



Elettra Sincrotrone Trieste



Elettra
Sincrotrone
Trieste



ShadowOui: a new G.U.I. for Shadow3

&

**OASYS: a new environment for
synchrotron radiation simulations**

Luca Rebuffi, Elettra-Sincrotrone Trieste (ITA)



Elettra
Sincrotrone
Trieste



1. Shadow and Orange



Elettra
Sincrotrone
Trieste



A modern Ray-tracing tool

RAY-TRACING ENGINE

SHADOW3

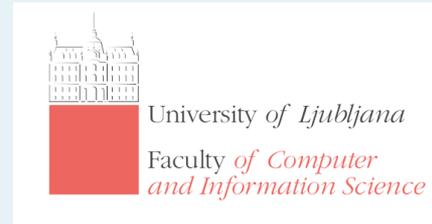
<http://forge.epn-campus.eu/projects/shadow3>



GUI AND DATA-FLUX ENGINE



<http://orange.biolab.si>



COMBINING POWERFUL TOOLS
TOGETHER

M. Sanchez del Rio, N. Canestrari, F. Jiang, F. Cerrina, "SHADOW3: a new version of the synchrotron X-ray optics modelling package", J. Synchrotron Rad. (2011), 18, 708–716

J. Demšar, B. Zupan, "Orange: From Experimental Machine Learning to Interactive Data Mining", White Paper (www.ailab.si/orange), Faculty of Computer and Information Science, University of Ljubljana(2004)

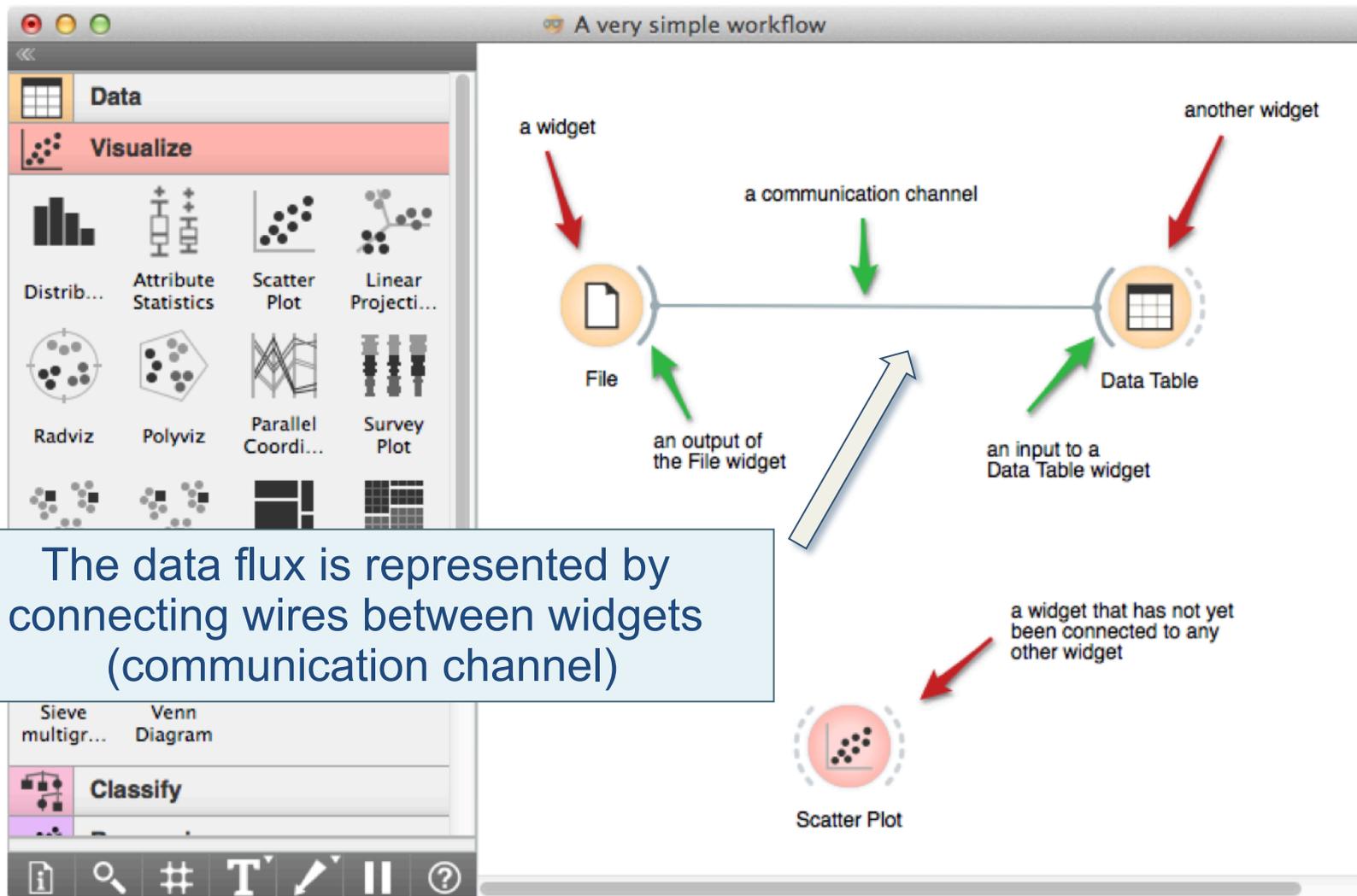


Elettra
Sincrotrone
Trieste



GUI powered by ORANGE

ORANGE is a python based software, representing objects containing data and procedures, as widgets in a desktop, exchanging data.



The data flux is represented by connecting wires between widgets (communication channel)

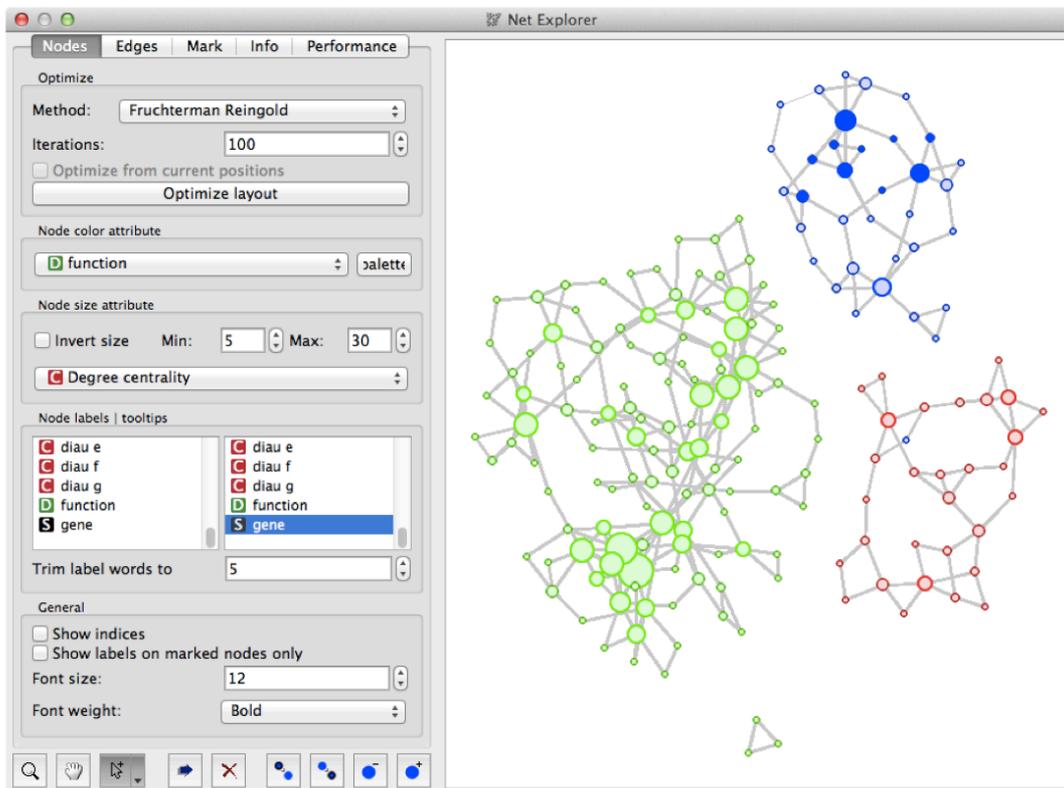
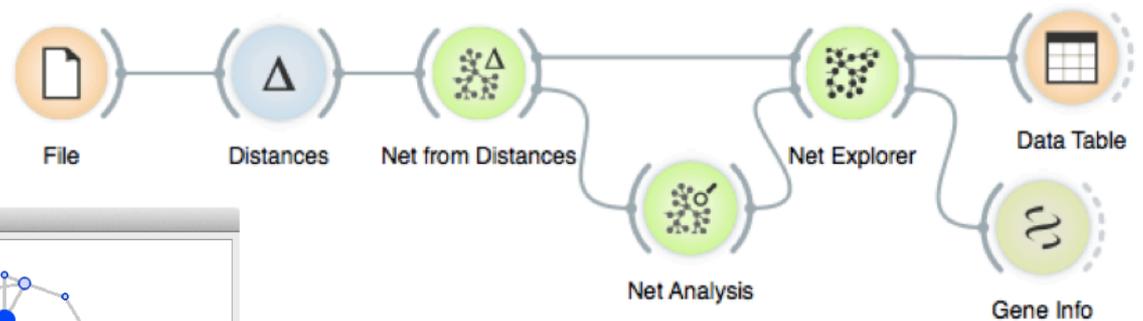


Elettra
Sincrotrone
Trieste



GUI powered by ORANGE

ORANGE is dedicated to data-mining and computer learning, and allows the user to build data-mining workflow with a friendly visual approach



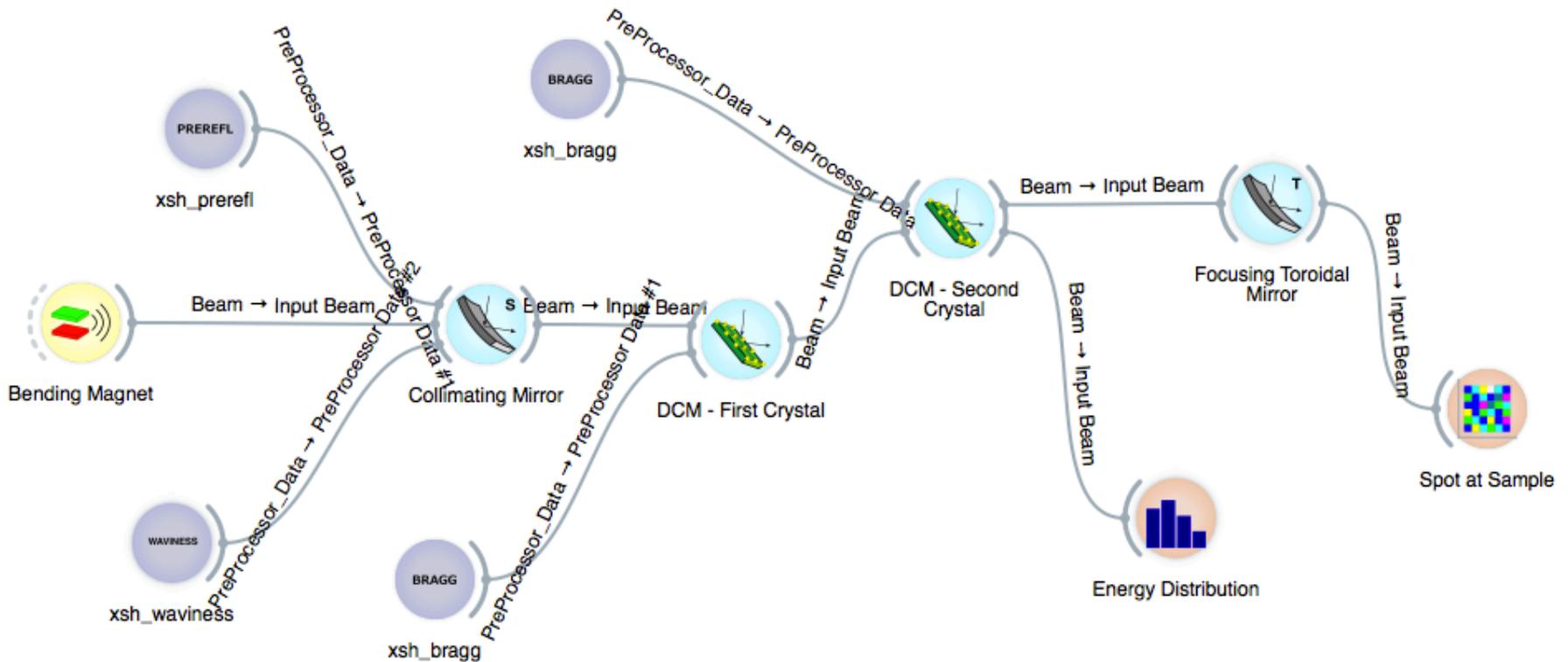


Elettra
Sincrotrone
Trieste



GUI powered by ORANGE

ORANGE infrastructure, engine and concept quite naturally adapts to describe a synchrotron radiation beamline, and in particular a radiation transport simulation!



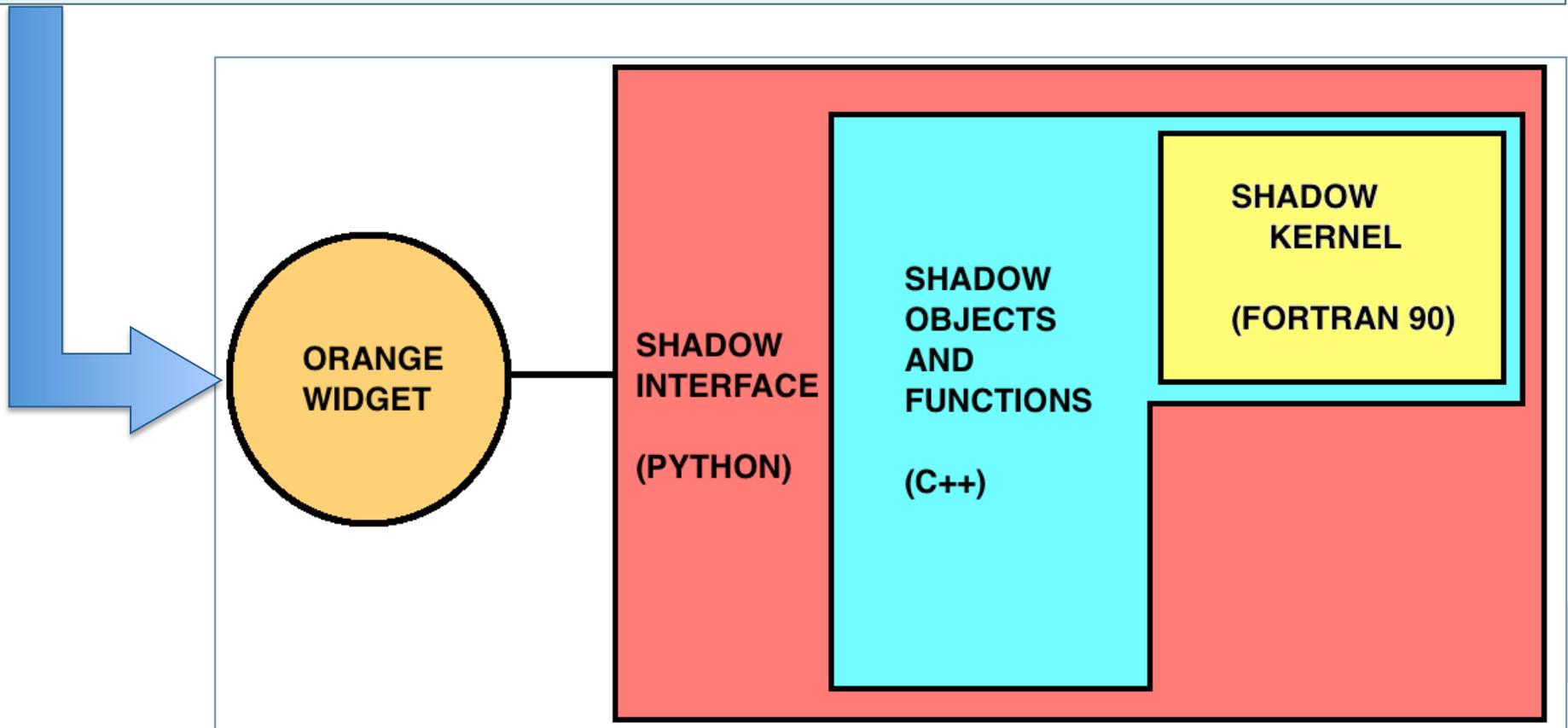


Elettra
Sincrotrone
Trieste



GUI powered by **ORANGE**

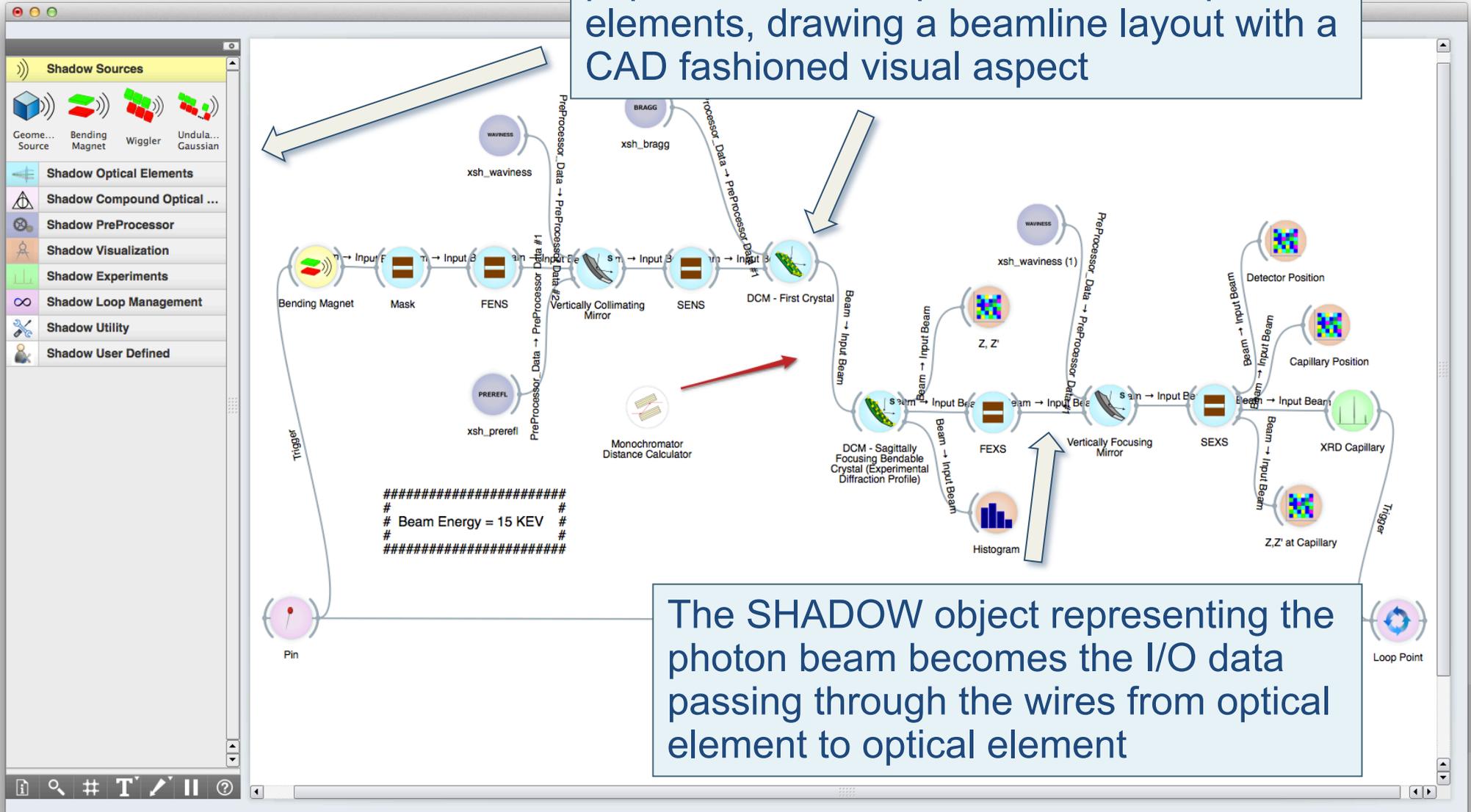
By calling SHADOW via its python API, we created SHADOW objects representing the different optical elements and photon sources as **ORANGE** widgets





GUI POWERED BY ORANGE

By interacting with the toolbox the user can populate the workspace area with optical elements, drawing a beamline layout with a CAD fashioned visual aspect



The SHADOW object representing the photon beam becomes the I/O data passing through the wires from optical element to optical element

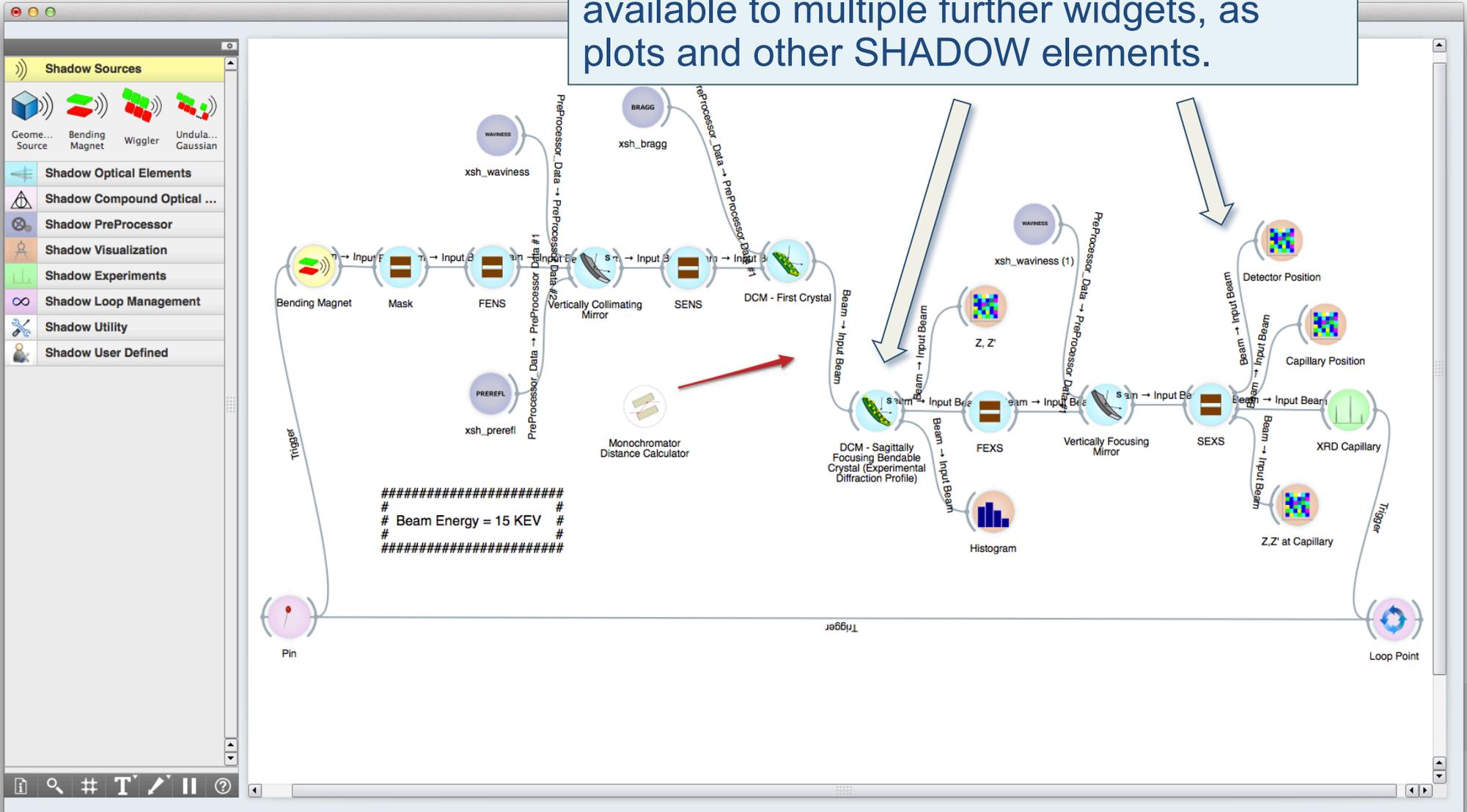


Elettra
Sincrotrone
Trieste



GUI powered by ORANGE

Thanks to ORANGE engine, the output of every widget (the SHADOW beam) is available to multiple further widgets, as plots and other SHADOW elements.



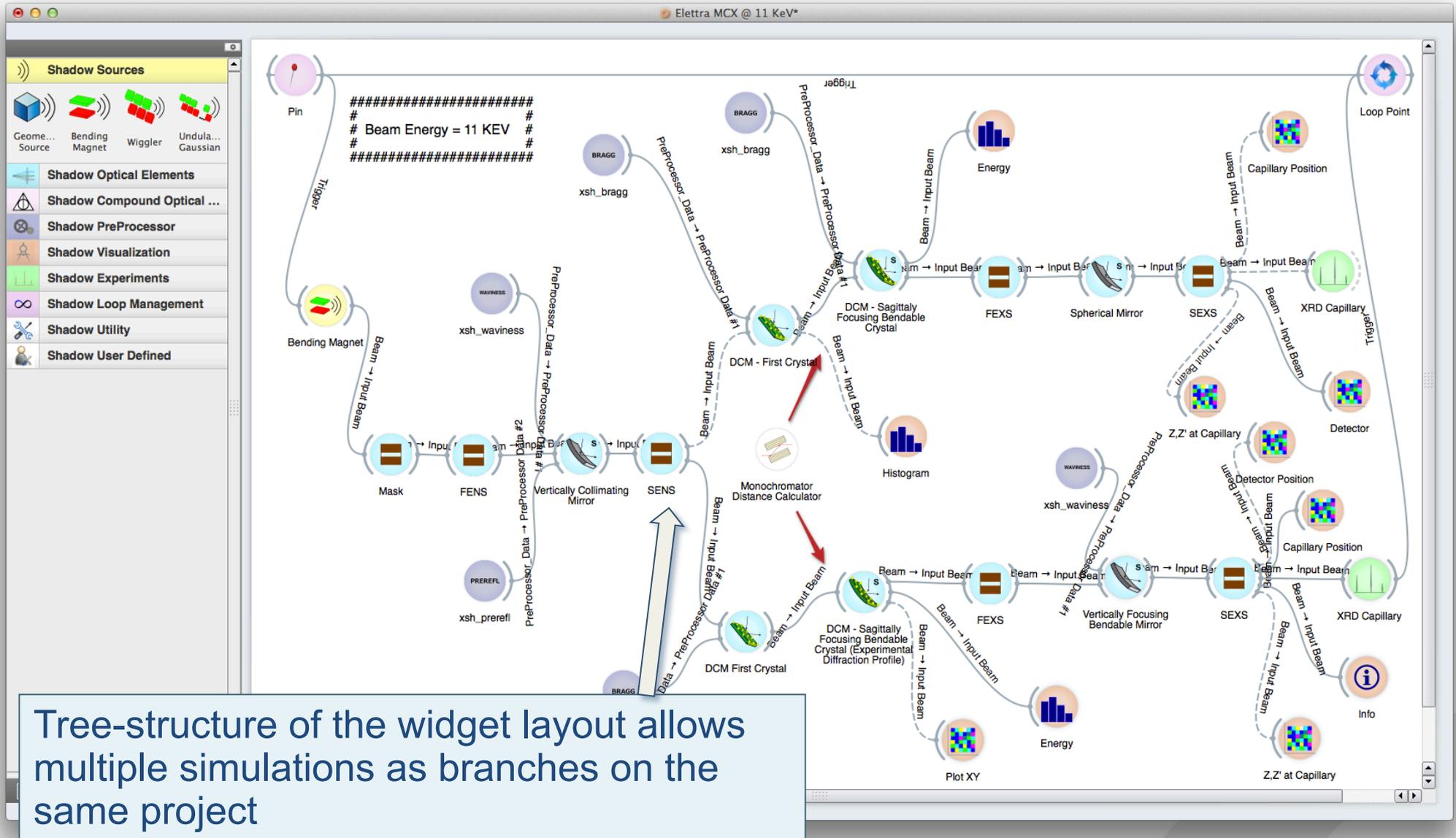


Elettra
Sincrotrone
Trieste



The European Synchrotron

GUI powered by ORANGE





Elettra
Sincrotrone
Trieste



GUI powered by ORANGE

The **ORANGE** engine has been customized to allow recursive simulations (with accumulation of the results), letting the user check results during time consuming simulations, and allowing simulations with an unlimited number of rays.

```

# Beam Energy = 15 KEV #
#####

```

The screenshot displays the Orange data science environment. On the left, a sidebar lists 'Shadow Sources' (Geome... Source, Bending Magnet, Wiggler, Undula... Gaussian), 'Shadow Optical Elements', 'Shadow Compound Optical ...', 'Shadow PreProcessor', 'Shadow Visualization', 'Shadow Experiments', 'Shadow Loop Management', 'Shadow Utility', and 'Shadow User Defined'. The main workspace shows a workflow diagram with nodes: Bending Magnet, Vertically Focusing Mirror, SEXS, XRD Capillary, Detector Position, Capillary Position, Z,Z' at Capillary, and Loop Point. Data flows are labeled with terms like 'Input Beam', 'Beam', and 'PreProcessor Data'. A central text box contains the text: 'The ORANGE engine has been customized to allow recursive simulations (with accumulation of the results), letting the user check results during time consuming simulations, and allowing simulations with an unlimited number of rays.' Below this text is a code snippet: '# Beam Energy = 15 KEV #' followed by a line of hash symbols. A 'Loop Point' node is connected to the start of the workflow, forming a loop. The window title is 'Elettra MCX @ 15 KeV*'.



Elettra
Sincrotrone
Trieste



GUI powered by ORANGE

Plots are now integrated in the I/O interface, for a rapid evaluation of the results.

The screenshot displays the 'Wiggler (1)' GUI interface. On the left, there are three main sections of controls:

- General Options:** Includes a checkbox for 'Display Shadow Output'.
- Monte Carlo and Energy Spectrum:** Contains input fields for 'Number of Rays' (100000), 'Seed' (0), 'Minimum Photon Energy (eV)' (14985.0), 'Maximum Photon Energy (eV)' (15015.0), 'Optimize Source? (reject rays)' (Using slit/acceptance), 'Max number of rejected rays (set 0 for infinity)' (10000000), 'Slit Distance [cm] (set 0 for angular acceptance)' (0.0), and several 'Min X [cm]/Min Xp [rad]', 'Max X [cm]/Max Xp [rad]', 'Min Z [cm]/Min Zp [rad]', and 'Max Z [cm]/Max Zp [rad]' fields.
- Machine Parameters:** Includes 'Electron Energy [GeV]' (2.0), 'Use Emittances?' (Yes), 'Sigma X [cm]' (0.0623000003), 'Sigma Z [cm]' (0.00700000022), 'Emittance X [rad.cm]' (7e-07), 'Emittance Z [rad.cm]' (7e-09), and 'Distance from Waist X [cm]' and 'Distance from Waist Z [cm]' (both 0.0).
- Wiggler Parameters:** Includes 'Type' (conventional/sinusoidal), 'Number of Periods' (29), 'K value' (21.0), and 'ID period [m]' (0.14).

At the bottom left are buttons for 'Run Shadow/Source' and 'Reset Fields'.

The main plotting area is titled 'Wiggler (1)' and 'Plotting Style' (Detailed Plot). It features a 'Select level of Plotting' dropdown and tabs for 'X,Z', 'X',Z'', 'X,X'', 'Z,Z'', and 'Energy'. The 'X,Z' tab is active, showing a 2D heatmap of intensity in the X-Z plane. To its right is a plot of 'Intensity' vs 'Frequency' with a red double-headed arrow indicating a range. Below the heatmap is a plot of 'Frequency' vs 'X [μm]' with a blue double-headed arrow indicating a range. An 'Info' panel on the right provides summary statistics:

Info	Value
Intensity	100000.000
Total Rays	100000
Total Good Rays	100000
Total Lost Rays	0
FWHM X [μm]	1390.3651
FWHM Z [μm]	152.7937

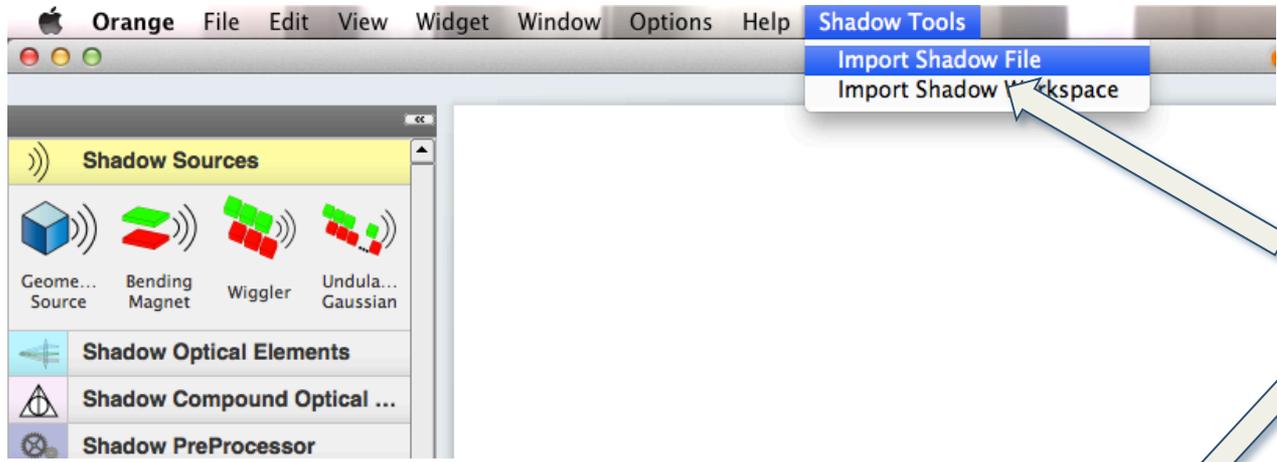
By double-clicking the widget its own I/O interface appears to interact with the user: fields and general layout has been migrated almost identical from the pristine SHADOW interface, to help users in the passage to the new tool



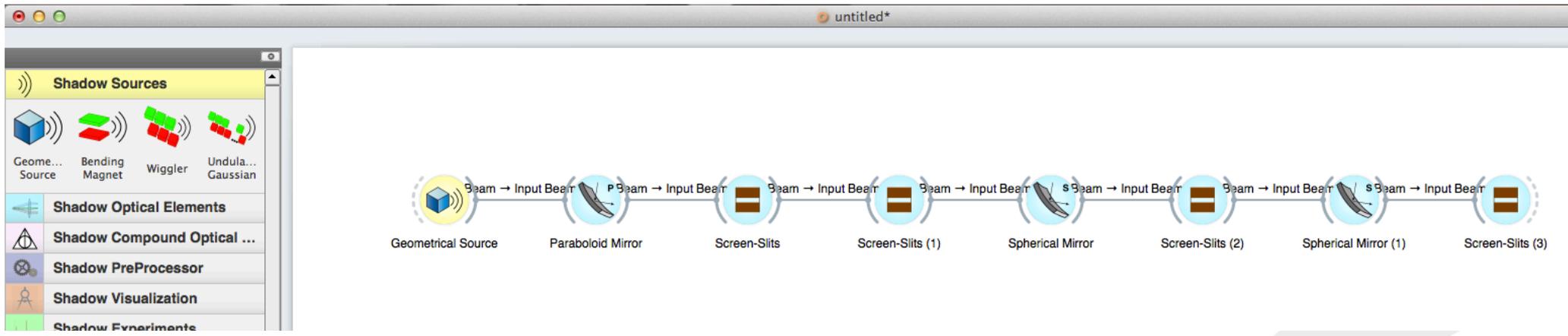
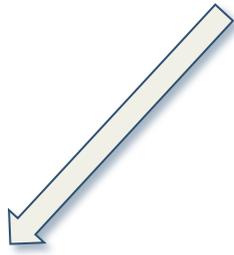
Elettra
Sincrotrone
Trieste



GUI powered by ORANGE



A dedicated import tool for old SHADOW projects has been implemented, to automatically create the widget structure and populate fields.



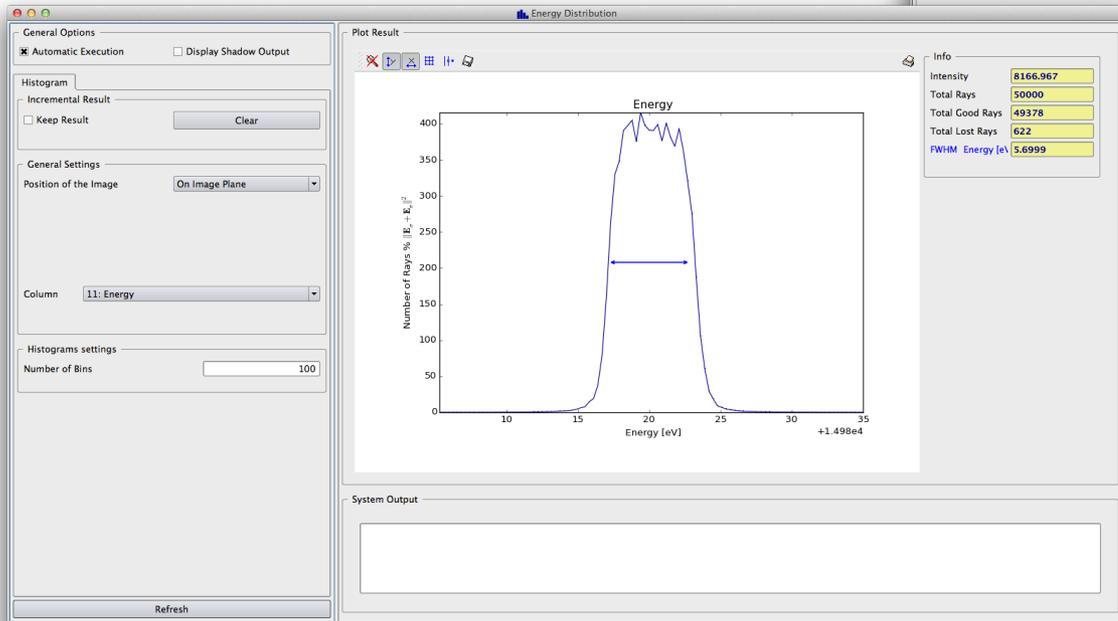
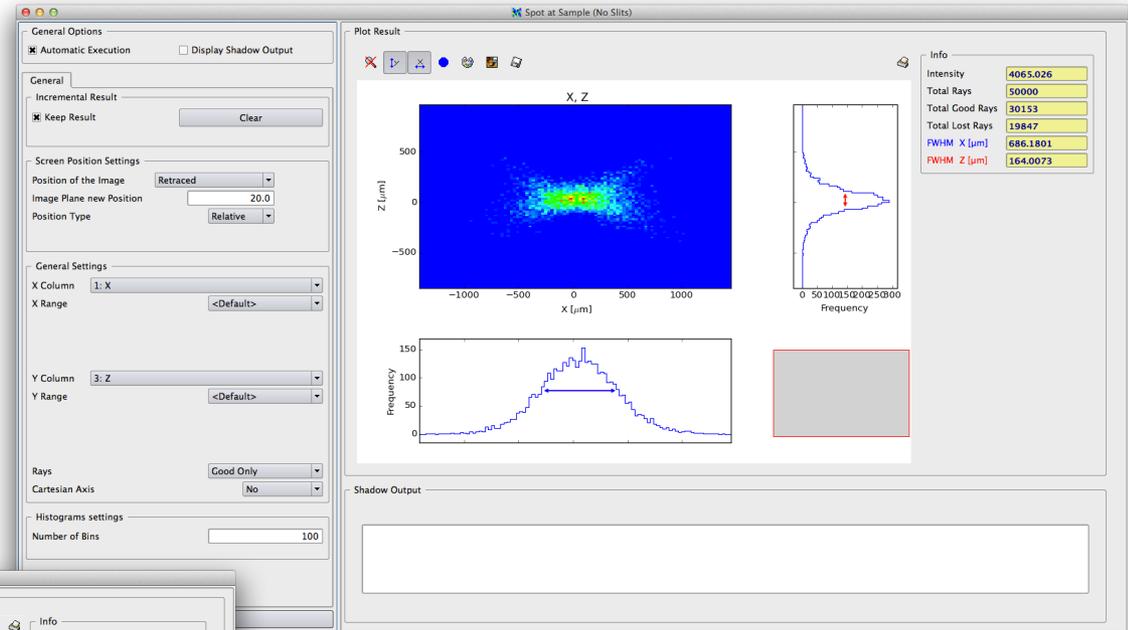


Elettra
Sincrotrone
Trieste



GUI powered by ORANGE

Using PyMCA (with Matplotlib backend) plotting library, several plotting functions have been introduced



Reproduce the same level of visual information given by the older SHADOW graphic user interface



Elettra
Sincrotrone
Trieste



2. OASYS



Elettra
Sincrotrone
Trieste



The OASYS Project



- ✓ **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

M. Sanchez del Rio, L. Rebuffi, J. Demšar, N. Canestrari, O. Chubar, "A proposal for an Open Source graphical environment for simulating X-ray optics", Proceedings of SPIE, Volume 9209 • New Advances in Computational Methods for X-Ray Optics III (2014).

L. Rebuffi & P. Scardi, "Calculation of the instrumental profile function for a powder diffraction beamline used in nanocrystalline material research", Proceedings of SPIE, Volume 9209 • New Advances in Computational Methods for X-Ray Optics III (2014).

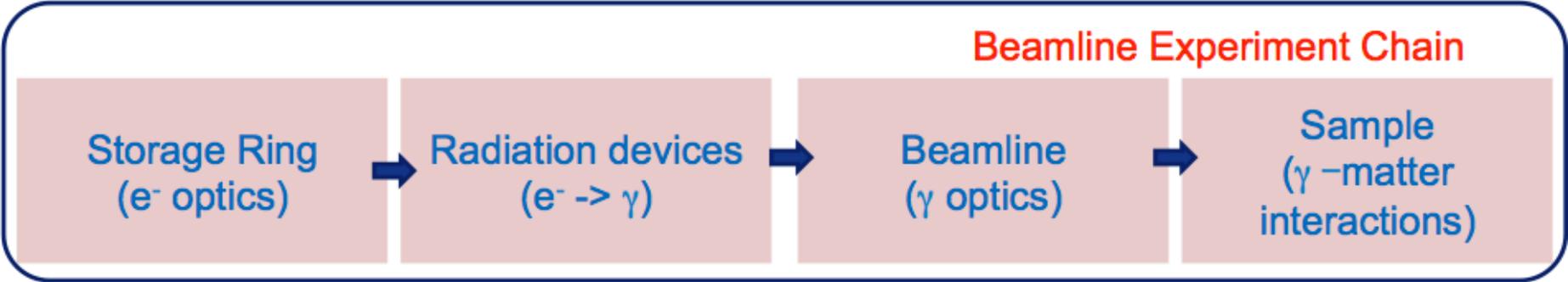


Elettra
Sincrotrone
Trieste

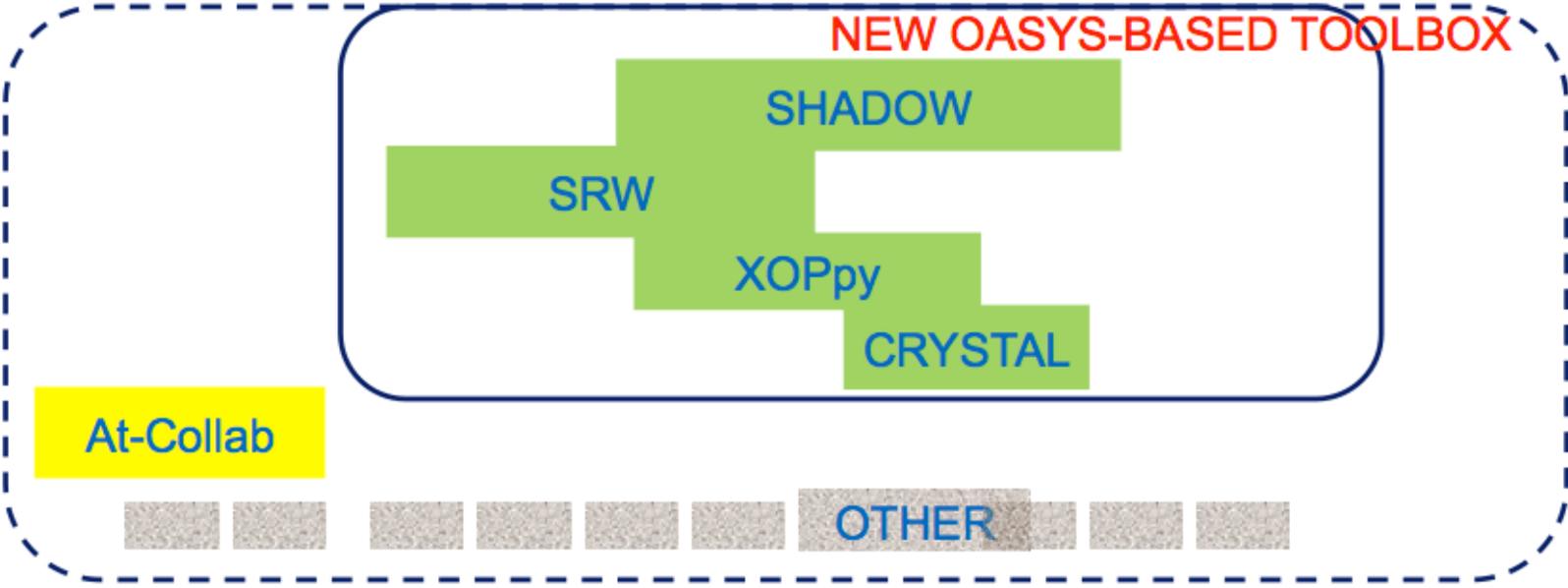


The OASYS Project: Virtual Experiments

Beamline Experiment Chain



NEW OASYS-BASED TOOLBOX



ON GOING

- A completely new User Interface for SHADOW is almost operational (L Rebuffi)
- A new SRW interface is planned; First tests done (M Glass)
- A new XOP under Oasys called XOPPY is under development (M Sanchez del Rio)
- An optics tools for crystals (including Stokes-based optics) is being coded (M Glass)



Elettra
Sincrotrone
Trieste



The OASYS Project: GUI - A dedicated Orange release



HOME SCREENSHOTS DOWNLOAD

Orange3

Orange3 is a ground-up reimplementaion of Orange.

1. We moved to **Python 3**; no backward compatibility.
2. We moved from custom data format to **numpy**.
3. We dropped our C++ code and use **Scikit-learn** (wherever possible) and **Cython** (where not).
4. Visual programming environment stays almost the same, just ported to Python 3.
5. No more PyQwt, OrangeQt... Viva **Pyqtgraph**!
6. Simpler widgets - above and under the hood. Like box plot below.
7. Widgets can read data from (Postgres) SQL and pass around queries. Viva **AXLE**!
8. We are rewriting the documentation, including the all-new **reference manual** for scripting.

Orange3 is in development and we plan to release its first version in late 2015. Check out our progress at [GitHub](#).

[Download latest bundle for OSX.](#)

[Download latest installer for Windows.](#)



University of Ljubljana
Faculty of Computer and
Information Science



In the last release of Orange 3 (Beta), CANVAS and WIDGETS are separated concepts, and Orange Canvas can be distributed without the Orange Widgets as an empty development environment





Elettra
Sincrotrone
Trieste



The OASYS Project: GUI - A dedicated Orange release



HOME SCREENSHOTS DOWNLOAD

Orange3

Orange3 is a ground-up reimplementation of Orange.

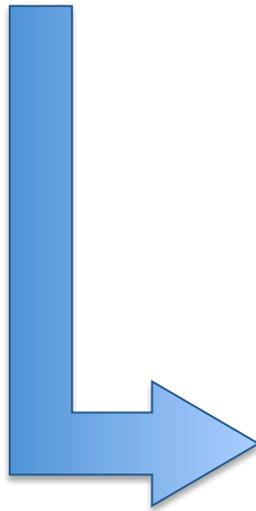
1. We moved to **Python 3**; no backward compatibility.
2. We moved from custom data format to **numpy**.
3. We dropped our C++ code and use **Scikit-learn** (wherever possible) and **Cython** (where not).
4. Visual programming environment stays almost the same, just ported to Python 3.
5. No more PyQwt, OrangeQt... Viva **Pyqtgraph**!
6. Simpler widgets - above and under the hood. Like box plot below.
7. Widgets can read data from (Postgres) SQL and pass around queries. Viva **AXLE**!
8. We are rewriting the documentation, including the all-new **reference manual** for scripting.

Orange3 is in development and we plan to release its first version in late 2015. Check out our progress at [GitHub](#).

[Download latest bundle for OSX.](#)

[Download latest installer for Windows.](#)

A customized version of Orange Canvas has been developed and released by University of Ljubljana and distributed through PyPI site, “absorbing” all the special features we developed in the prototyping phase of OASYS.



python™

» Package Index > OASYS > 0.1.3

PACKAGE INDEX »

- Browse packages
- Package submission
- List trove classifiers
- List packages
- RSS (latest 40 updates)
- RSS (newest 40 packages)
- Python 3 Packages
- PyPI Tutorial
- PyPI Security
- PyPI Support
- PyPI Bug Reports
- PyPI Discussion
- PyPI Developer Info

ABOUT »

NEWS »

DOCUMENTATION »

DOWNLOAD »

COMMUNITY »

FOUNDATION »

CORE DEVELOPMENT »

OASYS 0.1.3

OrAnge SYnchrotron Suite

Downloads ↓

OASYS (OrAnge SYnchrotron Suite) is a graphical environment for optic simulation software used in synchrotron facilities, based on Orange 3.

OASYS package requires Python 3.3 or newer.

Installing

OASYS is pip installable (<<https://pip.pypa.io/>>), simply run:

```
pip install oasys
```

to install it.

OASYS requires PyQt, which is not pip-installable in Python 3. You have to download and install it system-wide. If you are installing in a virtual environment make sure it is created with the `--system-site-packages` so it will have access to the installed PyQt4.

Starting OASYS

To start OASYS from the command line, run:

```
python3 -m oasys.canvas
```

The OASYS does not itself come with any widget components. Use the 'Options->Add-ons' menu entry to install the desired widget set.

Not Logged In

- Login
- Register
- Lost Login?
- Use OpenID

Status

Nothing to report



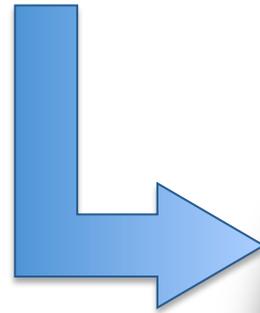
Elettra
Sincrotrone
Trieste



The OASYS Project: GUI - A dedicated Orange release



OASYS package is fully dependent from Orange Canvas native package:
IT IS NOT A SEPARATE BRANCH!



OASYS works as a mask: it “wraps” ORANGE modifying its default behaviour





Elettra
Sincrotrone
Trieste



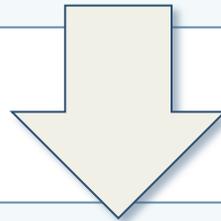
The OASYS Project: A common library for virtual experiments

OASYS is designed to be FULLY dependent from

THE OPTICS PACKAGE

a common input for every optics calculation backend:

Collection of entities (objects) containing the minimum amount of parameters needed to describe the beamline source and components



A STRATEGY TO COMBINE SEVERAL SIMULATION

A COMMON INTERFACE TO AVOID DATA REDUNDANCY



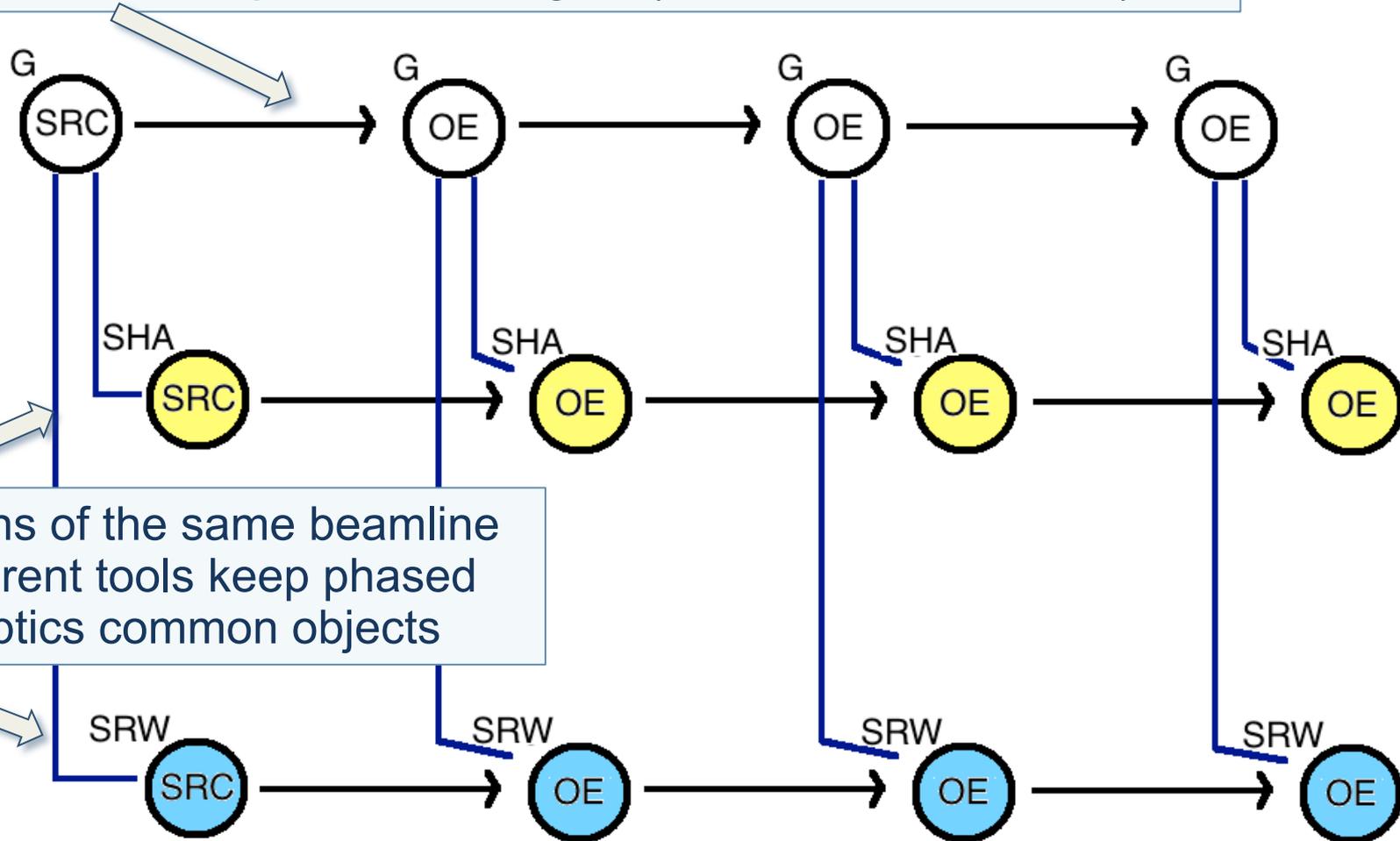
Elettra
Sincrotrone
Trieste



The OASYS Project

A common library for virtual experiments

Optics Common Widgets will represent the beamline at the highest abstraction level and will send Optics objects to populate common information in the specialized widgets (SHADOW, SRW, etc..)



Simulations of the same beamline with different tools keep phased with optics common objects

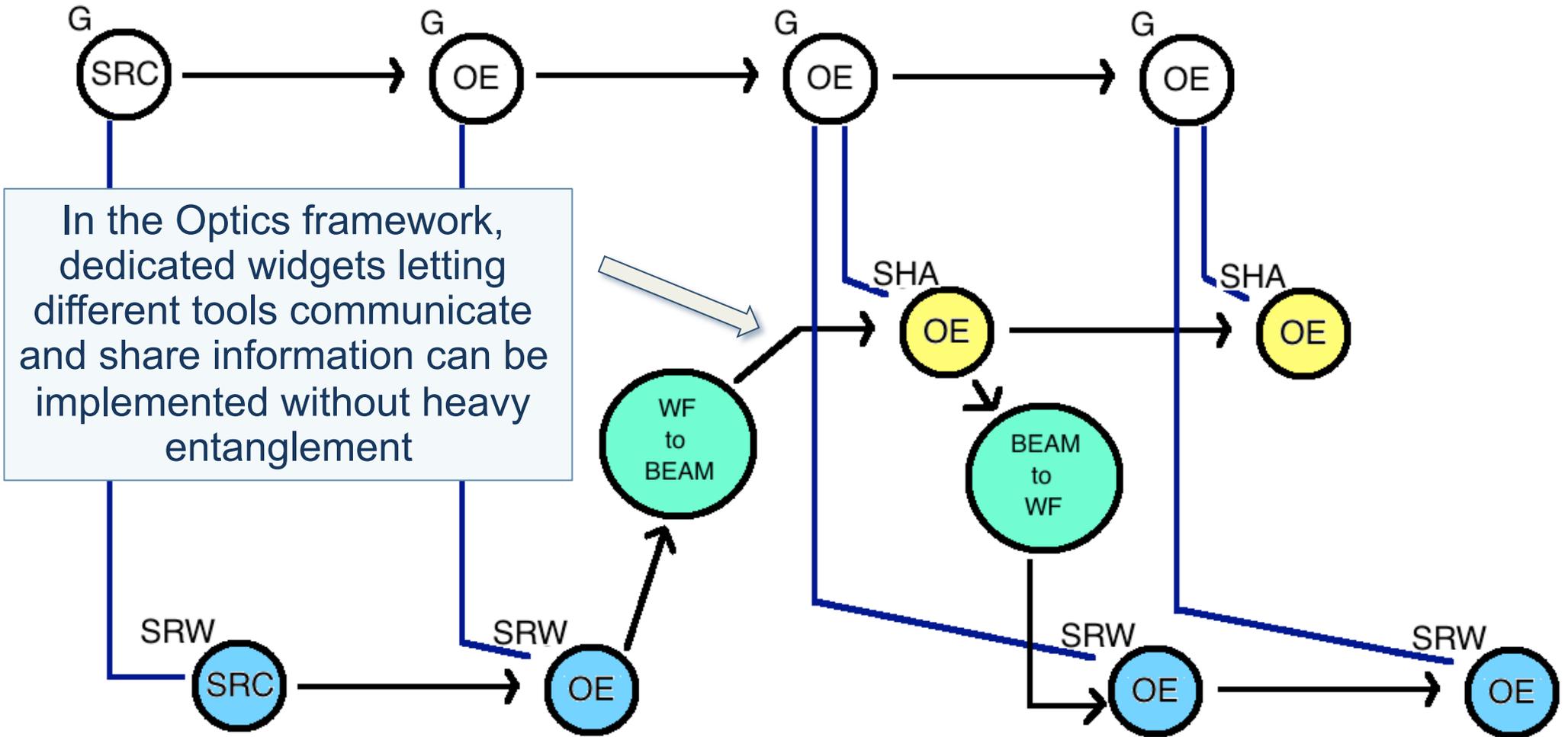


Elettra
Sincrotrone
Trieste



The OASYS Project

A common library for virtual experiments





Elettra
Sincrotrone
Trieste

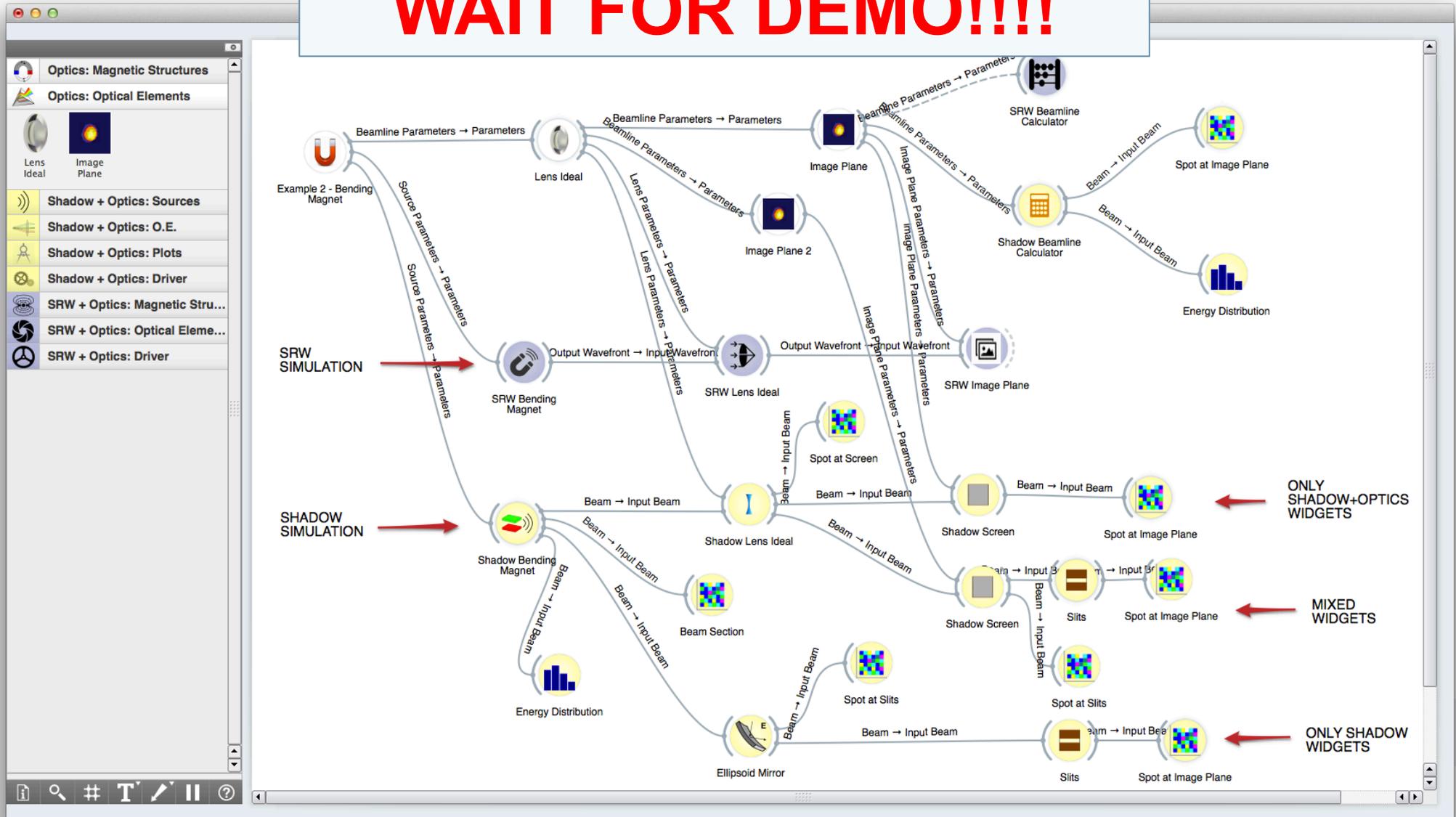


The European Synchrotron

The OASYS Project

A common library for virtual experiments

WAIT FOR DEMO!!!!





Elettra
Sincrotrone
Trieste



Thank you!



Elettra
Sincrotrone
Trieste



www.elettra.eu