



# PyMIDAS

## A Python interface to ESO-MIDAS

Opticon N3.6 face-to-face meeting 1.10.2005

Sami Maisala  
Tero Oittinen  
Marko Ullgrén  
Richard Hook

# Sampo general information

- The Sampo project is a 18-man-year contribution to Finland's entry fee to ESO
- Total duration of the project is three years
  - The project consists of several sub-projects
- The project is based in Finland
  - Four persons working at the Observatory, University of Helsinki, Finland
  - Three persons at CSC – Scientific Computing Ltd. in Espoo, Finland
  - Frequent visits to ESO HQ in Garching
- The first Sampo sub-project is PyMidas – Python interface to ESO-MIDAS



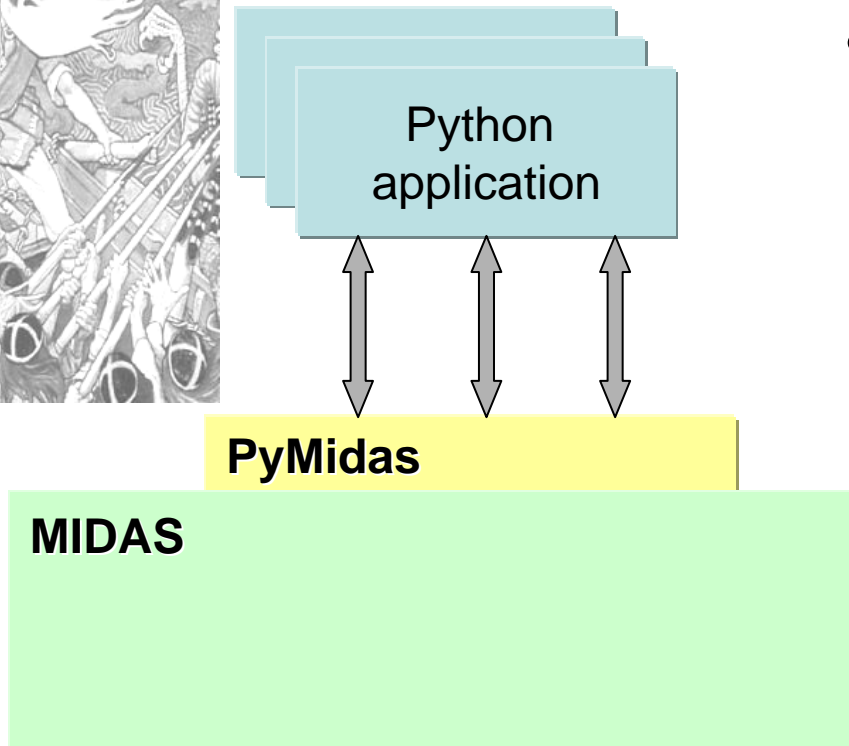
# Motivation



- Python language has become very popular within astronomy
  - Stable, robust and freely-available fully-functional, object oriented scripting language
  - PyRAF has become widely used
- ESO-MIDAS remains a widely used general purpose data processing system in the ESO community
  - A huge body of stable and tried-and-tested software
- Allow the use of ESO-MIDAS from Python
  - Co-operation with PyRAF
- Introducing the Sampo team to many aspects of astronomical software
- Leading to a useful pilot product

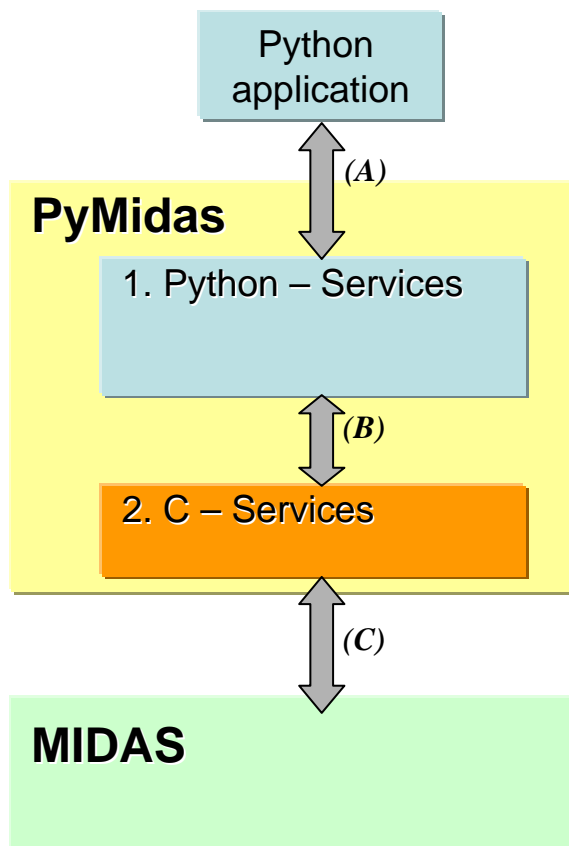
# PyMidas Overview

- Architecture is kept quite simple
  - single machine system
  - no component-container approach
- PyMidas forms a layer between Python and MIDAS



- PyMidas main tasks
  - Command parsing
  - Connecting to MIDAS background process using UNIX local sockets
  - Data transfer to/from MIDAS via files and sockets

# PyMidas - Layers



- Python application
  - import pymidas
- PyMidas
  - Python-Services
    - Offers Python style syntax for Python applications.
    - Initializing environments
    - Starting background MIDAS
  - C-Services
    - Communication with MIDAS
- MIDAS
  - running in background mode
  - command execution



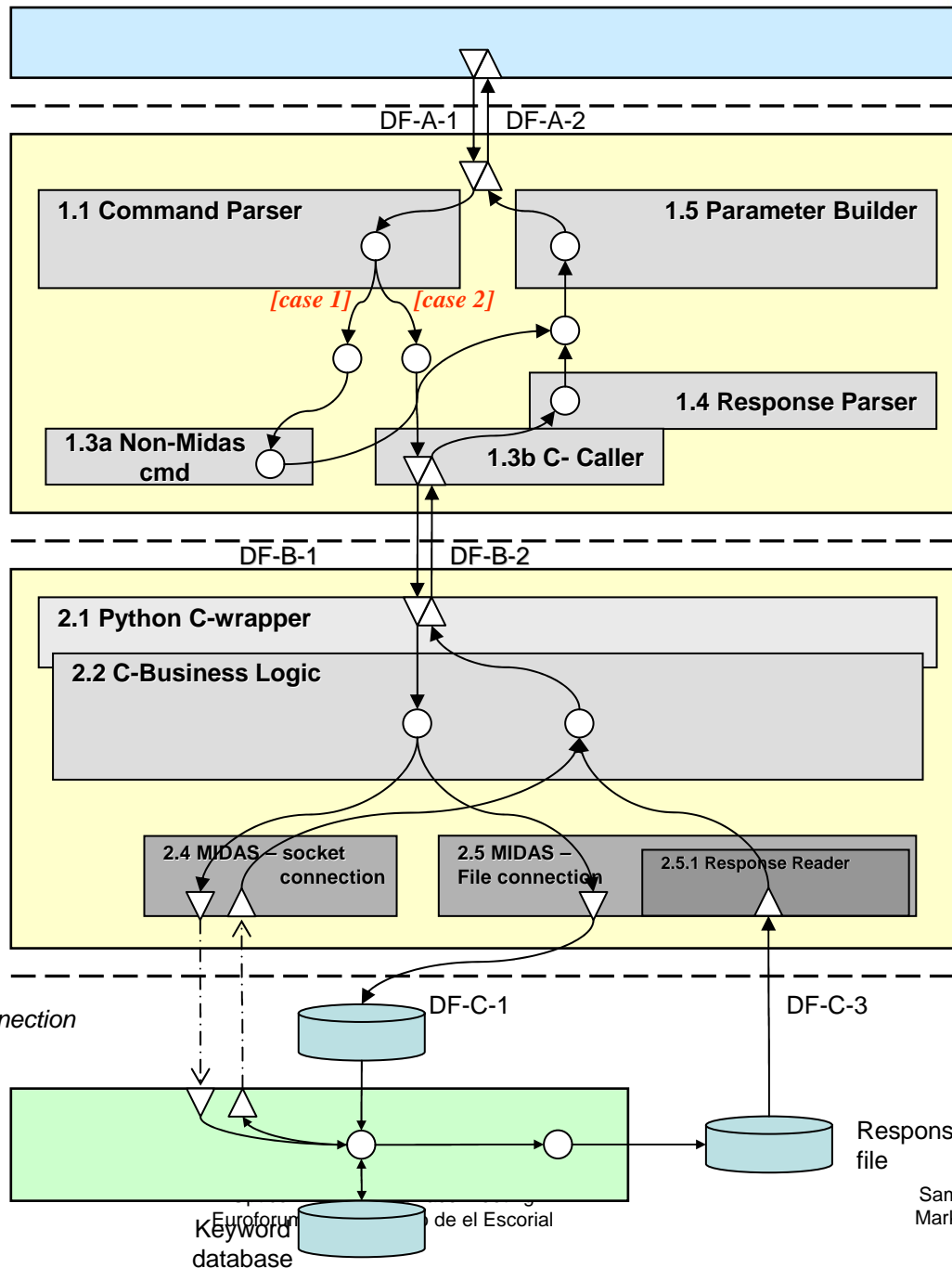
**A-Interface**

**PyMidas**

**B-Interface**

**C-Interface**

**MIDAS**



**1. Python-Services**

**2. C - Services**

"DF" = DataFlow  
 ○ = an action/a process  
 ▽ = a function call & return

# Supported platforms

- Target platform is Linux
  - tested distributions
    - Fedora core 2, 3 and 4
    - Suse 9.3 Linux
    - Red Hat Enterprise Linux 3
    - many others as well
  - Supports also Solaris
- Requires a custom MIDAS release
  - later MIDAS releases will support PyMidas
- Python 2.3 or later required



# Example script

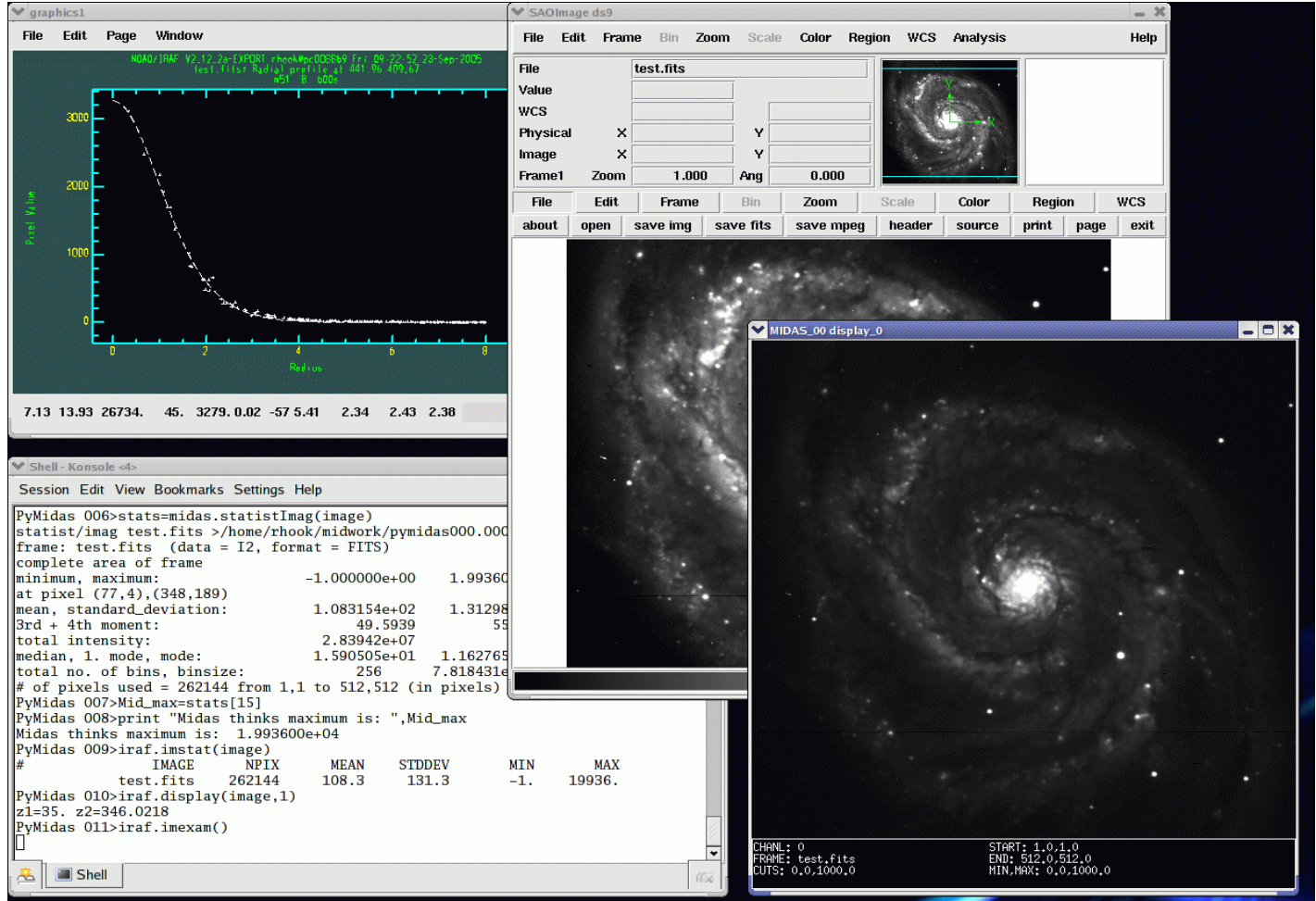


```

### A simple test script for PyMidas
# Load both PyMidas and PyRAF Python packages
from pymidas import midas
from pyraf import iraf
import os
# Main module
def run(image=None):
    # Default IRAF image of M51 if none specified
    if image == None:
        os.system('rm test*.fits')
        iraf.imcopy('dev$pix','test.fits')
        image='test.fits'
    # Get image stats from both MIDAS and IRAF
    midas.do('stat/image '+image)
    iraf.images()
    iraf.imstat(image)
    # Create the MIDAS image display
    midas.createDisp()
    # Display the input image using MIDAS
    midas.loadImag(image,'cuts=0,1000')
    # Set some names for output files
    outdr='test_drz.fits'
    outwt='test_wht.fits'
    # Loop over a range of angles
    for angle in range(36):
        print " ***** Angle is: ",angle*5.0," *****"
        # Rotate the image using drizzle in IRAF/STSDAS
        iraf.drizzle(image,outdr,outweig=outwt,scale=1.0+angle/10.,
kernel="turbo",rot=angle*5.0,outnx=512,outny=512)
        # Add an offset value using MIDAS
        midas.computeImag(outdr+' = '+outdr+'+'+str(angle*20.))
        # Load the image into the MIDAS
        midas.loadImag(outdr,'cuts=0,1000')

```

# Example

The screenshot displays the PyMIDAS software interface. The top-left window, titled 'graphics1', shows a profile plot with 'Pixel Value' on the y-axis (0 to 3000) and 'Radius' on the x-axis (0 to 6). A dashed line represents the profile, and a solid line represents a fit. Below the plot, a list of values is shown: 7.13 13.93 26734. 45. 3279. 0.02 -57 5.41 2.34 2.43 2.38.

The top-right window, titled 'SAOImage ds9', shows a galaxy image with a green box indicating a region of interest. The 'File' menu is open, showing options like 'open', 'save img', 'save fits', 'save mpeg', 'header', 'source', 'print', 'page', and 'exit'.

The bottom window, titled 'Shell - Konsole <4>', shows the command-line interface with the following text:

```

PyMidas 006>stats=midas.statistImag(image)
statist/imag test.fits >/home/rhook/midwork/pymidas000.000
frame: test.fits (data = I2, format = FITS)
complete area of frame
minimum, maximum:          -1.000000e+00   1.99360
at pixel (77,4),(348,189)
mean, standard_deviation:   1.083154e+02   1.31298
3rd + 4th moment:          49.5939      55
total intensity:           2.83942e+07
median, 1. mode, mode:     1.590505e+01   1.162765
total no. of bins, binsize: 256        7.818431e
# of pixels used = 262144 from 1,1 to 512,512 (in pixels)
PyMidas 007>Mid_max=stats[15]
PyMidas 008>print "Midas thinks maximum is: ",Mid_max
Midas thinks maximum is: 1.993600e+04
PyMidas 009>iraf.imstat(image)
#          IMAGE      NPIX      MEAN      STDDEV      MIN      MAX
test.fits  262144    108.3    131.3     -1.    19936.
PyMidas 010>iraf.display(image,1)
z1=35. z2=346.0218
PyMidas 011>iraf.imexam()
  
```

The bottom-right window, titled 'MIDAS\_00 display\_0', shows a zoomed-in view of the galaxy image. The status bar at the bottom of this window displays: CHANL: 0, FRAME: test.fits, CUTS: 0,0,1000,0, START: 1,0,1,0, END: 512,0,512,0, MIN,MAX: 0,0,1000,0.

# Q & A

- Discussion
- Thank you

