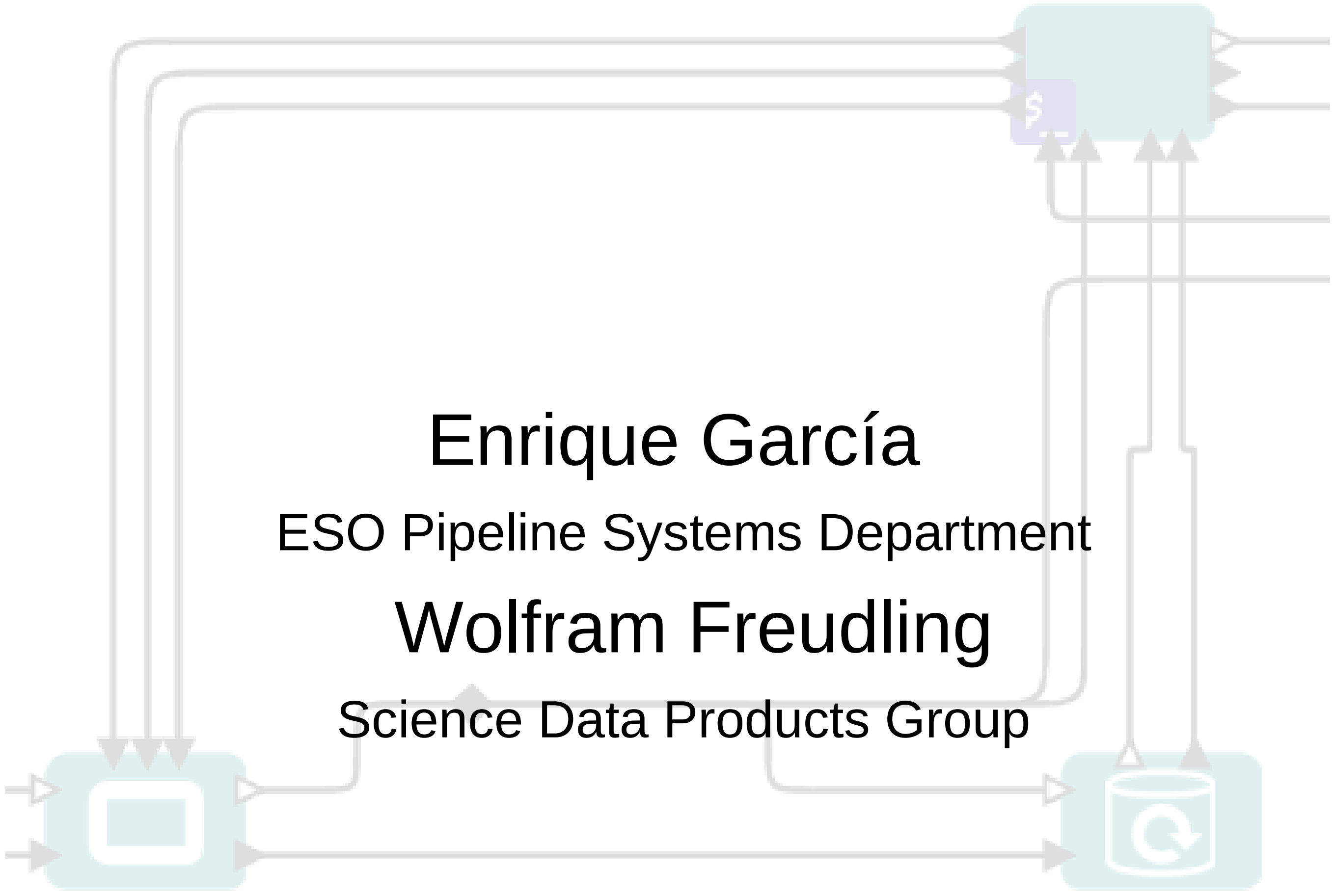
Advance Use of REFLEX Workflows



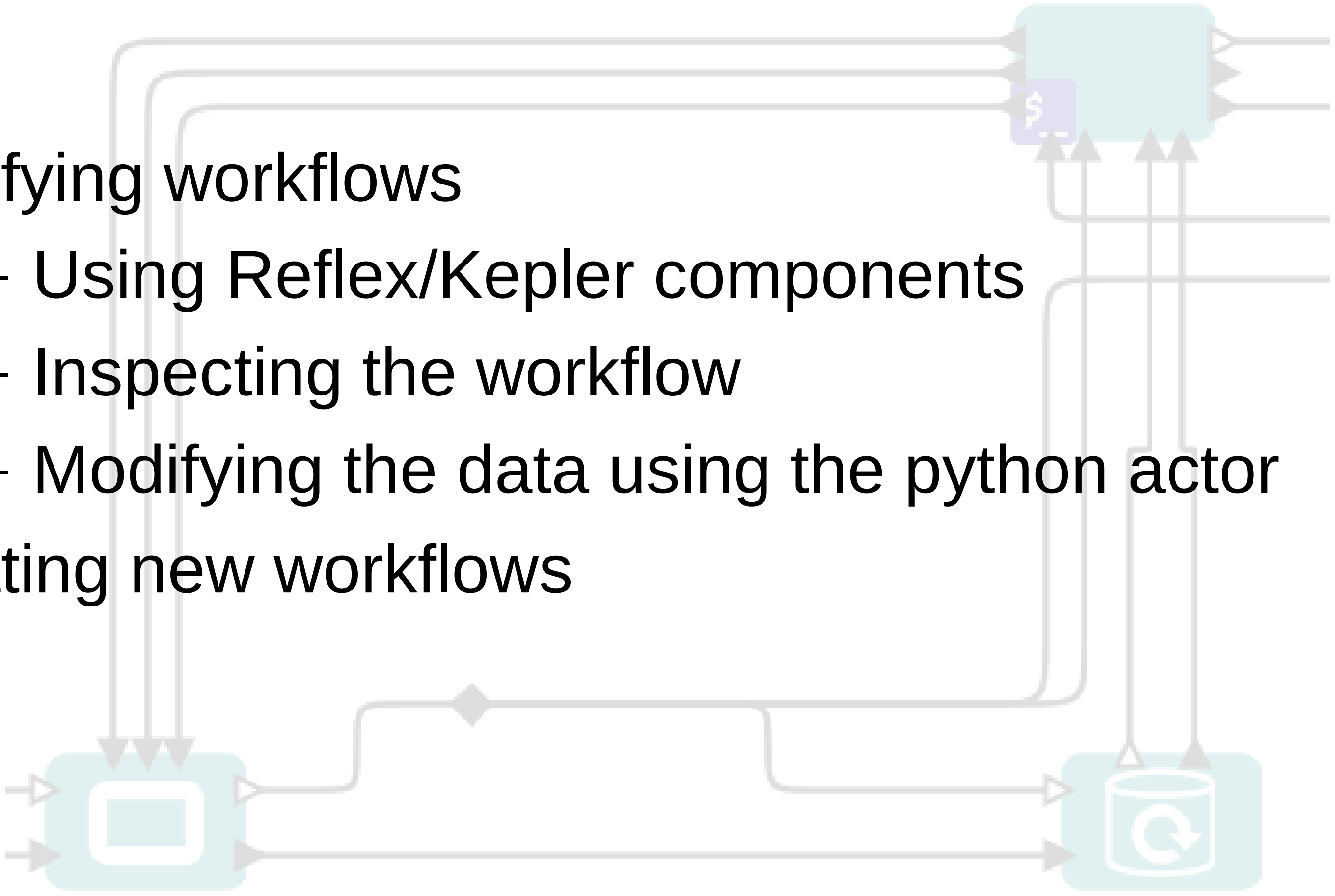
Enrique García

ESO Pipeline Systems Department

Wolfram Freudling

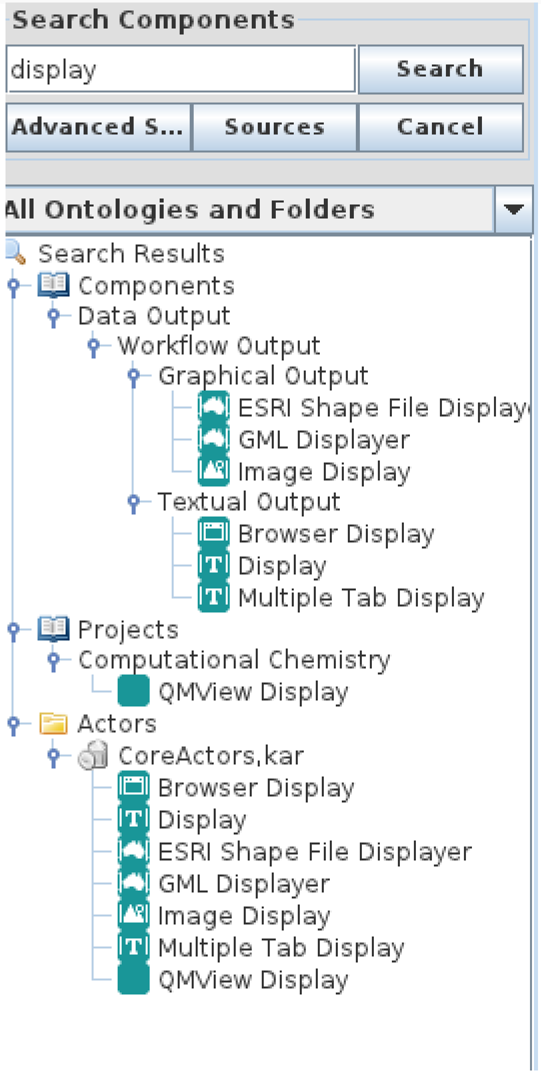
Science Data Products Group

Outline



- Modifying workflows
  - Using Reflex/Kepler components
  - Inspecting the workflow
  - Modifying the data using the python actor
- Creating new workflows

Using the components library


Search Components

display

Search

Advanced S... Sources Cancel

All Ontologies and Folders

- Search Results
- Components
  - Data Output
    - Workflow Output
      - Graphical Output
        - ESRI Shape File Display
        - GML Displayer
        - Image Display
      - Textual Output
        - Browser Display
        - Display
        - Multiple Tab Display
- Projects
  - Computational Chemistry
    - QMView Display
- Actors
  - CoreActors.kar
    - Browser Display
    - Display
    - ESRI Shape File Displayer
    - GML Displayer
    - Image Display
    - Multiple Tab Display
    - QMView Display

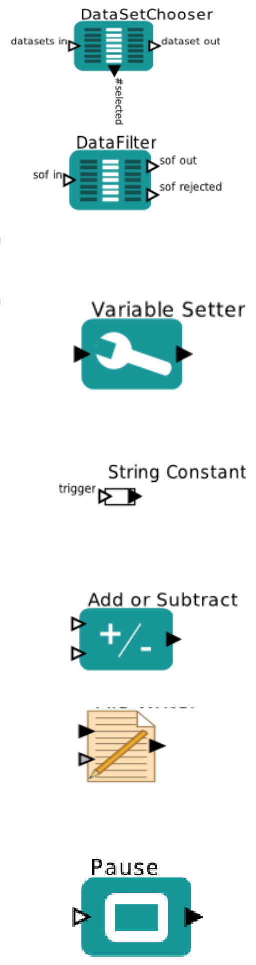
The search components utility allow to filter the available components by name.

The Reflex folder contains all the Reflex specific actors developed for astronomical reduction.

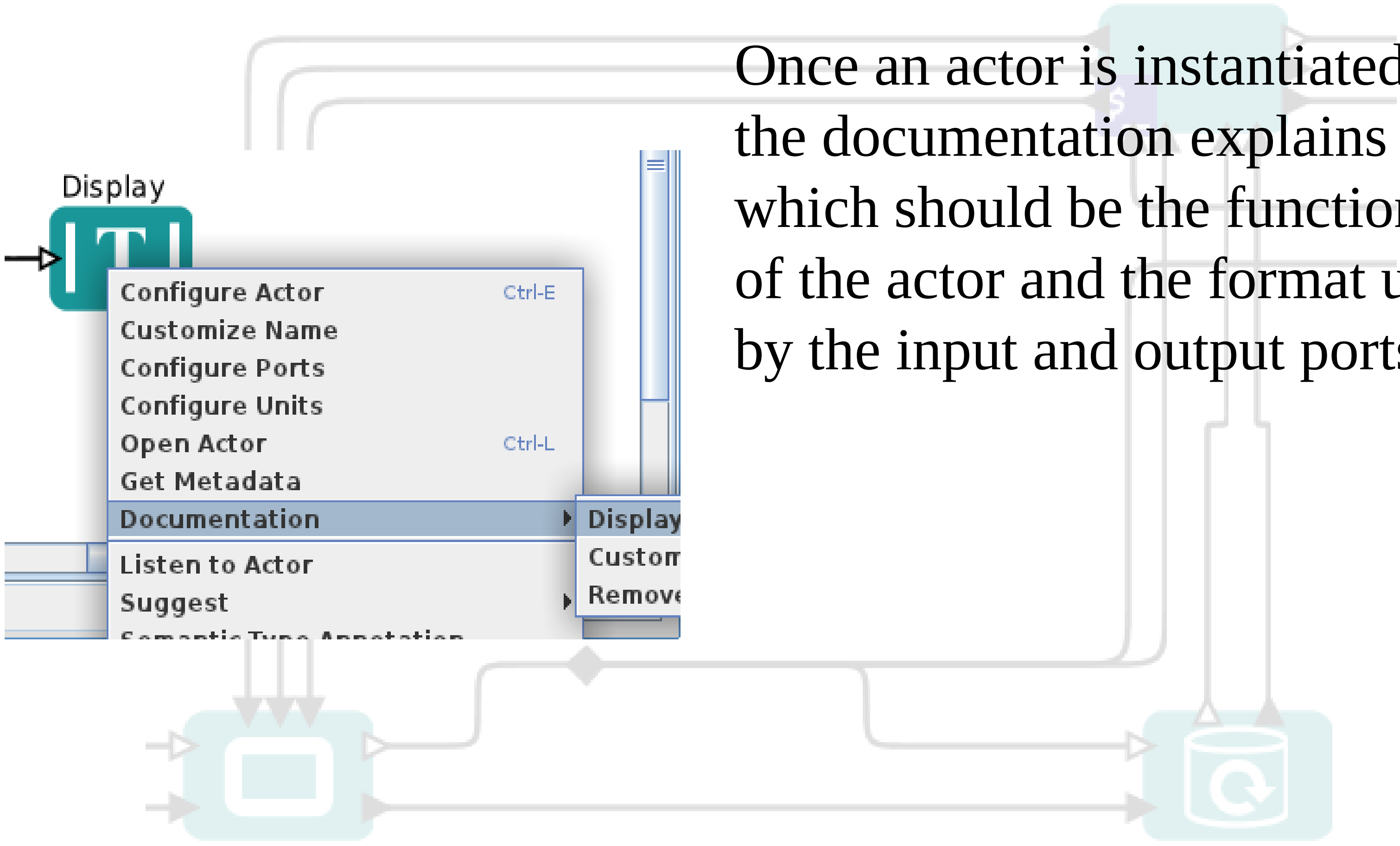
To use them, simply **drag and drop.**

Useful components

- Reflex components:
  - Data Organizer, DataSetChooser, Fits Router
  - DataFilter, RecipeLooper
  - PythonActor, IDLActor
- Kepler components:
  - Variable Setter
  - String Constant
  - Parameters (String, File, ...)
  - Arithmetic operations
  - File Writer
  - Directors
  - Composite Actor
  - Pause, Stop
  - Logical operations

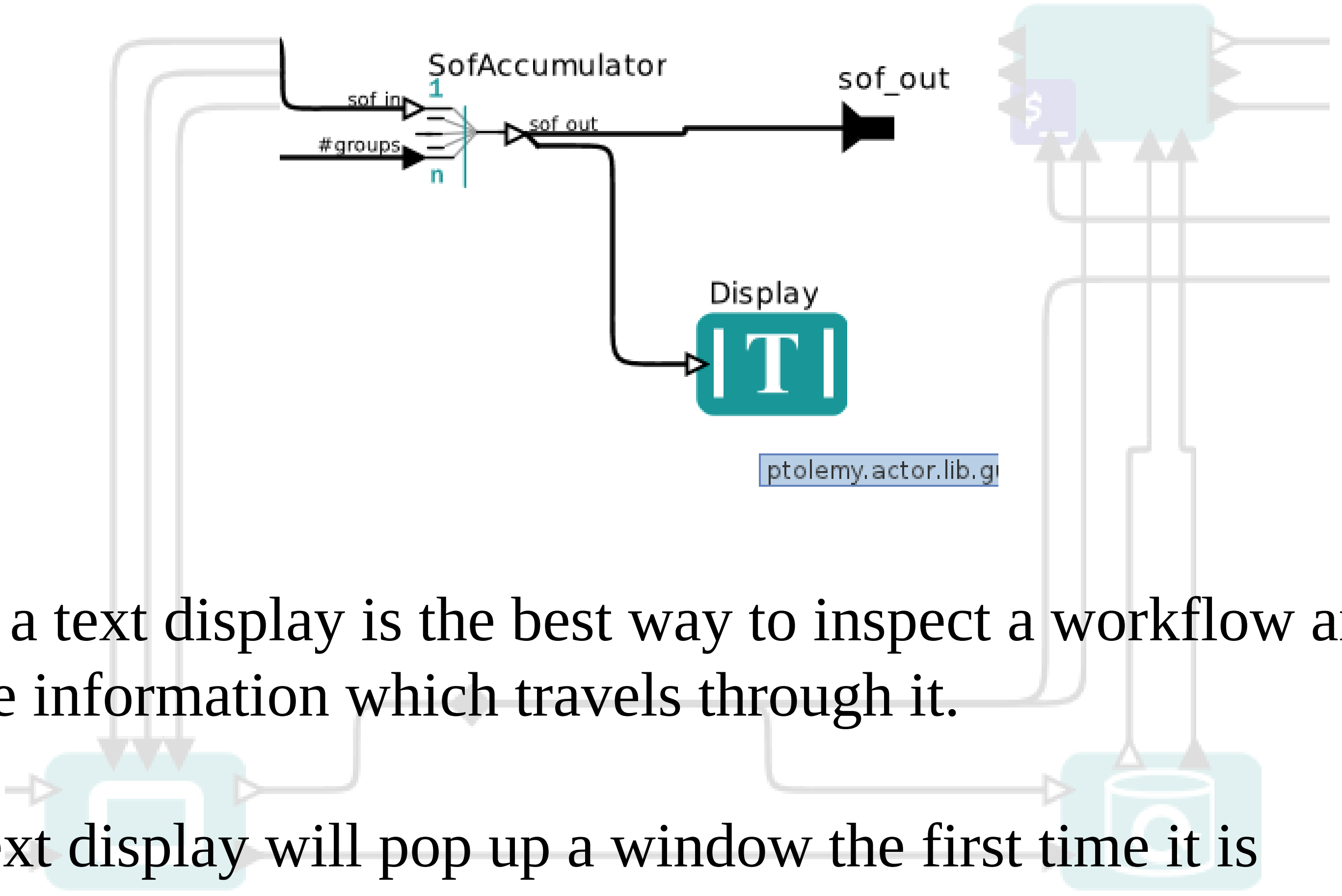


Browsing actor documentation



Once an actor is instantiated, the documentation explains which should be the functionality of the actor and the format used by the input and output ports.

Debuging: Using the Text Display

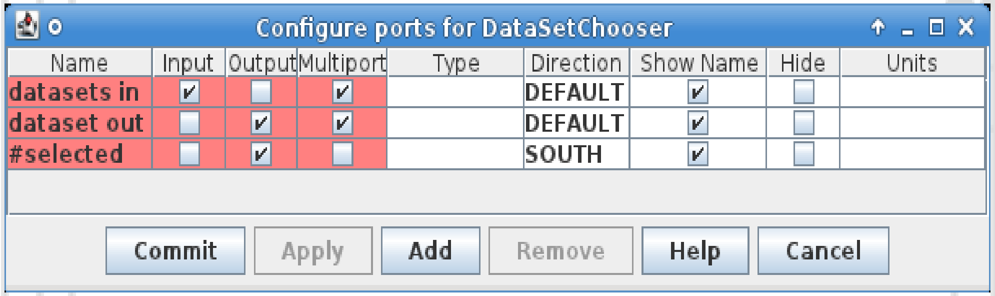


Using a text display is the best way to inspect a workflow and see the information which travels through it.

The text display will pop up a window the first time it is triggered.

Changing the ports

- Each actor has input and output ports.
  - To edit them, right click on “Configure ports”



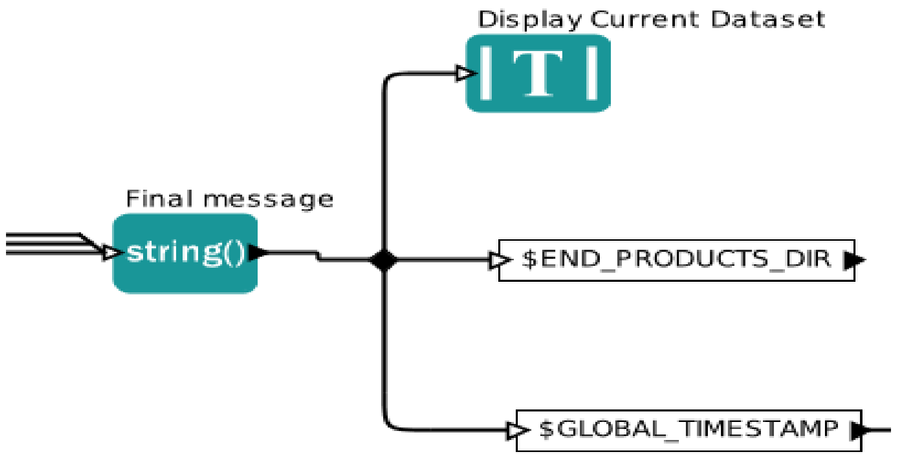
Configure ports for DataSetChooser

Name	Input	Output	Multiport	Type	Direction	Show Name	Hide	Units
datasets in	✓	☐	✓		DEFAULT	✓	☐	
dataset out	☐	✓	✓		DEFAULT	✓	☐	
#selected	☐	✓	☐		SOUTH	✓	☐	

- An input multiport can receive several tokens which are combined together
- An output multiport distributes the token to several actors
- Many output ports are single ports and cannot be changed.

The diamond

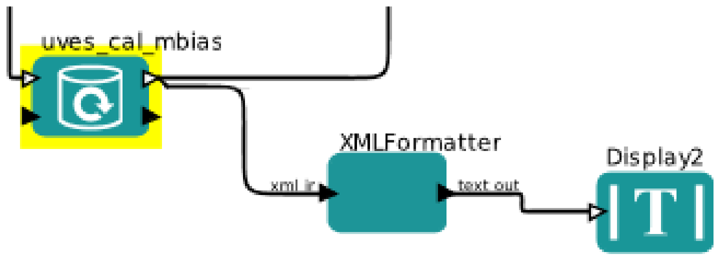
- To connect the output of a single port to several actors, the diamond will copy the output to several connections



- TIP: A connection cannot be initiated from a diamond, but rather from the other end of the connection.

Displaying human-readable output

- The Display actor will show the original XML format of the data.
- In order to show human-readable output, the XMLFormatter can decode all the XML-data which is transmitted along the workflow.

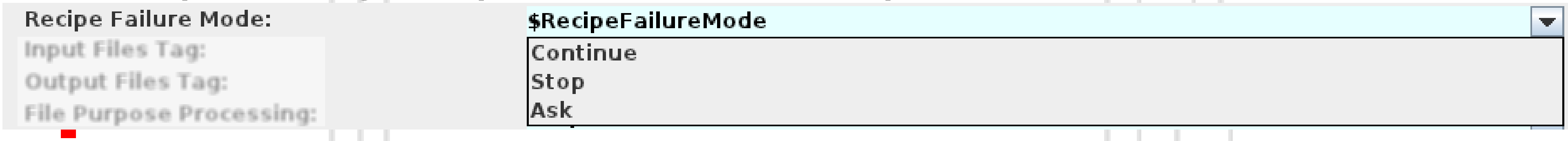


Debuging: Looking into the directories

Bookeping	Input SOF, output SOF. Recipe parameters DataOrganization
Logs	Recipe output logs
Tmp_Products	Recipe execution products (FITS files)
End_Products	Final scientific products with meaningful names

Re-executing a recipe

- Sometimes a recipe fails due a number of factors: bad parameters, wrong data, software bugs, etc...
- The workflow reacts to the failure of the recipe depending on parameter “Recipe Failure Mode”



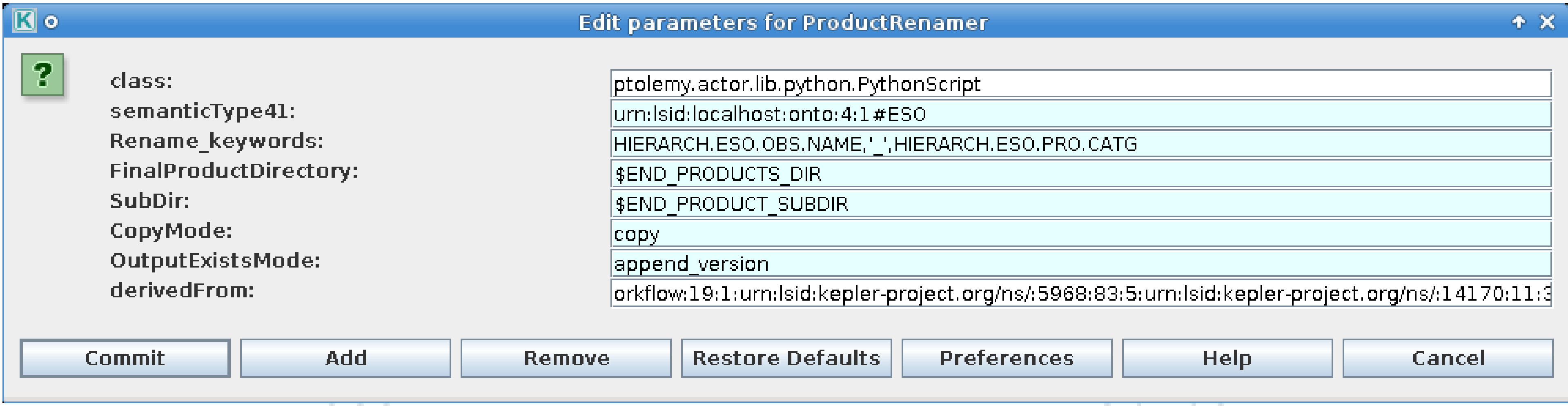
Recipe Failure Mode: $RecipeFailureMode
Input Files Tag:
Output Files Tag:
File Purpose Processing:

Continue
Stop
Ask

- For command-line fans, it is possible to go into the “Bookkeeping dir” and re-execute the recipe with esorex:

```
#esorex recipe_name data.sof
```

Product Renamer


Edit parameters for ProductRenamer

class:	ptolemy.actor.lib.python.PythonScript
semanticType41:	urn:lsid:localhost:onto:4:1#ESO
Rename_keywords:	HIERARCH.ESO.OBS.NAME,'_',HIERARCH.ESO.PRO.CATG
FinalProductDirectory:	$END_PRODUCTS_DIR
SubDir:	$END_PRODUCT_SUBDIR
CopyMode:	copy
OutputExistsMode:	append_version
derivedFrom:	orkflow:19:1:urn:lsid:kepler-project.org/ns/:5968:83:5:urn:lsid:kepler-project.org/ns/:14170:11:3

- Rename_keywords specifies the pattern used to create the file name, based on the available FITS keywords.
- The directory to be used will be FinalProductDirectory/SubDir. Subdir is usually defined by the workflow to be the execution timestamp.
- The actor is actually a jython script which can be modified by double-clicking it.

The python actor

- It is able to execute generic python code.
- To translate from/to Reflex ports to/from python script arguments a special syntax is used, with the help of a Python module
- To create a python actor, use the menu *Tools->Instantiate Component* and type *org.eso.PythonActor.*
- Similar to the python actor there is an IDL actor available.

The python actor (II)

- A python script can be “reflexed” using the following syntax:

```
from reflex import *
  parser = ReflexIOParser()
  #Define inputs
  parser.add_option("-i", "--in_sof", dest="in_sof")
  #Define outputs
  parser.add_output("-o", "--out_sof", dest="out_sof")

  (inputs, args) = parser.parse_args()
  outputs = parser.get_outputs()
  #Set the output
  outputs.out_sof = inputs.in_sof
  parser.print_outputs()
  sys.exit()
```

Importing Reflex

Define Inputs/Outputs

Getting inputs

Setting outputs

- TIP: Retrieve the script from **http://goo.gl/6P9sv**

The python actor (III)

- The real data processing looks like:

```
#Retrieve input
in_sof = inputs.in_sof

#Get the input files
files, dataset = parseSof(in_sof)

#Do the stuff
for file in files:
  hdulist = pyfits.open(file.name,mode='update')
  newdata = hdulist[0].data / 2
  hdulist[0].data = newdata
  hdulist.flush()
```

Parsing the Reflex format

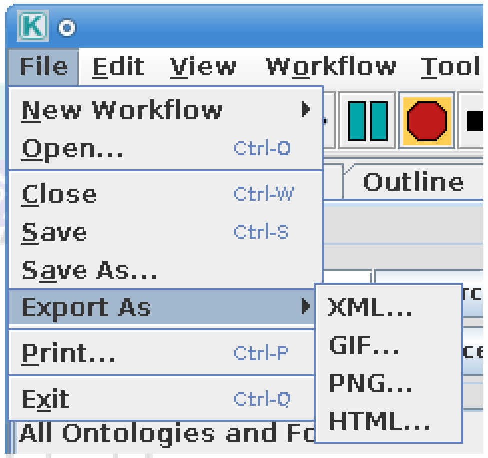
Using pyfits

Updating the file

- TIP: Retrieve the script from **http://goo.gl/6P9sv**

Saving and sharing a workflow

- There are two formats to save a workflow:
  - KAR. This is the default format. Binary format
  - XML. This has to be exported. “Human-readable”
- The paths to the python scripts and OCA rules file are saved together with the workflow.
- If one wants to share a workflow with a college:
  - Make sure that the pipeline is installed in the target system
  - Export as XML
  - Edit the XML and change the python scripts/OCA file paths
  - Open the workflow and change the data paths


File
Edit
View
Workflow
Tool
New Workflow
Open... Ctrl-O
Close Ctrl-W
Save Ctrl-S
Save As...
Export As
Print... Ctrl-P
Exit Ctrl-Q
All Ontologies and Fo
Outline
XML...
GIF...
PNG...
HTML...