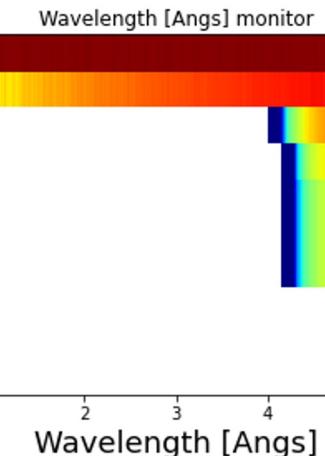
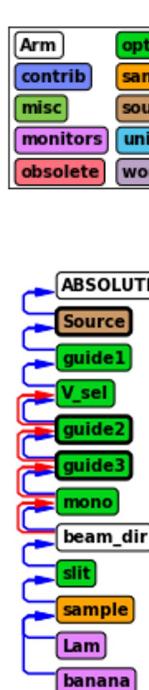
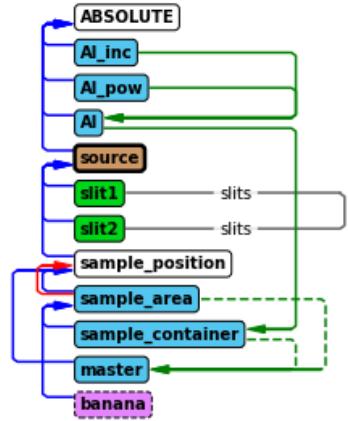
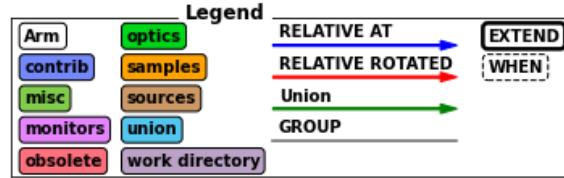
Run the updated instrument file.

Contents

- Setting up some standard materials
- Set up source
- Describing the geometry of a simple cryostat
- First geometry, a sample in a container
- Set up loggers to check what is going on
- Add master component
- Add banana monitor
- Run simulation
- Plotting the data
- Interpretation of results
- Adding a cryostat around the sample can
- Adjusting the logger view to see the larger cryostat area
- Viewing the instrument
- Run the updated instrument file
- Plot the data from the new simulation
- Interpreting the data
- Comparing situation with and without cryostat

# McStasScript

## Diagrams and widgets



### Running the simulation

The simulation can now be performed from the Jupyter Notebook using the widget interface.

In [24]: %matplotlib widget  
Instr.interface()

energy 10.0 // [meV] Energy of source  
delta\_energy 8.0 // [meV] Energy spread of source  
rotation\_y 180.0 // [deg] Rotation around vertical  
rotation\_x 0.0 // [deg] Rotation around horizontal  
material Pb // Material choice for extra material sample



Run

ncount 1E8

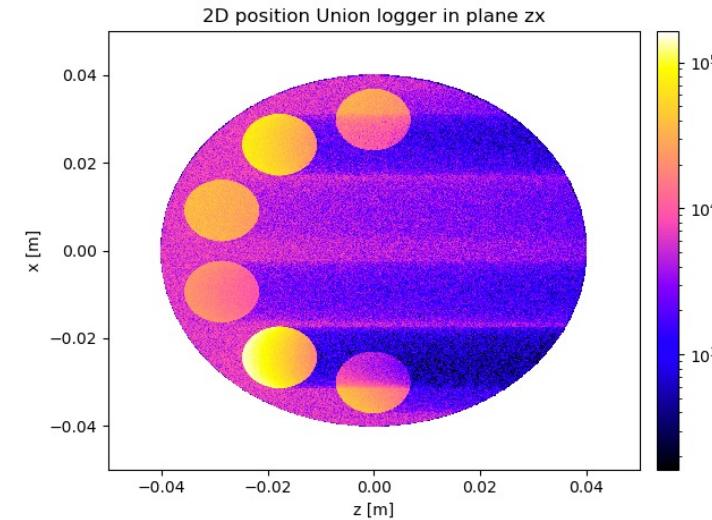
mpi 6

 Live results

Sim progress



Figure 1



Choose monitor

logger\_space\_zx\_all

Plot options

Log plot



Orders of magnitude

3

Colormap category

Miscellaneous

gnuplot2

# Instrument repository

Shervin Nourbakhsh

## Overview

GIT stores instrument descriptions in the form a Python scripts following the libpyvinyl API.

Advantages of GIT:

- History of changes
- Integrated issue tracking system
- Automatic workflows for validation
- Openly accessible
- Free hosting
- Easy to clone, fork for special needs of the RIs

## Sustainability

- Instrument description maintained by instrument experts at Ris
- Users contributing with validations and debugging

A dedicated python API is developed to allow end users (especially those not knowing git) to access the instrument description.

# Instrument repository

## Instrument repository API

- ▶ Setup the API

```
from instrumentdatabaseapi import instrumentdatabaseapi as API
repo = API.Repository()
repo.init()
```

- ▶ Load the instrument and units

- ▶ Institute name
- ▶ Instrument name
- ▶ Version: HEAD for the current or date of last day of validity  
YYYY-MM-DD
- ▶ Flavour: if different alternative descriptions are possible (e.g. detailed vs simplified)

```
myinstrument = repo.load("ILL", "ThALES" , "HEAD", "mcstas", "full", dep=
                           False)
import pint
ureg = pint.get_application_registry()
```

- ▶ Set simulation settings

```
myinstrument.set_instrument_base_dir("/tmp/ThALES_scan/")
myinstrument.sim_neutrons(500000)
myinstrument.set_seed(654321)
```

# Instrument repository

## Instrument repository API

- ▶ Set instrument parameters:

```
myinstrument.master["a2"] = myinstrument.energy_to_angle(4.98 * ureg.meV)
myinstrument.master["a4"] = 60 * ureg.degree
```

- ▶ Set sample parameters:

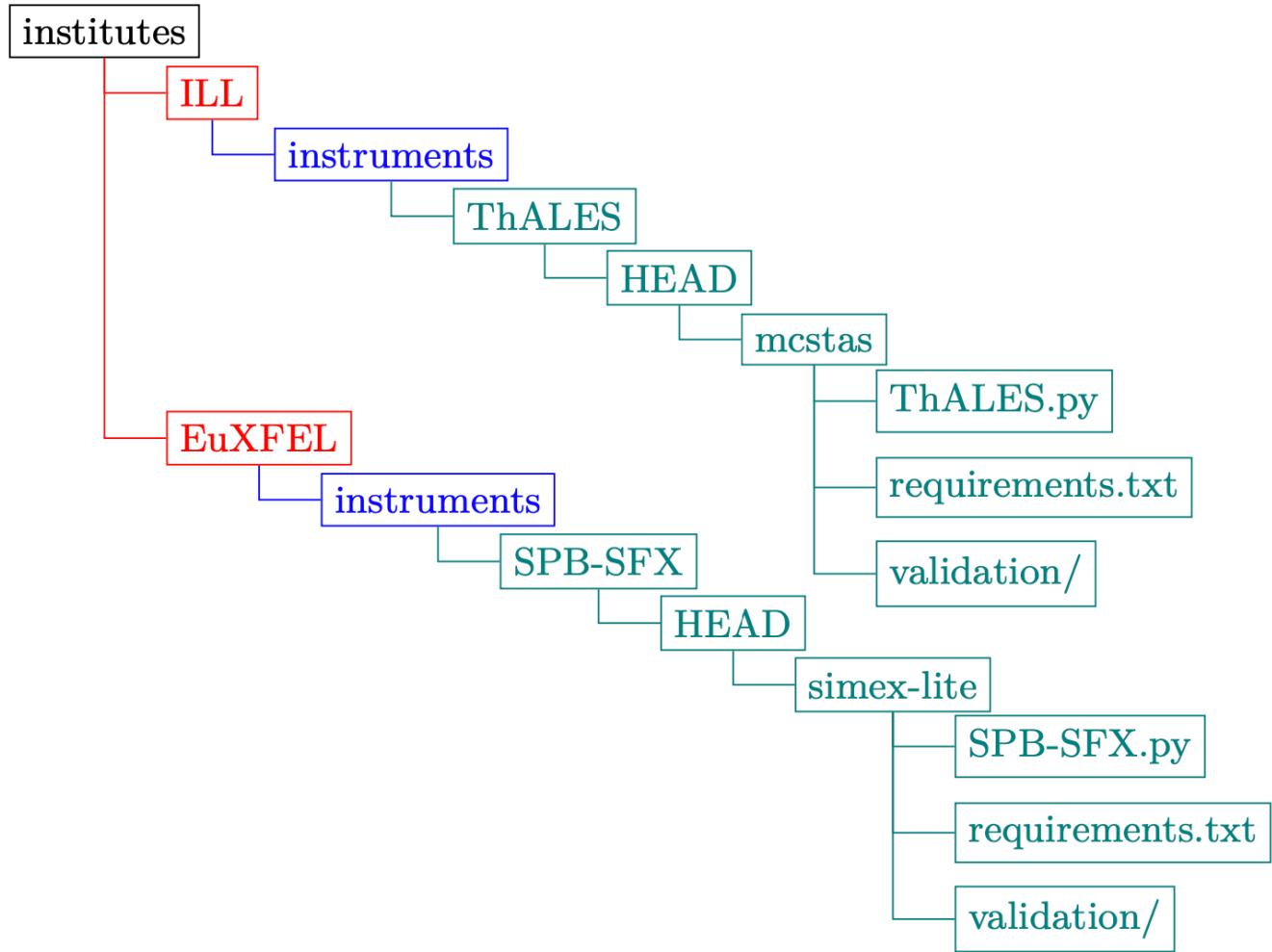
```
myinstrument.set_sample_by_name("vanadium")
myinstrument.sample_cylinder_shape(0.005, 0.01)
```

- ▶ Run simulations

```
np = (21 - 1) / 2
dEI = 0.05
import numpy
myinstrument.force_compile(false)
outputs = []
for energy in numpy.arange(4.98 - np * dEI, 4.98 + np * dEI, dEI):
    myinstrument.master["a6"] = myinstrument.energy_to_angle(energy * ureg.
                                                          meV)
myinstrument.run()
outputs.append(mynstrument.output)
```

# Instrument repository

## Instrument repository structure



# Digital Twin

Shervin Nourbakhsh

## Overview

### General idea of a Digital Twin

It is a digital replica of the physical elements of an instrument that can be controlled and provide feedback as the real instrument.

It can be implemented at different levels of fidelity to replicate the features that one is interested in.

### Ingredients of Digital Twin at ILL

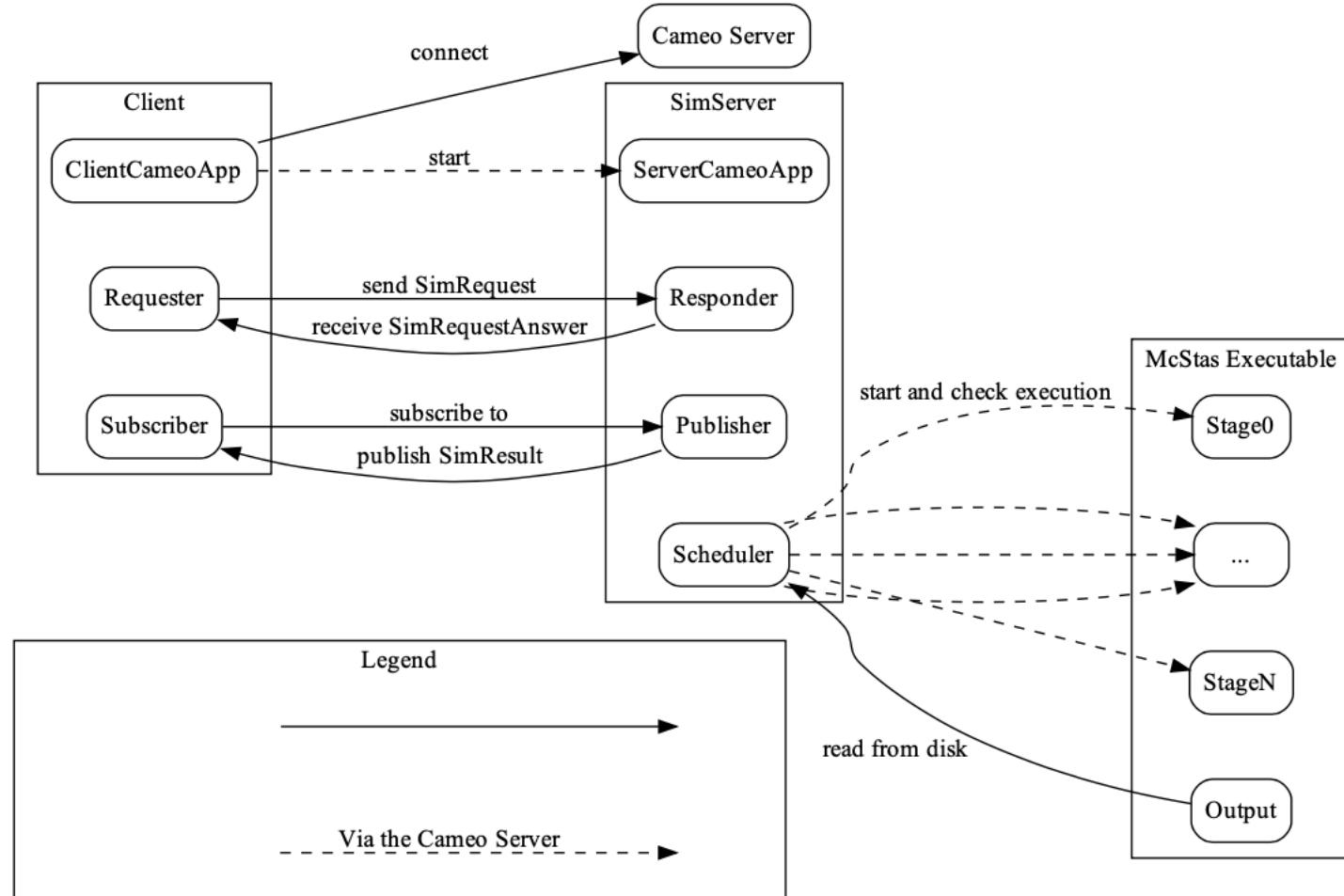
- An instrument fully described in the *instrument database*
- The physics of the sample provided by the user via Sqw files or other formats
- A set of sample environments fully described
- The data acquisition system (Nomad) used for both real and simulated data
- An application manager (Cameo) providing also basic communication patterns

A simulation server:

- processing simulation requests
- scheduling and parallelizing the simulation
- reading and sending the results to Nomad

# Digital Twin

## ILL system overview



# Digital Twin

## Results in Nomad

Nomad - You own control over Nomad. Click on the padlock to give it back.

File View Hardware Settings Command Editor Spy User Zoom Help

Internal use :

Hardware Settings Execution

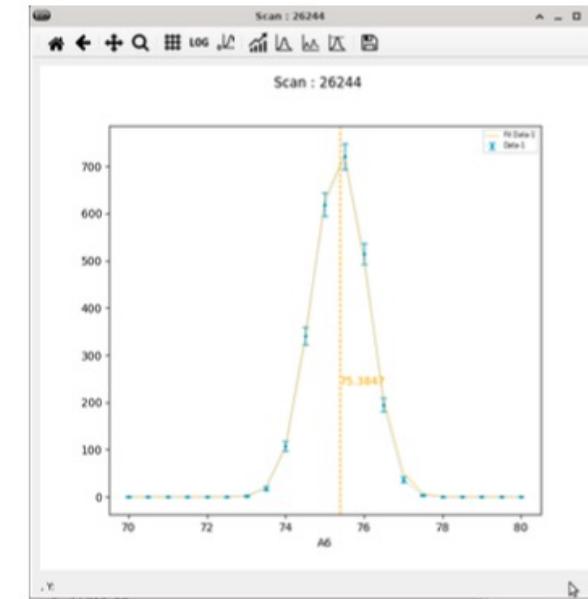
Launch pad

Command> sc a6 75 da6 0.5 np 21

Scan	Numor	started at	16:51:44			
Scan	Step	A6	Time	M1	M2	CNTS
Scan	1	70.00	0.00	0	0	0
Scan	2	70.50	0.00	0	0	0
Scan	3	71.00	0.00	0	0	0
Scan	4	71.50	0.00	0	0	0
Scan	5	72.00	0.00	0	0	0
Scan	6	72.50	0.00	0	0	0
Scan	7	73.00	0.00	0	0	2
Scan	8	73.50	0.00	0	0	18
Scan	9	74.00	0.00	0	0	108
Scan	10	74.50	0.00	0	0	341
Scan	11	75.00	0.00	0	0	619
Scan	12	75.50	0.00	0	0	721
Scan	13	76.00	0.00	0	0	515
Scan	14	76.50	0.00	0	0	195
Scan	15	77.00	0.00	0	0	36
Scan	16	77.50	0.00	0	0	3
Scan	17	78.00	0.00	0	0	0
Scan	18	78.50	0.00	0	0	0
Scan	19	79.00	0.00	0	0	0
Scan	20	79.50	0.00	0	0	0
Scan	21	80.00	0.00	0	0	0
Scan	Numor	26244	finished at	16:53:35		
Scan	SCALE	16 COUNTS/UNIT				
Scan						
Scan	1	70.00000	0	*	-	-
Scan	2	70.50000	0	*	-	-
Scan	3	71.00000	0	*	-	-
Scan	4	71.50000	0	*	-	-
Scan	5	72.00000	0	*	-	-

Setting view\_state to 1

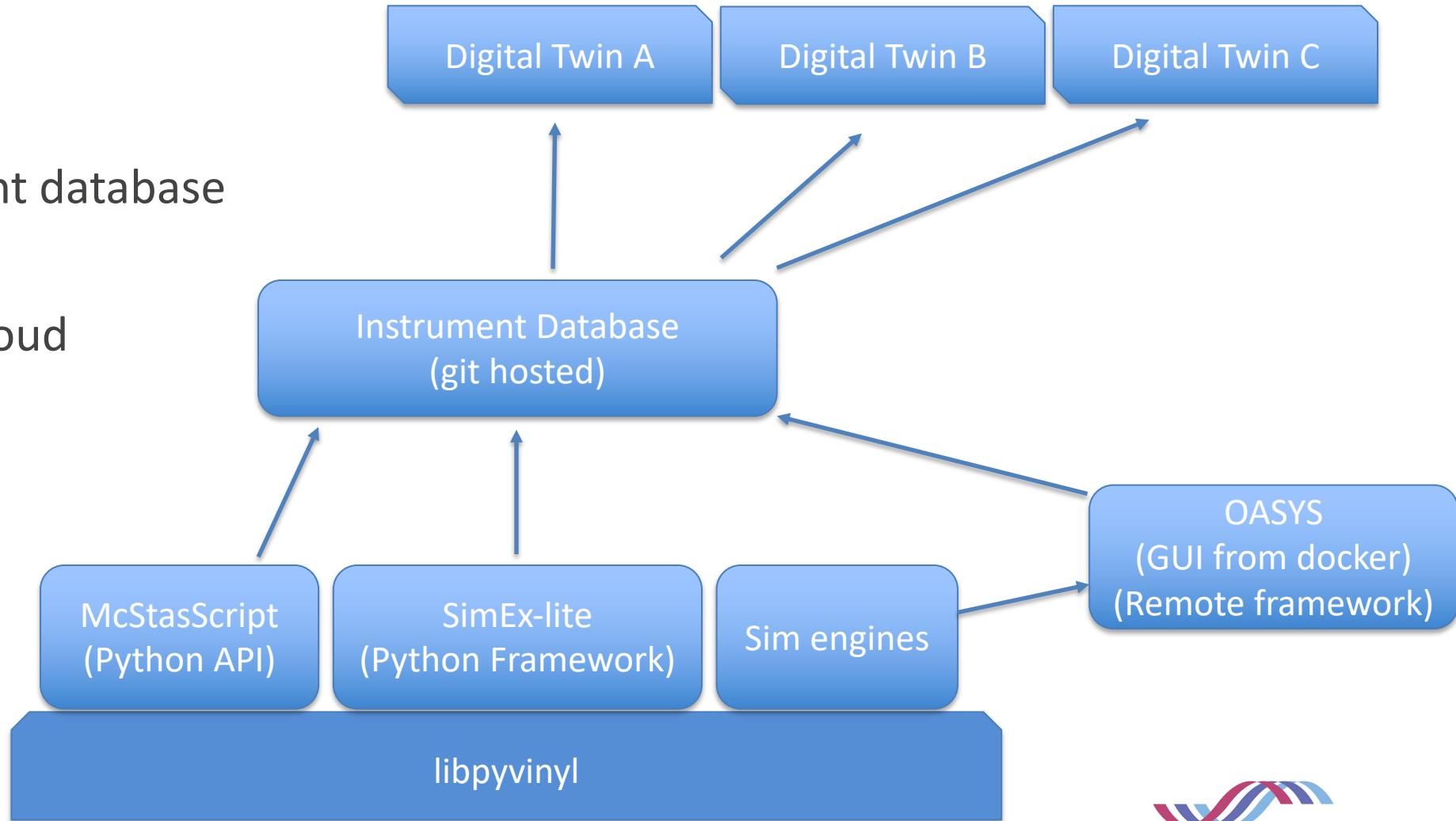
Stop  Pause 



# Summary

## Result of WP5 - ViNYL

- Common instrument database
- Interoperability
- Can be hosted in cloud



# Sustainability

## Future for the developed software

- All packages have developers interested in continuing support
- Reduced level compared to PaNOSC

Package	Current	After PaNOSC
libpyvinyl	Juncheng, Shervin, Mads, Carsten	Juncheng, Mads, Carsten
Instrument DB	Shervin	Mads
McStasScript	Mads	Mads
SimEx	Juncheng, Carsten	Juncheng, Carsten
Oasys	Aljosa, Manuel	Manuel



# PaNOSC Closing Event

Paving the way towards the PaN FAIR Data Commons

29-30 November 2022

Grenoble - France

## Thank you

Mads Bertelsen

[Mads.Bertelsen@ess.eu](mailto:Mads.Bertelsen@ess.eu)



PaNOSC has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement no. 823852