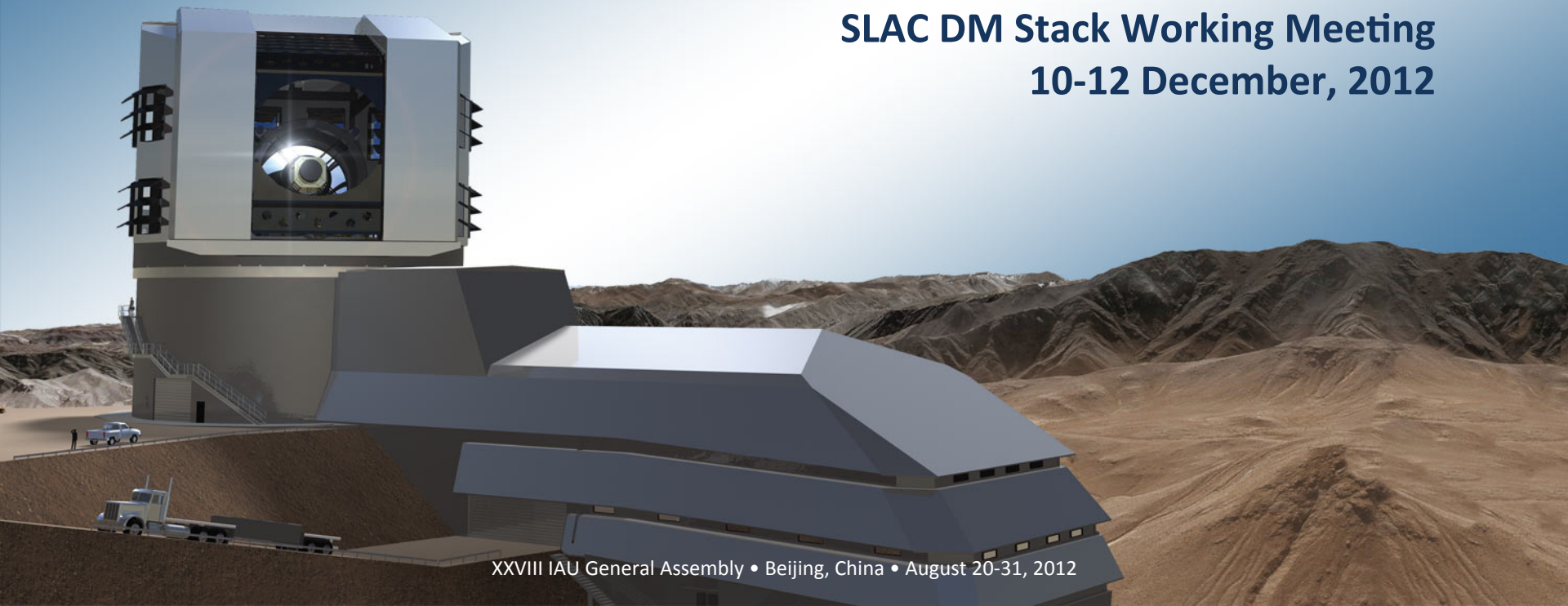
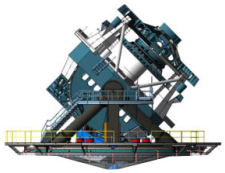
Data Management Software Stack Intro

Mario Jurić
LSST Data Management Project Scientist

SLAC DM Stack Working Meeting
10-12 December, 2012



XXVIII IAU General Assembly • Beijing, China • August 20-31, 2012



LSST Data Management Tasks

- Processes the incoming stream of images that the camera system generates.

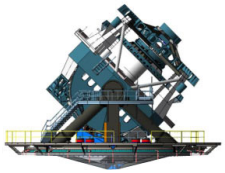
  ➔ Image Processing and Astronomical Algorithms Libraries

- Produce **transient alerts** and approximately once per year **create and archive a Data Release**, a static self-consistent collection of data products generated from all survey data taken from the date of survey initiation to the cutoff date for the Data Release.

  ➔ Middleware for data access and large-scale distributed and/or parallel computing

- **Make all LSST data available** through an interface that uses community-based standards, and **facilitate user data analysis and production of user-defined data products** at Data Access Centers (DACs) and external sites.

  ➔PB-scale databases and data access services



DM: What and Where

LSST DATA MANAGEMENT

ipac

SLAC

UI

Database

"DM Stack"

UNIVERSITY of WASHINGTON

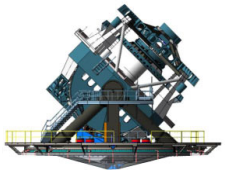
Astronomical Algorithms ("Apps")

NCSA SLAC

Middleware

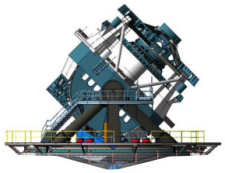
NCSA NOAO

Infrastructure



DM Stack

- **A rapidly maturing astronomical data reduction system**
  - **Most recently tested by building co-adds using SDSS Stripe 82 data**
  - **Will be used in production by the Hyper Suprime-Cam Survey on Subaru**
- **Well Prototyped (~SDSS level) Features:**
  - Instrumental signature removal
  - Point source photometry
  - Extended source photometry (model fitting)
  - Co-addition of images
  - Deblender
  - …
- **Under development (2013/14):**
  - Image differencing
  - Object characterization on multi-epoch data (StackFit/MultiFit)
  - Fault-tolerant middleware
  - …



Languages and Tools

LSST DATA MANAGEMENT

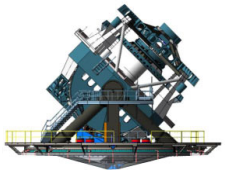
- LSST DM stack is written in Python 2.7, unless computational demands require the use of C++

- Languages
  - **C++**:
    - Computationally intensive code
    - Made available to Python via SWIG
  - **Python**:
    - All high-level code
    - Prefer Python to C++ unless performance demands otherwise

- ~60 packages (git repositories, ~corresponding to python packages)

- Build system: **scons**
- Version control: **git**

- Everything wrapped into **EUPS packages**
  - Allows one to install multiple versions of packages, and mix & match



Architecture

LSST DATA MANAGEMENT

Command-line driver scripts

Cluster execution middleware

Tasks (ISR, Detection, Co-adding, …)

Measurement Algorithms (meas_*)

Camera Abstraction Layer (obs_* packages)

…

Application Framework (comp. intensive C++, SWIG-wrapped into Python)

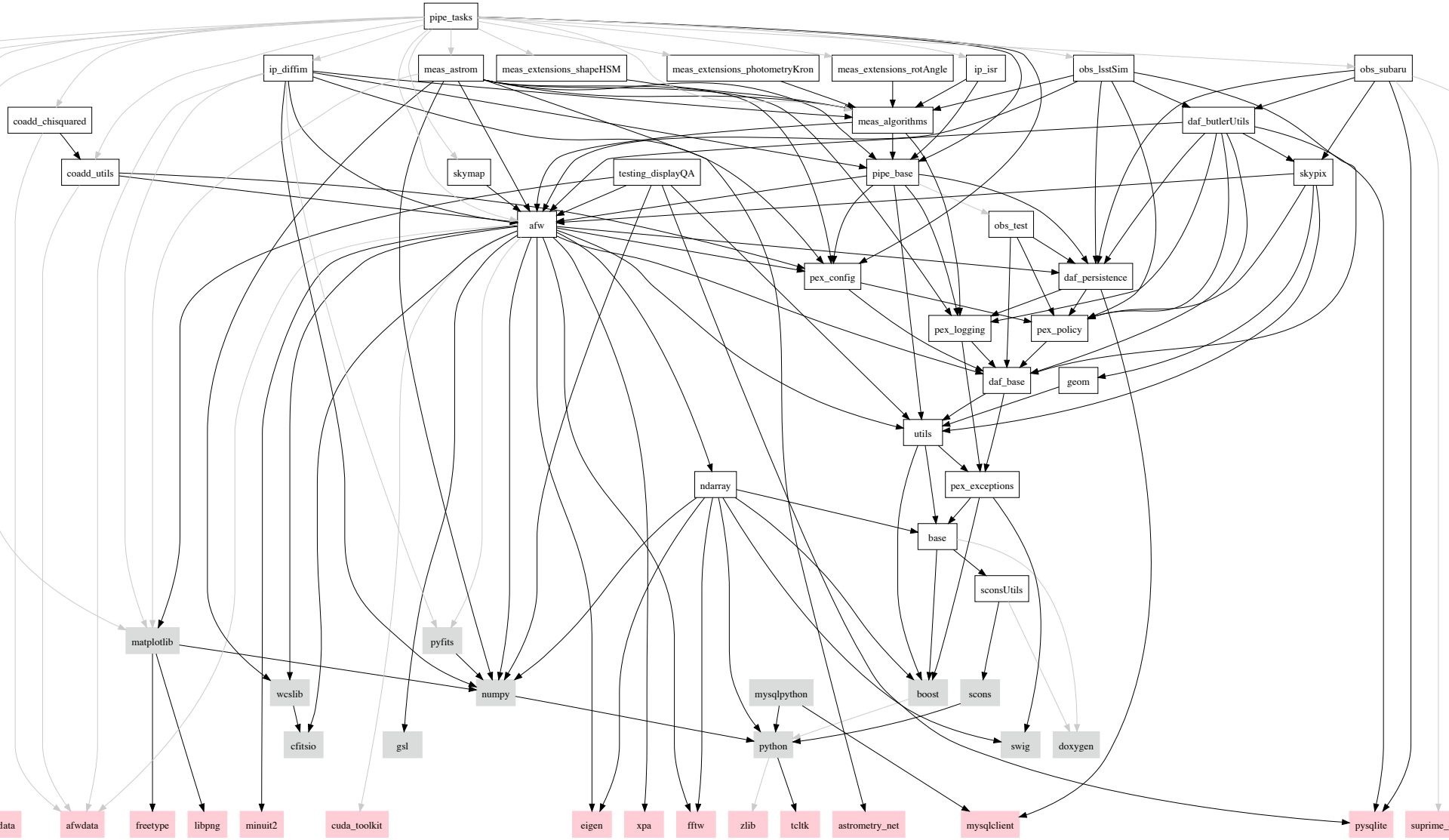
Middleware (I/O, configuration, …)

External C/C++ Libraries (Boost, FFTW, Eigen, CUDA ..)

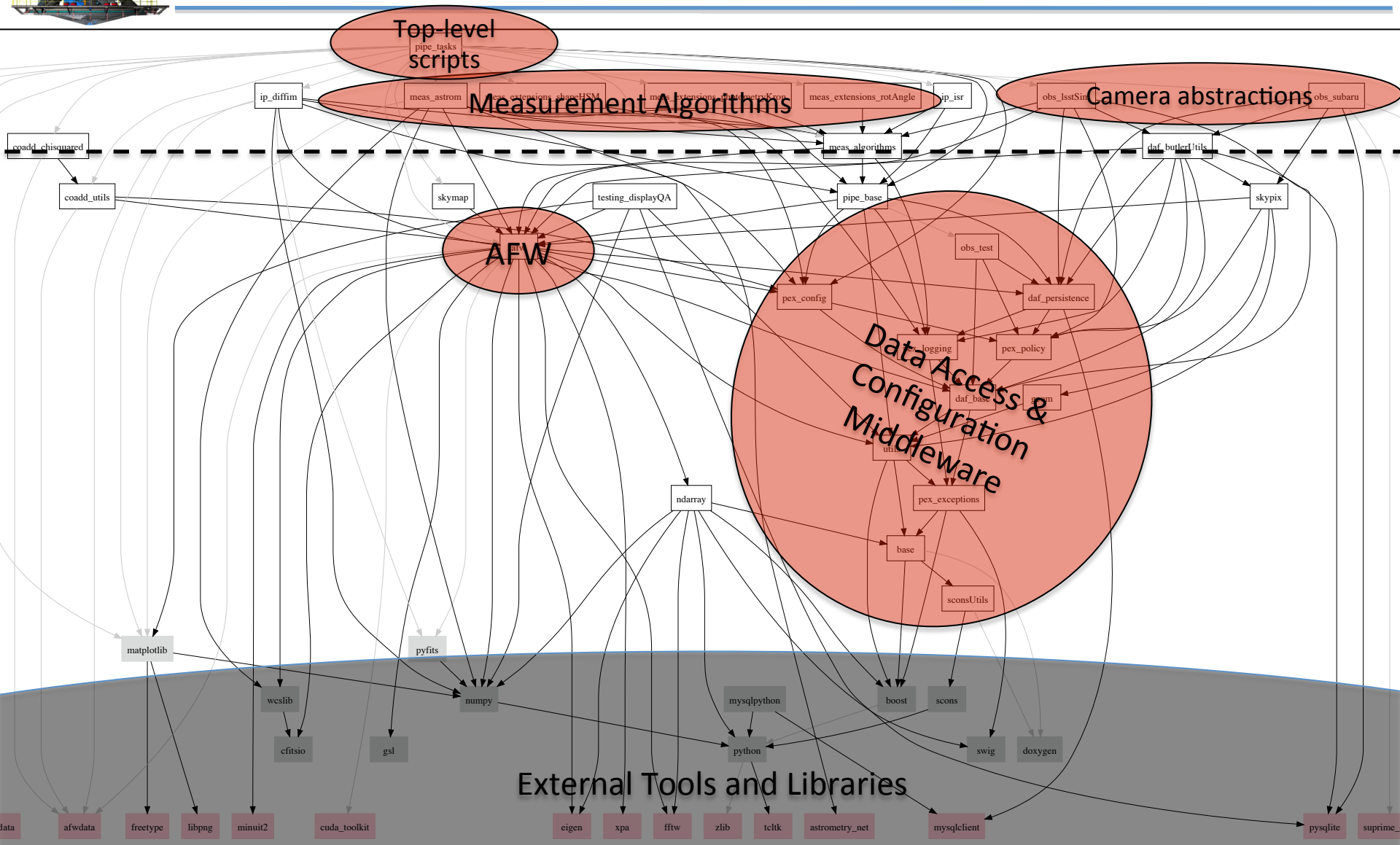
External Python Modules (numpy, pyfits, matplotlib, …)

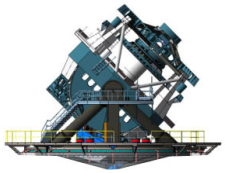
Red: Mostly C++ (but Python wrapped); Blue: Mostly Python; Black: External Libraries

Module Dependency Tree



Module Dependency Tree





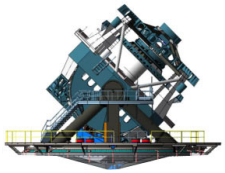
Platforms and Compilers

LSST DATA MANAGEMENT

- Operating Systems:
  - Red Hat Enterprise Linux 6 (reference platform)
    - Also known to work on:
      - RHEL5
      - Ubuntu 11.10
      - Others may work too.
  - OS X 10.7 (Lion)
    - 10.8 Mountain Lion known to work too

- Compilers:
  - gcc 4.4 or later
  - clang 3.0 or later
  - Note: plan to adopt some C++11 features in 2013

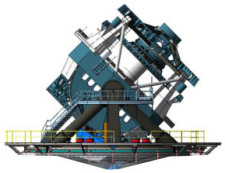
- Python 2.7
  - Currently build our own (except on OS X)



Levels of Buying into the DM Stack

LSST DATA MANAGEMENT

- Application and Measurement Libraries
  - afw and meas_* modules:
    - Image manipulation classes and algorithms
    - Flexible table classes
    - Image/Table FITS IO routines
    - Source detection routines
    - Source measurement routines (e.g., fluxes, shapes, etc…)
    - …



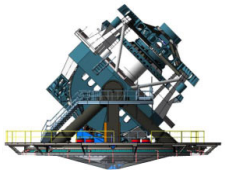
Levels of Buying into the DM Stack

LSST DATA MANAGEMENT

- Configuration and Data Access Middleware
  - pex_config, daf_* modules:
    - Configuration classes (with provenance)
    - Dataset repository management
    - Cons:
      - Less mature than afw and meas_*
      - You may already have something better in place

?

- Tasks framework
  - pipe_base, pipe_tasks, obs_* module:
    - Pros:
      - Can directly re-use complete DM tasks (e.g., instrumental signature removal)
      - Standardized argument parsing
      - Will work with any future DM orchestration mw / workflow mgmt tools
    - Cons:
      - More up-front effort than “just write main()”



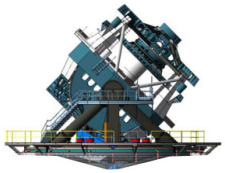
Command line examples

Process a single SDSS frame

```
processCcdSdss.py sdss /sdss/dr7/runs\
    --id run=1033 camcol=2 field=111 filter=g \
    --output /sdss/dr7/coadds
```

Co-add a number of SDSS frames

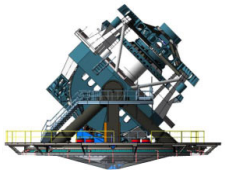
```
outlierRejectedCoadd.py sdss coadds_output \
    --id filter=g tract=3..5 patch=113,0^114,0^115,0^116,0
    --config \
        coaddName=goodSeeing \
        desiredFwhm=1.7 \
        psfMatch.kernel.active.kernelSize=13 \
        select.camcols=2,2 \
        select.strip=N \
        select.quality=2 \
        select.maxFwhm=2.5
```



Levels of Buying into the DM Stack

LSST DATA MANAGEMENT

- Pipeline Execution Middleware
- Workflow Management
- Database(*)
  - (*) Jacek and his team may have useful utilities/advice to offer on handling LSST images/catalogs with various database engines.



Levels of Buying into the DM Stack

LSST DATA MANAGEMENT

- Quality Assessment and Visualization
  - PipeQA framework:
    - Framework for writing tests operating on ***catalogs***
    - Cons:
      - Immature, will be rewritten/redesigned once we find the resources
    - Pros:
      - Exists. May be better than starting from scratch.
      - Any improvements/development you make may help DM
    - PipeQA example: http://goo.gl/zsQR0

PipeQA Example (http://goo.gl/zsQR0)

lsst1.ncsa.uiuc.edu/pipeQA/public/latest/summary.php?test=test_885335881-r_pipeQa.PsfShapeQaAnalysis&active=all

Q.A. Test Summary

Go to main rerun list.

Home | TestList | Group | Summary | Help

Astr Comp Empt Phot ap-cat Phot inst-cat Phot mod-cat Phot mod-inst Phot psf-ap Phot psf-cat Phot psf-inst Phot psf-mod PsfS Vign Zero

<<- prev-group (none) next-group ->> (none) <<- prev-r (none) next-r ->> (885335891-r)

test_885335881-r_pipeQa.PsfShapeQaAnalysis

[show description]

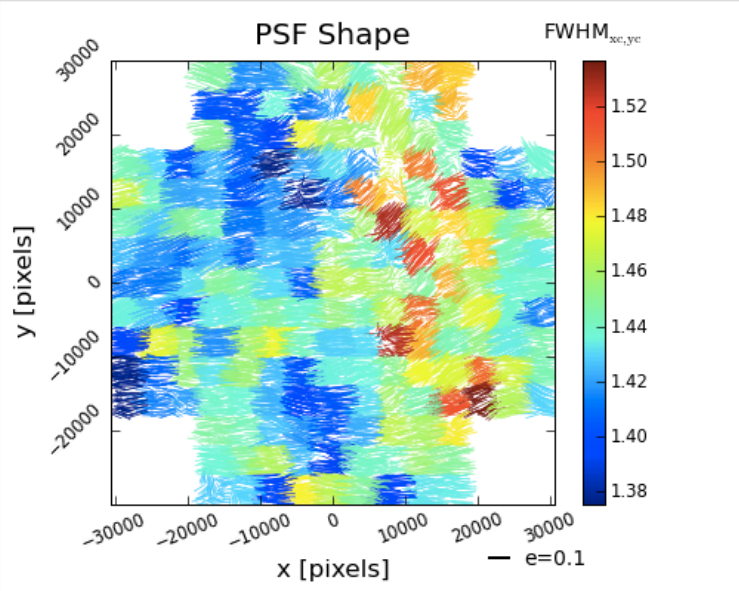


Figure 2.0: PSF ellipticity all
psfEllip-all.png: timestamp=2011-07_31 16:09:11

dataset: rplante_PT1_2_u_pt12prod_im3000 rerun=None
visit-filter: 885335881-r
Active: all

Show all

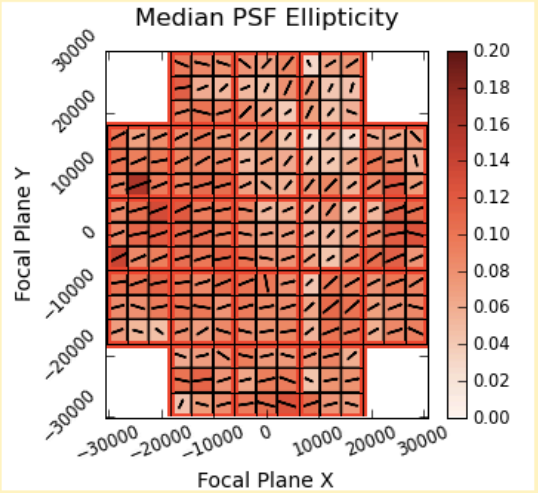
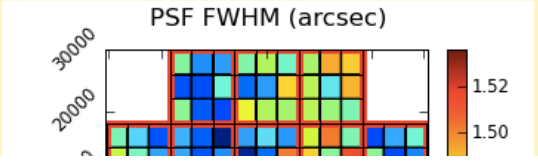
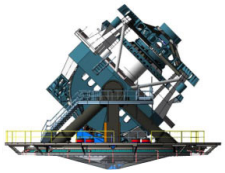


Figure 1.0: Median PSF Ellipticity
medPsfEllip.png: timestamp=2011-07_31 16:08:56

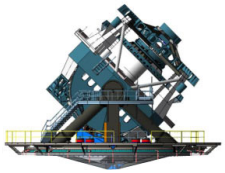
Show all





Getting the Source Code

- git repository:
  - `git://dev.lsstcorp.org/LSST/DMS` (anonymous access)
- code browser:
  - http://dev.lsstcorp.org/cgit/
  - Note: Browse subsets of repositories by appending a prefix, e.g.: http://dev.lsstcorp.org/cgit/LSST/DMS/meas will show all repositories matching DMS/meas*.
- git repository documentation
  - http://dev.lsstcorp.org/trac/wiki/GitDemoAndTutorial



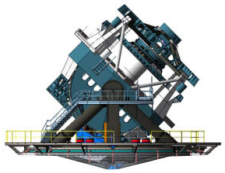
Documentation

LSST DATA MANAGEMENT

- Most (though not all) DM code is ***not*** well documented at the highest level. Documentation exists at the low level (doxygen generated). There's some documentation on the wiki, but it's poorly organized.
- Doxygen:
  - http://lsst-web.ncsa.illinois.edu/doxygen/testDoc/html/
- DM wiki:
  - http://dev.lsstcorp.org

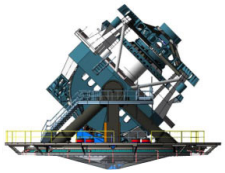
- Some packages have README files
- WARNING: Some documentation may be out of date!
- We will pay much more attention to documentation in the future.

- Consequences for this Workshop:
  - Listen carefully! Take notes!
  - Stop us if something is unclear, or if you want to know more!



Collaboration

- An “open source” development process
  - DM Stack stake-holders collaboratively propose, design, and implement new features
    - Can earn “commit rights” !
  - Overall vision and direction set by the ~~BDFLs~~ BDFTDOEs: DM Project Scientist and Manager
  - Stake-holder specific features exist in their forks of the Stack
- Proposed model (already in place with HSC-Survey DM):
  - Fork a stable release of the stack (e.g., v6_1)
  - If a feature is missing, implement it yourself (if DM hasn’t done it already!)
  - Submit patches with new features upstream (to DM)
  - Submit bug reports (and fixes) upstream
  - Periodically sync-up with the most recent DM release
- The tools we use designed to facilitate this (git, EUPS)

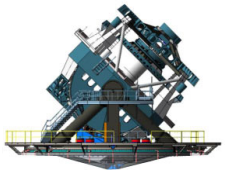


Communication

LSST DATA MANAGEMENT

Questions are welcomed and encouraged!

- Main DM mailing list:
  - lsst-data@lsstcorp.org
  - http://listserv.lsstcorp.org/mailman/listinfo/lsst-data
  - Very high traffic (chat room). Not recommended.
- Non-DM stack users list:
  - lsst-dm-stack-users@lsstcorp.org
  - http://listserv.lsstcorp.org/mailman/listinfo/lsst-dm-stack-users
  - All key DM developers subscribe to it. Recommended as primary destination for questions.
- People (in person/phone):
  - SLAC: K-T, Gregory
  - Princeton/East Coast: Robert
  - General: Mario



Bug Trackers

- Tickets
- Trac on http://dev.lsstcorp.org
- Have to open a Trac account first – just ask us.