



### Workflow Instructions

To run this workflow on the demo data:  
 - Turn on highlighting. Choose "Tools"-> "Animate at Runtime" from top menu and set it to "1".  
 - Press the "Run" button OR cntrl-R to start the workflow.

To run on a different data set:  
 - Click on ROOT\_DATA\_DIR and set as appropriate.  
 - All subdirectories of ROOT\_DATA\_DIR will be used.  
 - If desired, change END\_PRODUCT\_SUBDIR.  
 - Press the "Run" button OR cntrl-R to start the workflow.

To monitor the progress of the workflow in more detail:  
 - Open "Window" -> "Runtime Window" in top menu before starting the workflow.

### Setup Directories

Input:  
 \*ROOT\_DATA\_DIR: /home/freudl/eso/data/  
 \*RAWDATA\_DIR: \$ROOT\_DATA\_DIR/reflex\_input/ives\_bar  
 Working Directories:  
 \*BOOKKEEPING\_DIR: \$ROOT\_DATA\_DIR/reflex\_book\_keeping/ives\_bar  
 \*LOGS\_DIR: \$ROOT\_DATA\_DIR/reflex\_logs/  
 \*END\_PRODUCTS\_DIR: \$ROOT\_DATA\_DIR/reflex\_ives\_products

### Global Parameters

\*FITS\_VIEWER: fv = actor with interactive option  
 \*ESOREX\_OPTS: --suppress-profiles=TRUE = actor with interactive option esorex arguments  
 \*END\_PRODUCT\_SUBDIR: 2010-07-07T18:10:29/Science\_DataSet\_1 This is set automatically  
 \*GLOBAL\_TIMESTAMP: 2010-07-07T18:10:29 This is set automatically

# The ESO Recipe Flexible Workbench EsoReflex

Step 1:  
Data Organisation  
and Selection

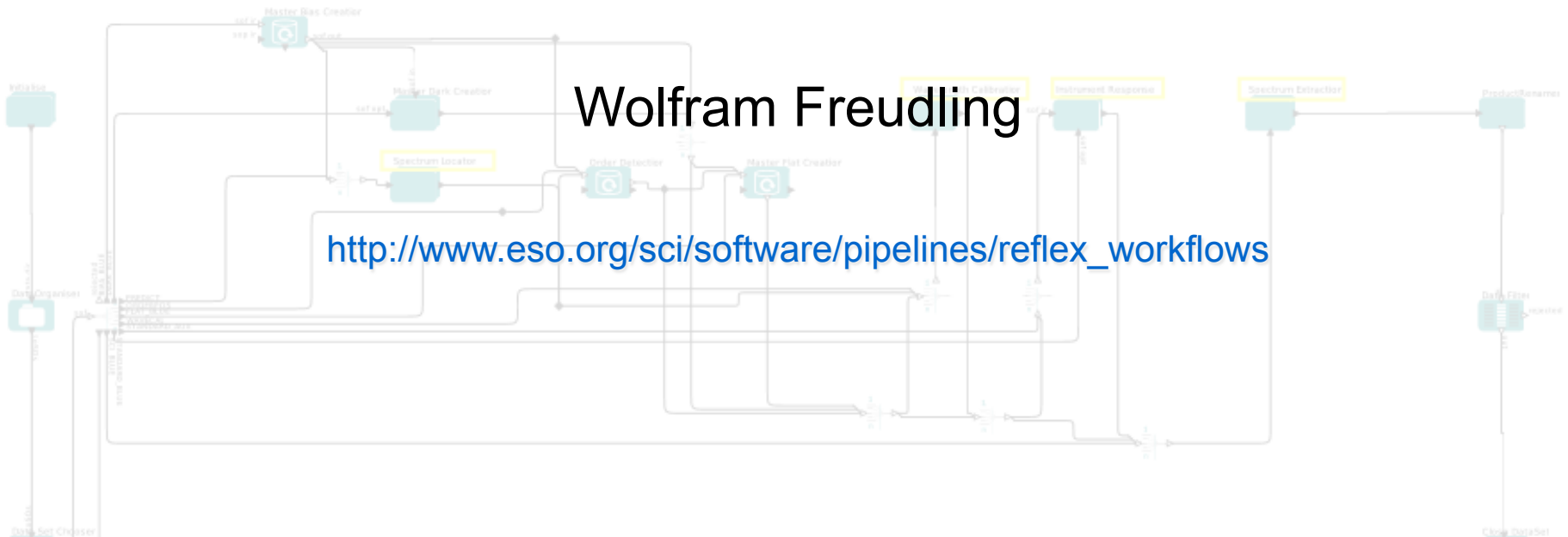
## II. Advanced Features

Step 4:  
Spectrum  
Extraction

Step 5:  
Output  
Organisation

Wolfram Freudling

[http://www.eso.org/sci/software/pipelines/reflex\\_workflows](http://www.eso.org/sci/software/pipelines/reflex_workflows)





# Workflow driven Data Reduction

A&A 559, A96 (2013)

A&A 559, A96 (2013)  
DOI: [10.1051/0004-6361/201322494](https://doi.org/10.1051/0004-6361/201322494)  
© ESO 2013

Astronomy  
&  
Astrophysics

## Automated data reduction workflows for astronomy

### The ESO Reflex environment

W. Freudling, M. Romaniello, D. M. Bramich, P. Ballester, V. Forchi, C. E. García-Dabó, S. Moehler, and M. J. Neeser

European Southern Observatory, Karl-Schwarzschild-Str. 2, 85748 Garching, Germany  
e-mail: [wfreudli@eso.org](mailto:wfreudli@eso.org)

Received 16 August 2013 / Accepted 14 October 2013

#### ABSTRACT

**Context.** Data from complex modern astronomical instruments often consist of a large number of different science and calibration files, and their reduction requires a variety of software tools. The execution chain of the tools represents a complex workflow that needs to be tuned and supervised, often by individual researchers that are not necessarily experts for any specific instrument.

**Aims.** The efficiency of data reduction can be improved by using automatic workflows to organise data and execute a sequence of data reduction steps. To realize such efficiency gains, we designed a system that allows intuitive representation, execution and modification of the data reduction workflow, and has facilities for inspection and interaction with the data.

**Methods.** The European Southern Observatory (ESO) has developed Reflex, an environment to automate data reduction workflows. Reflex is implemented as a package of customized components for the Kepler workflow engine. Kepler provides the graphical user interface to create an executable flowchart-like representation of the data reduction process. Key features of Reflex are a rule-based data organiser, infrastructure to re-use results, thorough book-keeping, data progeny tracking, interactive user interfaces, and a novel concept to exploit information created during data organisation for the workflow execution.

**Results.** Automated workflows can greatly increase the efficiency of astronomical data reduction. In Reflex, workflows can be run non-interactively as a first step. Subsequent optimization can then be carried out while transparently re-using all unchanged intermediate products. We found that such workflows enable the reduction of complex data by non-expert users and minimizes mistakes due to book-keeping errors.

**Conclusions.** Reflex includes novel concepts to increase the efficiency of astronomical data processing. While Reflex is a specific

Workflow It

To run this  
- Turn on  
from top  
- Press th

To run on a  
- Click on a  
All subdi  
- If desire  
- Press th

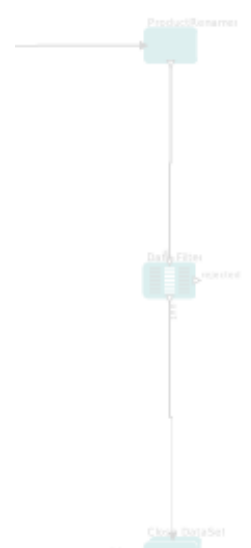
To monitor  
- Open "W  
starting

Ste  
Data Org  
and S

= actor with interactive option

ents:  
- DR and COGS\_DR  
Made won't work anymore.  
rset\_1 This is set automatically  
This is set automatically

Step 5:  
Output  
Organisation





# Command line interface examples

List all available workflows:

```
esoreflex -l
```

Load FORS workflow:

```
esoreflex fors_imaging
```

Process all new data in my directory without interaction:

```
esoreflex -n -RAW_DATA_DIR /data/fors_data fors_imaging
```

Rerun all previously failed datasets with nonstandard workflow parameter:

```
esoreflex -n -RAW_DATA_DIR /data/muse_data \  
-SelectDatasetMode failed -recomputeWCS true muse
```



file:/Users/wfreudli/KeplerData/workflows/MyWorkflows/fors\_imaging\_multitab.kar

Components | Data | Outline | Workflow

Search Components

Advanced ... Sources Cancel

All Ontologies and Folders

- Components
- Projects
- Statistics
- Demos
- Actors
- Dataturbine
- Directors
- Esoreflex
- Job
- MyWorkflows
- Opendap

0 results found.

**imaging Data (v. 5.1.4)**

**Global Parameters** = actor with interactive option

- RecipeFailureMode: Ask  
Global parameter for the behaviour when a recipe fails. 'Ask' means that each time a recipe fails, the choice to continue or stop will be presented. 'Continue' means that the workflow will ignore errors and continue. 'Stop' means the workflow will stop.  
Change "EraseDirs" to "true" to erase BOOKKEEPING\_DIR, TMP\_PRODUCTS\_DIR and LOGS\_DIR each time the workflow is run (Lazy Mode will not work anymore)
- EraseDirs: false
- FITS\_VIEWER: fv  
Program to use for the inspection of input/output products. Use full path name if it is not in the standard path.
- GlobalPlotInteractivity: true  
Set to "false" to disable interactive GUIs for the whole workflow. Each interactive actor can specify its own setting, which overwrites the choice given here.
- SelectDatasetMethod: Interactive  
Specify how datasets for processing are selected ("All", "New" = never tried before, "Reduced" = successfully run before, "Failed" = unsuccessfully run before), or set to "Interactive" for interactive selection.
- ProductExplorerMode: Triggered  
Specify when you want to see the ProductExplorer GUI. "Triggered" = show it after all data sets have been reduced. "Enabled" = show it after each dataset. "Disabled" = never show it

Step 3: Science Reduction

Step 4: Output Organisation

execution finished: 9836 ms. Memory: 487424K Free: 121548K (25%)





# Lazy mode: Don't redo unnecessary steps

● Lazy mode for recipes.

● It works by comparing the input of the current execution with *all* the previous recipe executions:

- All files must be the same
- All files must have the same checksum
- All files must have the same date
- All recipe parameters must be the same

● If a recipe at the beginning of the workflow is set to *Not-Lazy mode*, the input of the next recipes will be new and lazy mode will not be triggered.

● Lazy mode for DataOrganizer.

- It avoids the organization of all the data in subsequent workflow runs.
- It works similar to lazy mode for recipes

● Lazy mode for PythonActor:

- Checks input/outputs, without parsing them, e.g. checking files dates.



# Configuring ESOReflex

## Workflow Instructions

To run this workflow on the demo data:

- Turn on highlighting. Choose "Tools"-> "Animate at Runtime" from top menu and set it to "1".
- Press the "Run" button OR cntrl-R to start the workflow.

To run on a different data set:

- Click on "Data" and "Data Chooser".
- If you want to use the demo data, click on "Demo".
- Press the "Run" button OR cntrl-R to start the workflow.

To monitor the progress of the workflow in more details:

- Click on "Tools" and "Log".
- Store the workflow.

## Setup Directories

Input:

- \*ROOT\_DATA\_DIR: /home/fofux/ESOReflex/data/
- \*RAWDATA\_DIR: \$ROOT\_DATA\_DIR/esoreflex\_input/ives\_Bat

Working Directories:

- \*BOOKKEEPING\_DIR: \$ROOT\_DATA\_DIR/esoreflex\_books/ives\_Bat
- \*BOOKKEEPING\_TEMP\_DIR: \$ROOT\_DATA\_DIR/esoreflex\_books/ives\_Bat
- \*BOOKKEEPING\_TEMP\_PRODUCTS\_DIR: \$ROOT\_DATA\_DIR/esoreflex\_books/ives\_Bat
- \*BOOKKEEPING\_TEMP\_COGS\_DIR: \$ROOT\_DATA\_DIR/esoreflex\_books/ives\_Bat
- \*BOOKKEEPING\_TEMP\_PRODUCTS\_COGS\_DIR: \$ROOT\_DATA\_DIR/esoreflex\_books/ives\_Bat

Output:

- \*END\_PRODUCTS\_DIR: \$ROOT\_DATA\_DIR/esoreflex\_end\_products

\*END\_PRODUCTS\_DIR or ROOT\_DATA\_DIR is changed using the Browse button. The "\*" has to be removed manually.

## Global Parameters

\*PDS\_VIEWER: tv  
File viewer to use for the inspection of input/output products

\*BOOKKEEPING\_TEMP\_PRODUCTS\_COGS\_DIR: \$ROOT\_DATA\_DIR/esoreflex\_books/ives\_Bat  
BOOKKEEPING\_DIR, TMP\_PRODUCTS\_DIR and COGS\_DIR each time the workflow is run (Lazy Mode won't work anymore).

\*END\_PRODUCT\_SUBDIR: 2010-07-07T18:10:29/Science\_DataSet\_1  
This is set automatically

\*GLOBAL\_TIMESTAMP: 2010-07-07T18:10:29  
This is set automatically

Create configuration files `~/.esoreflex/esoreflexrc` :  
`esoreflex -create-config`

Step 1:  
Data Organisation  
and Selection

Step 2:  
Creation of Master  
Calibration Files

Step 3:  
Wavelength and Response  
Calibration

Step 4:  
Spectrum  
Extraction

Step 5:  
Output  
Organisation

Some interesting parameters:

`esoreflex.python-command`

`esoreflex.inherit-environment`

`esoreflex.python-esoreflex.python-path`





# 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:

Input Files Tag:

Output Files Tag:

File Purpose Processing:

\$RecipeFailureMode

- Continue
- Stop
- Ask

- The Product Explorer allows to open the bookkeeping directory for a given file, and download a file with the command that was used to create a file.
- it is possible to go into the “Bookkeeping dir” and re-execute a recipe. The call script is in file “cmdline.sh”:

```
cd ~/reflex_data/reflex_book_keeping/fors-ima/fors_img_science_1/latest  
./cmdline.sh
```

# Customizing workflows





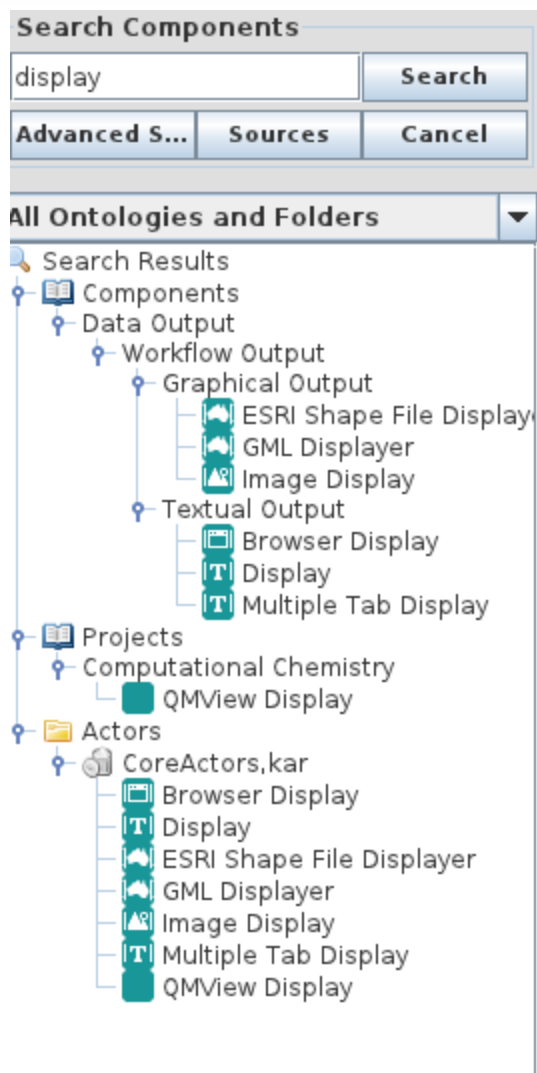


# Get Script:

```
wget http://www.eso.org/~wfreudli/tmp/miniscript.py
```

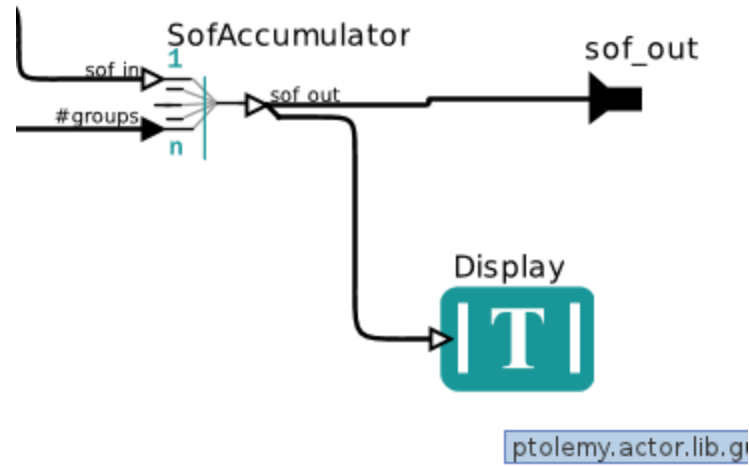


# Using the components library





# Debugging: Using the Text Display





# The Python actor

Workflow Instructions

To run this workflow on the demo data:  
- Turn on highlighting. Choose "Tools"-> "Animate at Runtime"  
from top menu and set it to "1".  
- Press the "Run" button OR cntrl-R to start the workflow.

To run on a different data set:  
- Click on "Run" button  
- Change "DataDir" to the appropriate  
subdirectory in the "DataDir" menu  
- Press the "Run" button OR cntrl-R to start the workflow.

To monitor progress of the workflow:  
- Open "Workflow" window  
- Press the "Run" button OR cntrl-R to start the workflow.

Setup Directories

Input:  
\*ROOT\_DATA\_DIR: /home/fof/eso/eso\_data  
\*RAWDATA\_DIR: \$ROOT\_DATA\_DIR/reflex\_rawfiles\_Bar  
Working Directories:  
\*WORKING\_DIR: \$ROOT\_DATA\_DIR/reflex\_workingfiles\_Bar  
\*END\_PRODUCT\_DIR: \$ROOT\_DATA\_DIR/reflex\_end\_products

Global Parameters

\*FITS\_VIEWER: fv  
Fits viewer to use for the inspection of input/output products  
\*FitsOptions: --suppress-prefix=TRUE  
fits arguments  
\*DataDir: /home/fof/eso/eso\_data  
Change "DataDir" to suit to enable  
\$WORKING\_DIR, \$TMP\_PRODUCTS\_DIR and \$END\_PRODUCT\_DIR  
each time the workflow is run (Lazy Mode won't work anymore).

= actor with interactive option

- 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 components menu on the left and search for *Eso-reflex* -> *Scripting.kar* -> *PythonActor*.





# The python actor (II)

- The real data processing looks like this:

```
#Get the input files
```

```
files = inputs.in_sof.files
```

```
#Do the stuff
```

```
for file in files:
```

```
    input_image = pyfits.open(file.name).data
```

```
    output_image= input_image * 100
```

```
    pyfits.writeto(file.name.replace(".fits", "_new.fits"), output_image)
```