AstroGrid: Virtual Observatory Service

Nicholas Walton
AstroGrid and Euro-VO Technology Centre
Project Scientist
(Institute of Astronomy,
University of Cambridge)

Outline

- Online Demo of AG system
- Practical Session
- Discussion Session
  - the good
  - the bad
  - suggestions and lessons learned

AstroGrid 2006.3 Release: Aug 2006

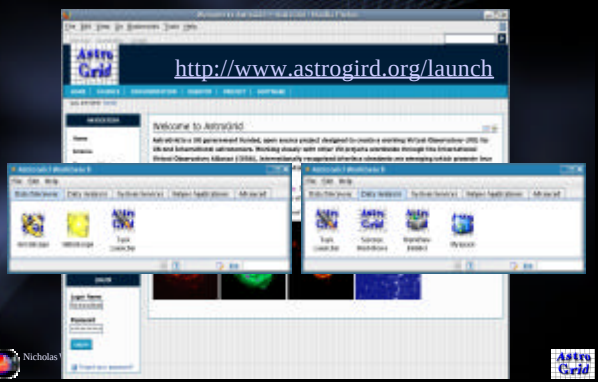
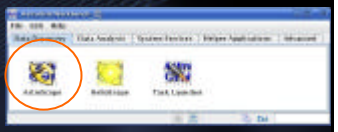


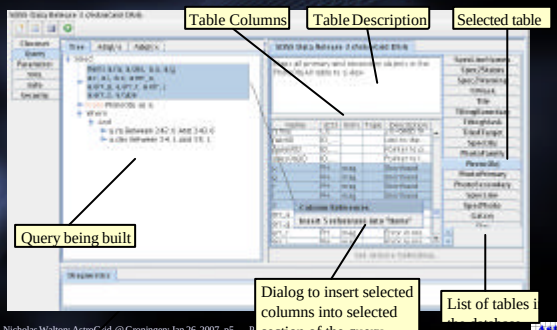
Image Discovery with AstroScope



- Workbench
  - AstroScope
  - Heliscope for solar
- Search on a position
  - RA, Dec
  - Object Name
- Investigate and explore data sources
  - images: Aladin
  - catalogues: TopCat
  - spectra: Splat, VOSpec
- Try: Arp 220

Catalogue Access

- Catalogue data available both as FITS tables on a per pointing basis
- In release – unified object catalogues



Examples to follow:

- http://www2.astrogrid.org/science/data-access
  - http://www2.astrogrid.org/science/data-access/data-access-worksheet
  - http://www2.astrogrid.org/science/science-examples-stars/iphas-catalogue-images/

Python Scripting

- Command line access to Astrogrid
- Utilises python
- Slides: Eduardo Gonzalez Gonzales
  - AstroGrid science team
  - eglez@ast.cam.ac.uk

What is Python?

- Programming language: interpreted, object oriented, high-level, dynamic semantics
- Simple, easy to learn, read, use
- Short code (compared to e.g. C/Java)
- Extensible in C/C++/Fortran/Java/anyother
- Extremely portable (PalmOS, WindowsCE, OS/2, PlayStation, BeOS, VMS, ...)
- PRODUCTIVITY

What is Python?

```
astrogrid:~$ ipython
astrogrid:~$ ipython
Python 2.4.4c1 (#2, Oct 11 2006, 21:51:02)
Type "copyright", "credits" or "license" for more information.

IPython 0.7.2 -- An enhanced Interactive Python.
?         -> Introduction to IPython's features.
%magic    -> Information about IPython's 'magic' % functions.
help      -> Python's own help system.
object? -> Details about 'object'. ?object also works, ?? prints more.

In [1]: a, b = 10, 20

In [2]: a, b = b, a

In [3]: print a, b
20 10

In [4]: def gcd(a,b):
   ...:     while a!= 0:
   ...:         a,b = b%a, a
   ...:     return b
   ...:

In [5]: print gcd(20,256)
4

In [6]:
```


Capabilities

- MySpace read, write, list, delete
- Registry query, xquery, resolve
- CEA & JES: query, execute, monitor
- SIAP, SSAP, SkyQuery
- PLASTIC
- Simbad resolver, Vizier, NED
- Control workbench UI from command line

Connecting to the ACR server

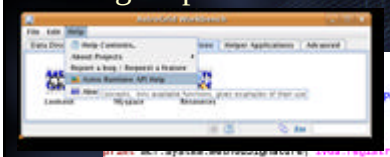
```
# Import generic modules
import sys, os

# Import XMLRPC library
import xmlrpclib

# Read endpoint from configuration file
prefix = open(os.path.expanduser("~/.astrogrid-desktop")).next().rstrip()
endpoint = prefix + "xmlrpc"

print "Endpoint to connect to is", endpoint

# Connect to server
acr = xmlrpclib.Server(endpoint)
```

This can be added into a module astrogrid.py and imported from other scripts.

Getting help

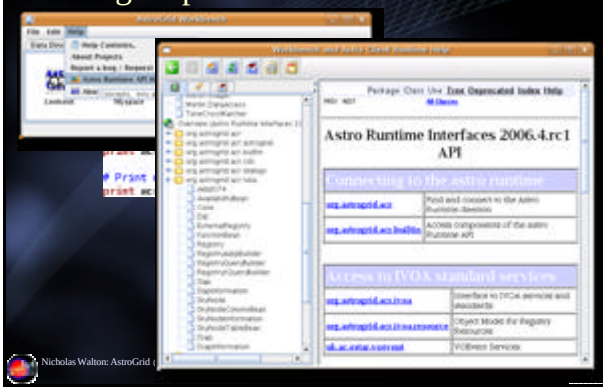
```
# Connect to server
from astrogrid import *

# Print first 10 methods provided by the XMLRPC server
print acr.system.listMethods()[:10],'\n'

# Print method signature
print acr.system.methodSignature('ivoa.registry.keywordSearch'), '\n'

# Print method help
print acr.system.methodHelp('ivoa.registry.keywordSearch')
```

Getting help



```
# Print method help
print acr.system.methodHelp('ivoa.registry.keywordSearch')
```

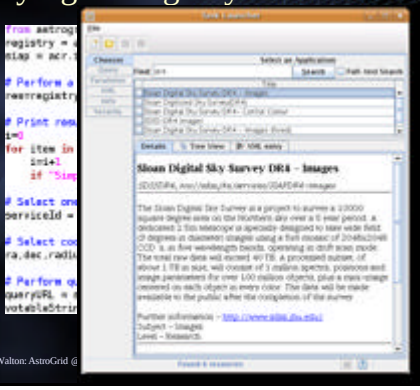

Getting help



Querying the registry

```
from astrogrid import *
registry = acr.ivoa.registry
siap = acr.ivoa.siap

# Perform a keyword search in the registry
resr=registry.keywordSearch('dr4',True)

# Print results that are SIAP services
i=0
for item in res:
    i=i+1
    if "SimpleImageAccess" in item['type']: print "%s. %s" % (i, item['id'])

# Select one of the services
serviceId = 'ivo://sdss.jhu/services/SIAPDR4-images'

# Select coordinates and radius
ra,dec,radius=180.0,0.0,0.1

# Perform query
queryURL = siap.constructQuery(serviceId,ra,dec,radius)
votableString = siap.getResults(queryURL)
```


Querying the registry



Running an application

```
# Return list of registered applications
apps=acr.astrogrid.applications.list()

# Select an application - eg. SDSS DR3 images
id="ivo://uk.ac.le.star/SDSS-DR4/images/CEA-application"

# Create template
struct = acr.astrogrid.applications.createTemplateStruct(id,"default")

# Fill in values
struct['input']['POS']['value']='180.0,0.0'
struct['input']['SIZE']['value']='0.1'

# Submit application
doc = acr.astrogrid.applications.convertStructToDocument(struct)
execId = acr.astrogrid.applications.submit(doc)

execInfo = acr.astrogrid.applications.getExecutionInformation(execId)
while execInfo['status'] == 'RUNNING' or execInfo['status'] == 'PENDING':
    time.sleep(10)
    execInfo = acr.astrogrid.applications.getExecutionInformation(execId)

# Read results
image=acr.astrogrid.applications.getResults(execId)
```


More Examples

- Save to / load from MySpace
- List MySpace contents
- Return status of submitted jobs

- See more examples in:

http://wiki.astrogrid.org/pub/Astrogrid/AcrRecipes/examples.zip

Suggested Python Exercises

- Given an input list of coordinates (or object names):
  - Search selected services (SDSS, ...) for images, retrieve results and images to MySpace.
  - Search for catalogues (SDSS, 2MASS, ...) and query them returning results to MySpace
- Run SExtractor on images, cross match catalogues

Suggested Python Exercises

- Backup contents of MySpace to local disk
- Batch upload directory contents to MySpace
- List submitted jobs, delete those canceled or terminated in error
- Send a VOTable to TOPCAT using PLASTIC