A Tour of LSST Data Management

Kian-Tat Lim
DM Project Engineer
and System Architect

LSST 2015
Bremerton, WA | August 16 - 22, 2015
What We Do

• Accept and archive images and metadata
• Generate data products
• Provide Open Source software "stack"
• Provide limited science user facilities
Accept and Archive Images and Metadata

- Science
- Calibration
- Engineering
- Auxiliary Instrumentation
- Commands, Events, Telemetry
Generate Data Products

• Data Products Definition Document: http://ls.st/dpdd
• Level 1
  – Alert Production
• Level 2
  – Data Release Production
    – Level 1 Reprocessing
• Calibration Products Production
  – Nightly
  – Periodic
  – Annual
Level 1 Requirements

• High velocity, near-real-time
  – Updates every 37 seconds (but not streaming)
  – Reliably capture measurements as they occur
• Measurements of differences of each image from template
  – Dia = Difference Image Analysis
• DiaSources and forced DiaSources contain measurements
• DiaObjects summarize properties over time
• SSOObjects for Solar System bodies
## Level 1 – Table Sizes

<table>
<thead>
<tr>
<th>Table name</th>
<th># columns in DR1</th>
<th># rows in db [B]</th>
<th>Data size [TB]</th>
</tr>
</thead>
<tbody>
<tr>
<td>DiaObject</td>
<td>57</td>
<td>13-&gt;33</td>
<td>4-&gt;13</td>
</tr>
<tr>
<td>SSOObject</td>
<td>69</td>
<td>&lt;0.01</td>
<td>&lt;0.005</td>
</tr>
<tr>
<td>DiaSource</td>
<td>45</td>
<td>2-&gt;36</td>
<td>0.4-&gt;12</td>
</tr>
<tr>
<td>DiaForcedSource</td>
<td>9</td>
<td>12-&gt;252</td>
<td>0.6-&gt;12</td>
</tr>
</tbody>
</table>
Level 1 Images

- Calibrated visit images
- Difference images
- Image differencing templates
- "Postage stamps"
Level 2 Requirements

- Annual reprocessing of all data to date
- Consistent, read-only data warehouse
- Large volume, 10+ PB eventually
- Complex exploratory and analytical queries
  - 1 million queries per day, seconds to hours response time
- Objects contain summarized properties (divided into commonly-used and infrequently-used)
- Sources contain single-epoch measurements (if necessary)
- ForcedSources contain forced photometry measurements
# Level 2 – Table Sizes

<table>
<thead>
<tr>
<th>Table name</th>
<th># columns in DR1</th>
<th># rows [B]</th>
<th>Data size [TB]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object (narrow)</td>
<td>243</td>
<td>17-&gt;38</td>
<td>22-&gt;64</td>
</tr>
<tr>
<td>Object (extras)</td>
<td>4,227</td>
<td>Varies, largest up to ~500-&gt;~1,200</td>
<td>357-&gt;1,043</td>
</tr>
<tr>
<td>Source</td>
<td>113</td>
<td>320-&gt;6,340</td>
<td>127-&gt;3,445</td>
</tr>
<tr>
<td>ForcedSource</td>
<td>7</td>
<td>885-&gt;38,500</td>
<td>35-&gt;1,540</td>
</tr>
</tbody>
</table>
Level 2 Images

- Calibrated visit exposures
- Master calibration images
- Deep coadds
- Small sections of other coadds
  - Short-period
  - Best-seeing
  - PSF-matched
- Special programs
Provide Open Source Software "Stack"

- "Enable LSST science by creating a well documented, state-of-the-art, high-performance, scalable, multi-camera, open source, O/IR survey data processing and analysis system"
Provide Limited Science User Facilities

- Chilean Data Access Center
- US Data Access Center
Level 3 Requirements

• User-defined processing and external catalog import
• Queries in database
• Processing "alongside" database
• Results into per-user areas
• Can be shared with group or public
• Must allow joins with Level 2
• Processing of Level 2 intermediates ("Level 2.5")
What We Do Not Do Do

• Archive or process Camera images during Construction
  – Free to use stack, can provide help
  – Need to be compatible for transfer before Commissioning

• Science
  – Measurements, not results
  – Characterization, not classification
Tour of LSST Stack

- Core stack
- Top-level products (currently)
  - lsst_distrib
  - qserv_distrib
  - dax_webserv, _client
  - lsst_sims
Languages, Platforms, and Tools

• The LSST 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
• Red Hat Enterprise Linux 6 (gcc), MacOS X 10.9 (clang), others work

• Build system: scons
• Version control: git

• Everything wrapped into EUPS packages
  – Allows one to install multiple versions of packages, and mix & match
<table>
<thead>
<tr>
<th>Red: Mostly C++ (but Python wrapped); Libraries</th>
<th>Blue: Mostly Python; Libraries</th>
<th>Black: External Libraries</th>
</tr>
</thead>
<tbody>
<tr>
<td>Application Framework (comp. intensive C++, SWIG-wrapped into Python)</td>
<td>Camera Abstraction Layer (obs_* packages)</td>
<td>...</td>
</tr>
<tr>
<td>Measurement Algorithms (meas_*)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Tasks (ISR, Detection, Co-adding, ...)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Command-line driver scripts</td>
<td>Cluster execution middleware</td>
<td></td>
</tr>
</tbody>
</table>
Basics

• **ndarray**: efficient n-dimensional numpy-compatible arrays
• **geom**: Cartesian and spherical geometry in Python
• **daf_base**
  – DateTime: TAI/UTC, MJD/JD/J2000, string conversions
  – PropertySet/List: simple key/value pairs (e.g. FITS headers)
• **pex_exceptions**: basic exception handling
• **pex_logging**: basic logging
Application Framework

• A library of primitives for image processing
  • **image**: pixel array, mask, metadata including WCS, filter, calibration, defects
  • **math**: kernels, convolution, Gaussian processes, interpolation, least squares
  • **coord**: on sky and observatory positions
  • **geom**: Cartesian and spherical geometry in C++
  • **cameraGeom**: camera physical and electronic description
  • **detection**: footprint, Peak, Source, PSF
  • **table**: persistence of detections and exposure metadata
Measurement

• **meas_base**: framework with plugins
  – **meas_extensions_multiShapelet**
  – **meas_extensions_photometryKron**
  – **meas_extensions_shapeHSM**

• **meas_algorithms**

• **meas_deblender**

• **meas_multifit**: multi-component galaxy properties
Pipeline (Task) Framework

- **pex_config**: parameter and plugin configuration
- **pipe_base**: argument handling
Astrometric and Photometric Calibration

- **meas_astrom**: single-visit astro/photometric calibration
- **meas_mosaic**: cross-visit astro/photometric calibration
Coaddition and Differencing

- **skymap**: overlapping rectangular patches covering the sky
- **coadd_utils**: coaddition utilities
- **ip_diffim**: difference imaging
ISR and Camera-Specific Customization

- **ip_isr**: instrument signature removal
- **obs_***: observatory-specific packages
  - test
  - cfht
  - decam
  - lsstSim
  - sdss
  - sst
  - subaru
  - wiyn
Top Level pipe_tasks

- processCcd/processCoadd/processImage
  - calibrate (repair, astrometry, measurePsf)
  - detection
  - deblend
  - measurement
- makeSkyMap
- makeCoaddTempExp (warpAndPsfMatch)
- assembleCoadd (selectImages, interpImage, matchBackgrounds, scaleZeroPoint)
- processCoadd
- forcedPhot
- imageDifference (registerImage)
Command line examples

Process a single SDSS frame

```bash
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

```bash
makeCoaddTempExp.py calexp_dir --output coadd_u_dir \ 
   --id filter=u tract=0 patch=375,0^375,1^375,2
assembleCoadd.py coadd_u_dir \ 
   --config maxMatchResidualRatio=2.5 \ 
   maxMatchResidualRMS=0.5 \ 
   --id filter=u tract=0 patch=375,0^375,1^375, run=5823
```
Data Butler and Database

- **daf_persistence**: general persistence framework for repositories of datasets (Data Butler)
- **daf_butlerUtils**: astronomy-specific tailoring of Butler framework
- **db**: generic SQL database utilities
- **qserv**: scalable distributed database with spherical partitioning and neighbor query support
Execution Control Middleware

- **ctrl_orca**: orchestration of jobs and task graphs
- **ctrl_execute**: parallel task execution tool
- **ctrl_platform_***: platform configurations
- **ctrl_stats**: tools for computing execution statistics
- **ctrl_events**: event messaging system
- **ctrl_evmon**: event monitoring (complex event processing)
QA Framework

- **testing_pipeQA**: compute QA measures and plots (bit-rotted)
- **testing_displayQA**: generate Web pages with QA results
Where To Start

• LSST Software User's Guide: https://confluence.lsstcorp.org/x/HAE-

• LSST DM Developer Guide: https://confluence.lsstcorp.org/x/bYJI

• DM Decision Making Process: https://confluence.lsstcorp.org/x/FYBaAQ
Communications

• Talk to your T/CAM and/or Tech Lead
• Talk to "management"
• HipChat (includes one-to-one video)
• Discourse (trial)
• ls.st/sup Google Hangout, BlueJeans
• dm-devel mailing list
• Confluence, JIRA
Construction/Commissioning Phases

Organization

DM Project Manager
DM Project Scientist
DM Project Engineer

Science Quality and Reliability Engineering

Science Database and Data Access Services (SLAC)

Processing Services/ Tools and Archive Site (NCSA)

Base Site and Long-Haul Networks (NOAO)

Alert Production and Global Calibration (UW)

Data Release Production, Calibration Products, and App Framework (Princeton)

Science User Interface (IPAC)

DM System Architect
DM System Interfaces Scientist