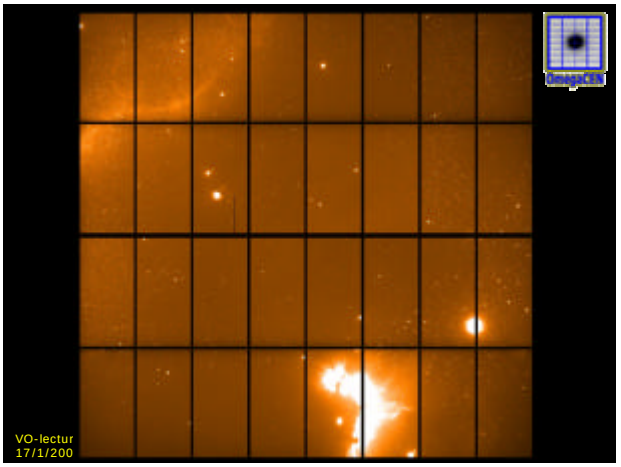
## Astro-Wise Wide Field imaging

Facilitate: handling, calibration, quality control, pipelining, user tuned research, archiving, disseminating results

100's Tbyte of image data and
10's Tbyte of catalogue data } (O)MegaCAM
With production spread over EU KIDS, OmegaWhite VESUVIO,VST16

What-ever –> object model / scalability
Where-ever -> federations, GRIDS
Who-ever -> Python as glue (+GUIs)







## VST



## Virtual Survey Telescope

raw pixel data ⇔ pipelines/cal files ⇔ catalogues
all integrated in one information system
100% data lineage
distributed services
processing GRID
Storage GRID
Methods/services GRID
NL:Leiden, Nijmegen, Groningen, Amsterdam
EU: Napoli, Munchen, Bonn, Heidelberg, Paris
World: Santiago



## AstroWise paradigm

| "Classical" paradigm Forward chaining | Target processing - Awe Backward chaining |
|---|---|
| waterfall model TIER architecture | User hunts upstream |
| driven by input raw data | Driven by query of user |
| Process in pipeline workflow | Process in bits and pieces on the fly Backward chaining |
| Operators push data | User pulls data |
| Results in releases | Provide information system |
| Static archives – publish | Dynamic archives –publish Internet |
| Raw data - obsolete | Raw data is sacred |



## Astro-Wise VO Properties Benefits integrated dynamic db

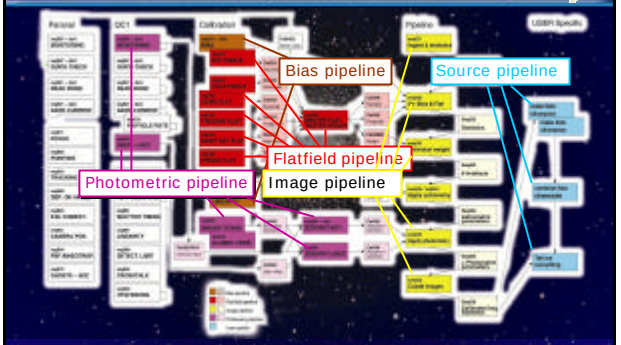
- on-the fly re-processing
- 5LS: 5 Lines Script
- All bits are traced
- Administration for parallel processing
  compute GRID SETI@home
- Global solutions –astrometry/photometry
- Build–in workflow
- Fully user tunable – own provided script
- Context: projects/surveys, instruments, mydb
- Publish directly in EURO-VO

## time

- Calibrations vary in time due to
  - Physical changes
    - eg gain of detectors, atmosphere, flexure in telescope
  - Our insight in these changes, better modeling
  - Bugs in code and improved coding

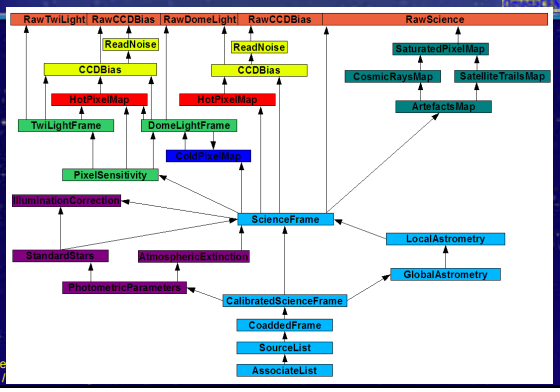


## Astro-Wise Pipelines






## TARGET






## Target processing: ++ the make metaphor

```
awe> targethot=HotPixelMap.get(date='2003-02-14', chip='A5382')
```

The processing chain is

ReadNoise <-- Bias <-- HotPixels

```
> > class HotPixelMap(ProcessTarget):
> > > def self.make()

> class ProcessTarget():
> > > def get(date, chip)  # if not exist/up-to-date then make()
> > > def exist()          # does the target  exist?
> > > def uptodate()       # is each dependency up to date?
```

Fully recursive



## Example 5LS

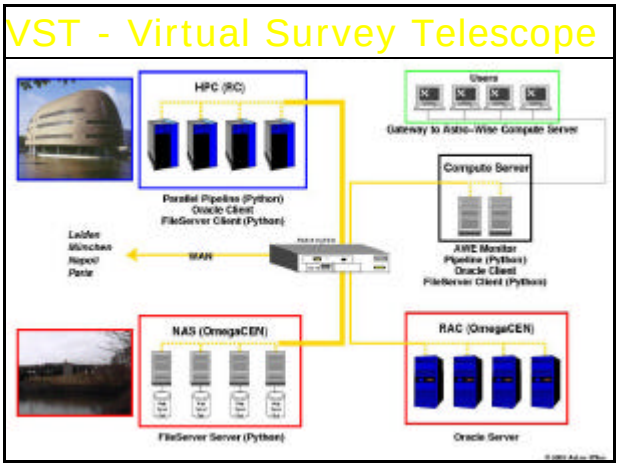
```
#Find ScienceFrames for a ccd named ccd53 and filter

    Awe> q = (ReducedScienceFrame.chip.name == 'ccd') and
      (ReducedScienceFrame.filter == '841')
# From the query result, get the rms of the sky in image
    Awe> x = [k.imstat.stdev for k in q]

# get the rms of the used Masterflat
    Awe> y = [k.flat.imstat.stdev for k in q]

# Make a plot
    Awe> pylab.scatter(x,y)
```



## Awe - GRIDS

- Collected in database – Oracle 10g snd→ federated servers using STREAms
- distributed services →Virtual Survey Telescope
  - Code base, documentation, how-to's (CVS)
  - processing GRID
  - Storage GRID
  - Access to everything archive - SQL
  - Methods/services GRID
- facilitate research environment
  - Linux Python prompt
  - Bundled in web services www.astro-wise/portal

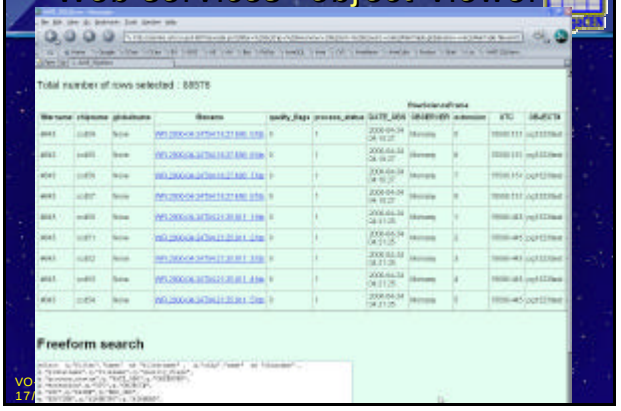
## VST - Virtual Survey Telescope




## www.ASTRO-WISE/portal november 2005

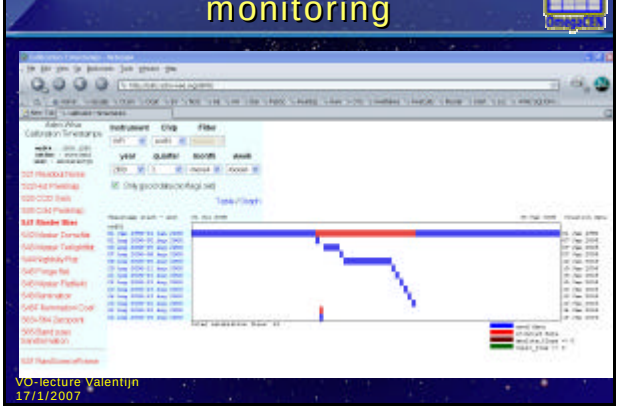



## Web services- object viewer






## QC - calibration scientist monitoring





## Web services- object maker compute GRID