Large Synoptic Survey Telescope
Data Management Applications Design

Mario Jurić*, R.H. Lupton, T. Axelrod, J.F. Bosch, P.A. Price,
G.P. Dubois-Felsmann, Ž. Ivezić, A.C. Becker, J. Becla,
A.J. Connolly, J. Kantor, K-T Lim, D. Shaw,
for the LSST Data Management

Friday 23rd October, 2015, 13:23hrs

Abstract

The LSST Science Requirements Document (the LSST SRD) specifies a set of data product guidelines, designed to support science goals envisioned to be enabled by the LSST observing program. Following these guidlines, the details of these data products have been described in the LSST Data Products Definition Document (DPDD), and captured in a formal flow-down from the SRD via the LSST System Requirements (LSR), Observatory System Specifications (OSS), to the Data Management System Requirements (DMSR). LSST Data Management subsystem has the responsibility for design, implementation, deployment and execution of software pipelines necessary to generate these data products. This document, in conjunction with the UML Use Case model (LDM-134), describes the design of the scientific aspects of those pipelines.

*Please direct comments to <mjuric@lsst.org>.

Contents

1 Preface

The purpose of this document is to describe the design of pipelines belonging to the Applications Layer of the Large Synoptic Survey Telescope (LSST) Data Management system. These include most of the core astronomical data processing software that LSST employs.

The intended audience of this document are LSST software architects and developers. It presents the baseline architecture and algorithmic selections for core DM pipelines. The document assumes the reader/developer has the required knowledge of astronomical image processing algorithms and solid understanding of the state of the art of the field, understanding of the LSST Project goals and concepts, and has read the LSST Science Requirements (SRD) as well as the LSST Data Products Definition Document (DPDD).

This document should be read in conjunction with the LSST DM Applications Use Case Model (LDM-134). They are intended to be complementary, with the Use Case model capturing the detailed (inter)connections between individual pipeline components, and this document capturing the overall goals, pipeline architecture, and algorithmic choices.

Though under strict change control[1], this is a ***living document***. Firstly, as a consequence of the "rolling wave" LSST software development model, the designs presented in this document will be refined and made more detailed as particular pipeline functionality is about to be implemented. Secondly, the LSST will undergo a period of construction and commissioning lasting no less than seven years, followed by a decade of survey operations. To ensure their continued scientific adequacy, the overall designs and plans for LSST data processing pipelines will be periodically reviewed and updated.

[1] LSST Docushare handle for this document is `LDM-151`.

2 Introduction

2.1 LSST Data Management System

To carry out this mission the Data Management System (DMS) performs the following major functions:

- Processes the incoming stream of images generated by the camera system during observing to produce transient alerts and to archive the raw images.
- Roughly once per year, creates and archives a Data Release ("DR"), which is 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. The data products (described in detail in the DPDD), include measurements of the properties (shapes, positions, fluxes, motions, etc.) of all detected objects, including those below the single visit sensitivity limit, astrometric and photometric calibration of the full survey object catalog, and limited classification of objects based on both their static properties and time-dependent behavior. Deep coadded images of the full survey area are produced as well.
- Periodically creates new calibration data products, such as bias frames and flat fields, that will be used by the other processing functions, as necessary to enable the creation of the data products above.
- Makes all LSST data available through an interface that utilizes, to the maximum possible extent, community-based standards such as those being developed by the Virtual Observatory ("VO"), and facilitates user data analysis and the production of user-defined data products at Data Access Centers ("DAC") and at external sites.

The overall architecture of the DMS is discussed in more detail in the Data Management System Design (DMSD) document. The overall architecture of the DMS is shown in Figure 1.

This document discusses the role of the Applications layer in the first three functions listed above (the functions involving *science pipelines*). The fourth is discussed separately in the SUI Conceptual Design Document (SUID).

Figure 1: Architecture of the Data Management System

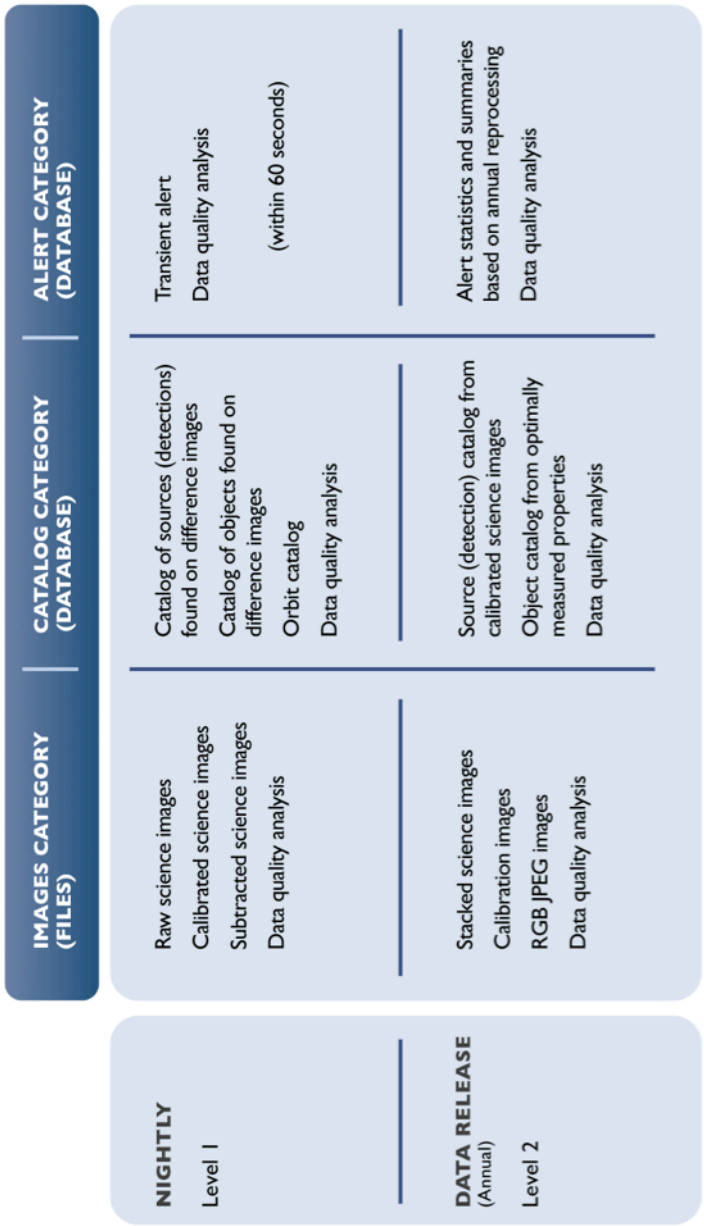

Figure 2: Organization of LSST Data Products

2.2 Data Products

The LSST data products are organized into three groups, based on their intended use and/or origin. The full description is provided in the Data Products Definition Document (DPDD); we summarize the key properties here to provide the necessary context for the discussion to follow.

- **Level 1** products are intended to support timely detection and follow-up of time-domain events (variable and transient sources). They are generated by near-real-time processing the stream of data from the camera system during normal observing. Level 1 products are therefore continuously generated and / or updated every observing night. This process is of necessity highly automated, and must proceed with absolutely minimal human interaction. In addition to science data products, a number of related Level 1 "SDQA"[2] data products are generated to assess quality and to provide feedback to the Observatory Control System (OCS).

- **Level 2** products are generated as part of a Data Release, generally performed yearly, with an additional data release for the first 6 months of survey data. Level 2 includes data products for which extensive computation is required, often because they combine information from many exposures. Although the steps that generate Level 2 products will be automated, significant human interaction may be required at key points to ensure the quality of the data.

- **Level 3** products will be generated by the users of LSST, using LSST software and/or hardware. LSST DM is required to facilitate the creation of Level 3 data products by providing suitable APIs, software components, and computing infrastructure, but will not by itself create any Level 3 data product. Once created, Level 3 data products may be associated with Level 1 and Level 2 data products through database federation. Where appropriate, the LSST Project, with the agreement of the Level 3 creators, may incorporate user-contributed Level 3 data product pipelines into the DMS production flow, thereby promoting them to Level 1 or 2.

The organization of LSST Data Products is shown in Figure 2.

[2]Science Data Quality Analysis

Level 1 and Level 2 data products that have passed quality control tests will be accessible to the public without restriction. Additionally, the source code used to generate them will be made available, and LSST will provide support for builds on selected platforms.

2.3 Science Pipelines Overview

We recognize four major groups of science pipelines residing in the Applications layer:

- **Level 1 Pipelines**, grouped under the **Alert Production** element of the WBS, are designed to generate Level 1 data products. These include:
  - ***Single Frame Processing ("SFM") Pipeline***, (WBS 02C.03.01) to reduce acquired visits and detect and characterize astrophysical sources present in these visits.
  - ***Image Differencing Pipeline*** (WBS 02C.03.04), to create difference images, and detect and characterize sources in them.
  - ***Association Pipeline*** (WBS 02C.03.02), to associate sources detected in the difference images with known objects.
  - ***Alert Generation Pipeline*** (WBS 02C.03.03), to generate and transmit alerts to time-domain events (e.g., transients) to the astronomical community, and
  - ***Moving Object Pipeline*** (WBS 02C.03.06), to identify, link and compute orbits for Solar System objects detected in difference images.

  Level 1 pipelines run as the data are being acquired. They primarily focus on image differencing, and the reduction of information extracted from difference images. The algorithms they employ are designed and chosen to complete processing on minute (alert production) to day (Solar System object pipelines) time scales. They are also rerun as a part of Data Release Production (DRP), potentially in somewhat different configurations to achieve greater precision at the expense of increased runtime.

- **Level 2 Pipelines** run annually or semi-annualy (for the first year of data), and are designed to generate deep co-adds and catalogs stemming from analysis of direct image data. These include:
  - ***PSF Estimation Pipeline***, (WBS 02C.04.03), to estimate the PSF properties and variation across the focal plane for each visit, to the degree of precision required by the SRD. Note that the work of this pipeline goes beyond the typical single-CCD PSF estimation present in the SFM pipeline.
  - ***Image Coaddition Pipeline***, (WBS 02C.04.04), to generate and characterize coadded images of the sky, as well as create templates for image differencing.
  - ***Object Detection and Deblending***, (WBS 02C.04.05), to detect sources in images of the sky and decompose them individual astronomical objects.
  - ***Object Characterization Pipeline***, (WBS 02C.04.06), to characterize (perform measurements of) astrophysical objects detected in LSST imaging (both in single frames and coadds).
- **Calibration Pipelines** process the collected calibration data and perform calibration of LSST instruments and data products. These include:
  - ***Calibration Products Pipeline***, (WBS 02C.04.02), that generates the necessary calibration data products (e.g., master flats, biases, atmospheric models, etc.). It is run periodically as new calibration data are acquired.
  - ***Photometric Calibration Pipeline***, (WBS 02C.03.07), that performs global photometric self-calibration of the Level 2 dataset.
  - ***Astrometric Calibration Pipeline***, (WBS 02C.03.08), that performs global astrometric self-calibration of the Level 2 dataset.

  The calibration products pipeline is also rerun as a part of Data Release Processing. Global self-calibration steps run in DRP only.
- **Science Data Quality Assessment (SDQA) pipelines and toolkits**, to enable collection, computation, visualization, monitoring and

analysis of data quality metrics across all pipelines. These are divided into:

  - ***Science Data Quality Assessment Pipeline***, (WBS 02C.01.02.02), that provides low-level data collection functionality for SDQA and
  - ***Science Data Quality Analyst Toolkit***, (WBS 02C.01.02.02), that provides the visualization, analysis and monitoring capabilities for SDQA.

In addition to these four, we recognize two other, cross-cutting, elements of DMS functionality:

- **Common Image and Catalog Processing framework**, known as the **Application Framework (afw)**, (WBS 02C.03.05, 02C.04.01), that collects base classes and algorithms used by the DM Applications layer. The framework is split in two WBS elements, to reflect the multi-institutional nature of the work, but is functionally viewed as a single, integrated, component (class library).
- The **Science Pipeline Toolkit**, (WBS 02C.01.02.03), a collection of software components (and design principles) designed to enable construction of Level 3 pipelines relying on reusable lower-level components produced in support of other LSST DM software.

2.3.1 Level 1 Pipelines Overview

The production of Level 1 products is generally performed nightly, directly fed by the output data stream from the Camera SDS[3] during observing. This data stream contains both unprocessed (raw) camera images, and images that have been corrected for crosstalk by the SDS on the Summit. The normal observing pattern is to take two 15 second exposures of the same field in immediate succession. These two exposures together form a *visit*, which is the typical image data unit processed by the rest of the DM system.

The logical flow of Level 1 processing is shown in the Use Case diagram presented in Figure 3. For every observation, the following sequence of events will unfold:

[3]Science Array Data Acquisition (DAQ) Subsystem

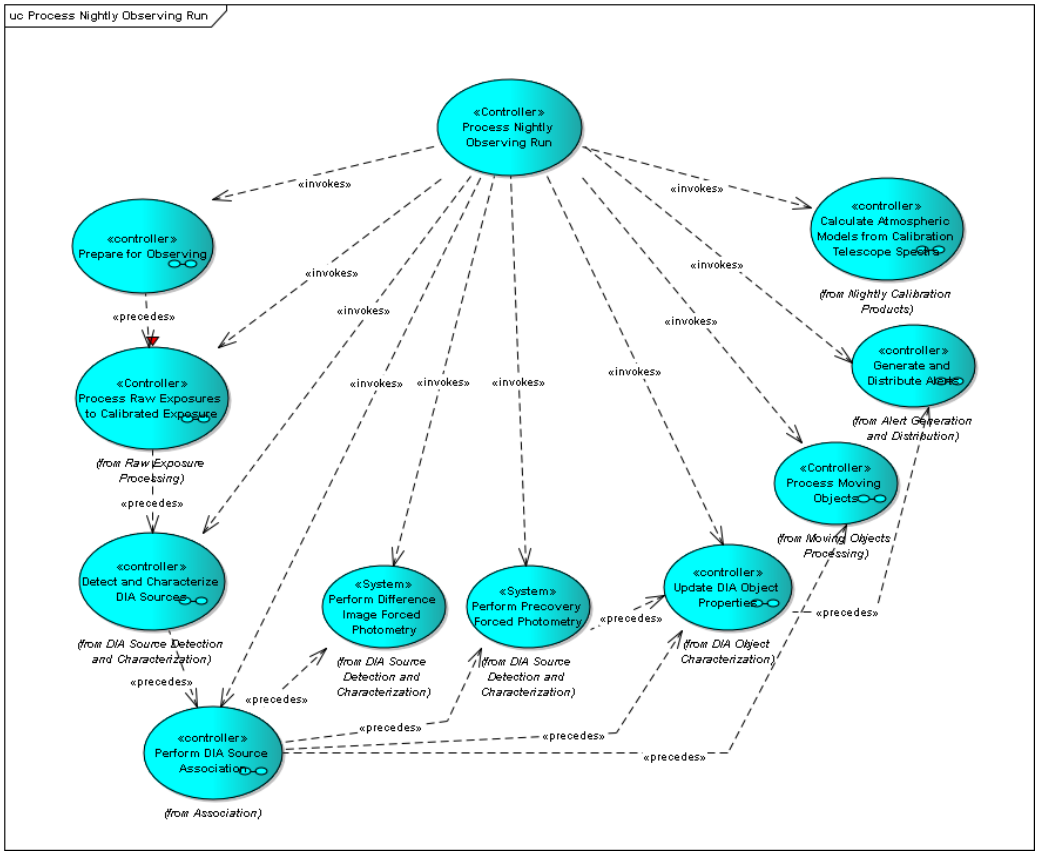

Figure 3: Level 1 Processing Flow Diagram

1. A visit is acquired (`Prepare for Observing`) and reduced (`Process Raw Exposures to Calibrated Exposure`) to a single *visit image*. This includes instrumental signature removal[4], combining of snaps, etc.

2. The visit image is differenced against the appropriate template and `DIASources` are detected (`Detect and Characterize DIA Sources`). If necessary, deblending is performed at this stage.

   The flux and shape of the DIASource are measured on the difference image. PSF photometry is performed on the visit image at the position of the `DIASource` to obtain a measure of the absolute flux.

3. The Level 1 database is searched for a `DIAObject` or an `SSObject` with which to positionally associate the newly discovered `DIASource`. If no match is found, a new `DIAObject` is created and the observed `DIASource` is associated to it.

   If the `DIASource` has been associated to an `SSObject` (a known Solar System object), it will be flagged as such and an alert will be issued. Further processing will occur in daytime (`Process Moving Objects`).

4. Otherwise, the associated `DIAObject` measurements will be updated with new data (`Update DIA Object Properties`). All affected columns will be recomputed, including proper motions, centroids, light curves, nearest Level 2 `Objects`, etc.

5. An alert is issued (`Generate and Distribute Alerts`) that includes all required components, as described in the DPDD.

6. For all `DIAObjects` overlapping the field of view to which a `DIASource` from this visit has *not* been associated, forced photometry will be performed (`Perform Difference Image Forced Photometry`). No alerts will be issued for these measurements.

Within 24 hours of discovery, LSST DM system will perform *precovery* PSF forced photometry on any prior difference image overlapping the predicted position of new `DIAObjects` taken within the past 30 days (`Perform Precovery Forced Photometry`).

[4]E.g., subtraction of bias and dark frames, flat fielding, bad pixel/column interpolation, etc.

Similarly, in daytime after the nightly observing run, atmospheric models from the calibration telescope spectra will be calculated (`Calculate Atmospheric Models from Calibration Telescope Spectra`) and made available to the users.

In addition to these, the Moving Object Pipeline (MOPS; WBS 02C.03.06; `Process Moving Objects`) will also be run in daytime. It is described in its own section of this document, with a detailed design in a separate Moving Object Pipeline Design Document (MOPSD).

2.3.2 Level 2 Pipelines Overview

Figure 4 presents a high-level overview of the Level 2 data processing workflow. Logically[5], the processing begins with single-frame (visit) image reduction and source measurement, followed by global astrometric and photometric calibration, coadd creation, detection on coadds, association and deblending, object characterization, and forced photometry measurements. The UML Use Case model (LDM-134) captures these activities in the `Produce a Data Release` diagram.

The following is a high-level description of steps which occur during regular Level 2 data processing:

1. *Single Frame Processing*: Raw exposures are reduced to *calibrated visit exposures*, and `Sources` are independently detected, deblended, and measured on all visits. Their measurements (instrumental fluxes and shapes) are stored in the `Source` table. This step is performed by the ***Single Frame Processing Pipeline*** (WBS 02C.03.01).

2. *Relative calibration*: The survey is internally calibrated, both photometrically and astrometrically using the ***Astrometric*** (WBS 02C.03.08) and ***Photometric Calibration Pipelines*** (WBS 02C.03.07). Relative zero points over the focal plane and astrometric corrections are computed for every visit.

3. *Coadd creation*: Deep, seeing optimized, and short-period per-band coadds are created in *ugrizy* bands, as well as deeper, multi-color,

[5]The actual implementation may parallelize these steps to the extent possible; see LDM-230, the Automated DM Operations Document (DM OpsCon).

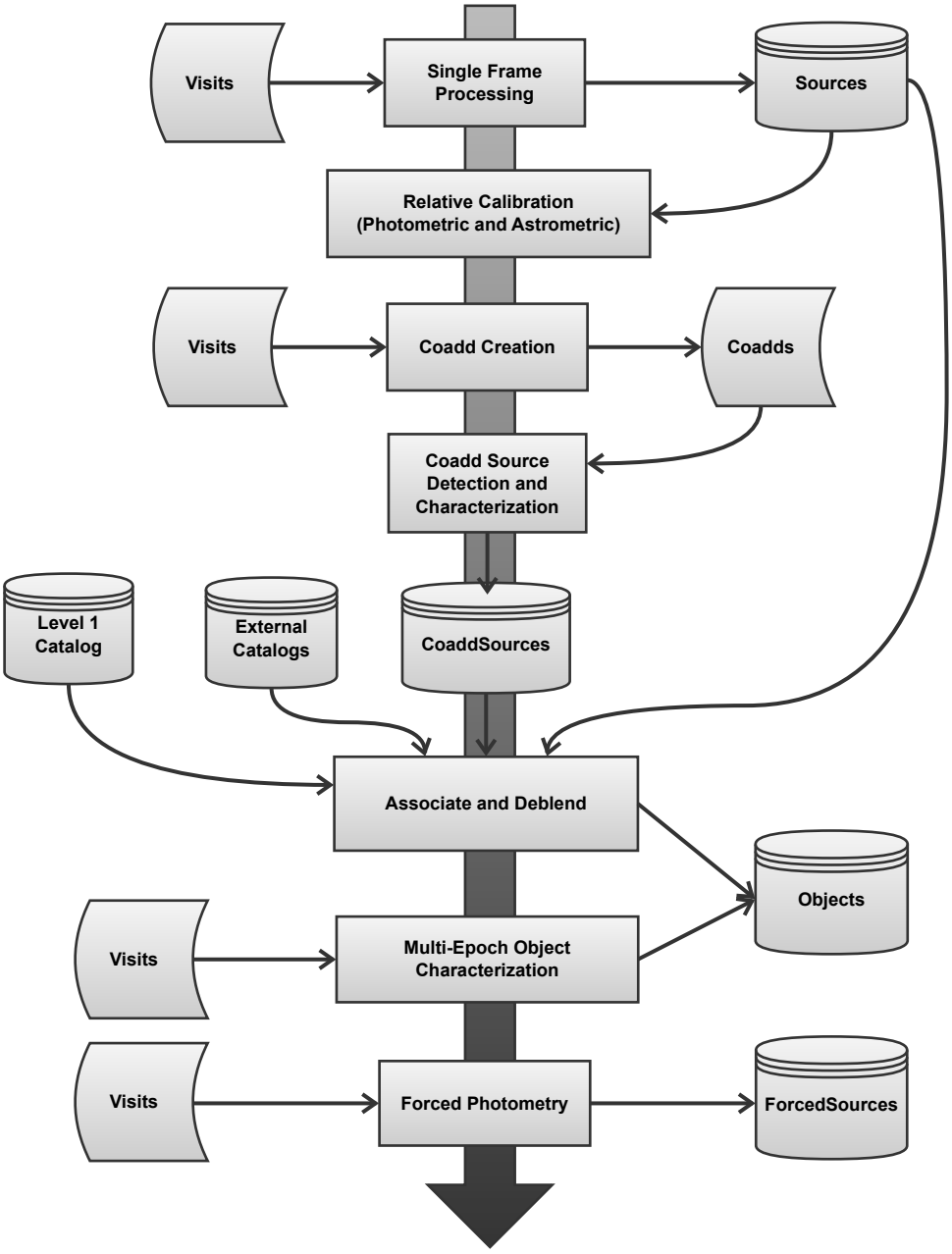

Figure 4: Level 2 Processing Overview

coadds. This task is performed by the ***Image Coaddition Pipeline*** (WBS 02C.04.04). Transient sources (including Solar System objects, explosive transients, etc), will be rejected from the coadds.

4. *Coadd source detection.* Sources will be detected on all coadds generated in the previous step. The source detection algorithm will detect regions of connected pixels, known as *footprints*, above the nominal S/N threshold in the *PSF-likelihood image* of the visit. Each footprint may have one or more *peaks*, and the collection of these peaks (and their membership in the footprints) are the output of this stage. This information will be stored in a catalog of `CoaddSources`. The detection is performed by the ***Object Detection and Deblending*** system (WBS 02C.04.05).

5. *Coadd source deblending and characterization.* The next stage in the pipeline will decompose the `CoaddSources` into a set of individual astronomical sources which is consistent across all bands, a process known as *deblending*. The deblender may make use of the catalogs of `Sources` and `CoaddSources`, catalogs of `DIASources`, `DIAObjects` and `SSObjects` detected on difference images, and objects from external. The deblended objects will then be characterized by measuring their positions, shapes and fluxes on the coadded images and by fitting galaxy models. This functionality is contained within the ***Object Detection and Deblending*** system (WBS 02C.04.05) and the ***Object Characterization Pipeline*** (WBS 02C.04.06).

6. *Multi-epoch object characterization.* A set of measurements (including predefined classes of model fits) will be performed on each of the `Objects` identified in the previous step, taking all available multi-epoch data into account. Model fits will be performed using *MultiFit*-type algorithms. Rather than coadding a set of images and measuring object characteristics on the coadd, MultiFit simultaneously fits PSF-convolved models to the objects multiple observations. This reduces systematic errors, improves the overall S/N, and allows for fitting of time-dependent quantities degenerate with shape on the coadds (for example, the proper motion). The models we plan to fit will *not* allow for flux variability. Object characterization is a part of the ***Object Characterization Pipeline*** (WBS 02C.04.06).

7. *Forced Photometry.* Source fluxes will be measured on every visit, with the position, motion, structural parameters, and deblending characterized in the previous step kept fixed. This process of *forced photometry*, will result in the characterization of the light-curve for each object in the survey. Forced photometry is functionally a part of the ***Object Characterization Pipeline*** (WBS 02C.04.06).

2.3.3 Enabling Level 3 Pipelines

Level 3 capabilities are envisioned to enable science cases requiring further custom user-specified processing, especially the kind that would greatly benefit from co-location within the LSST Data Access Center. The high-level requirement for Level 3 is established in §3.5 of the LSST SRD.

To enable Level 3 use cases, LSST Data Management pipelines will be designed in a modular fashion to maximize the potential for reusability and synergy between Level 3 and Levels 1 and 2.

For example, a typical Level 3 use case will be to perform a different kind of measurement on objects detected in the course of Level 2 processing. A user will be able to do this by reusing the desired components of Level 2 processing, plugging in (via Python `import` directives in the appropriate configuration file) the modules for their custom measurement, and executing the pipeline. The ***Science Pipeline Toolkit*** (WBS 02C.01.02.03) will provide the necessary components to support user-driven construction and execution of custom pipelines.

2.3.4 Science Data Quality Analysis Pipeline and Toolkit

Science Data Quality Analysis requirements are described in the Data Quality Assurance Plan (LSE-63) document. They will be implemented by the ***SDQA Pipeline*** (WBS 02C.01.02.02; the data collection backend) and the ***SDQA Toolkit*** (WBS 02C.01.02.02; the data analysis front-end).

LSST QA will include four main components, which to some extent reflect the Level 1-3 structure of LSST data products. Level 0 QA is software development related, Level 1 QA relates to nightly operations, Level 2 QA relates to data releases, and Level 3 QA is science based.

- **Level 0 QA** includes the extensive and thorough testing of the DM subsystem during the pre- commissioning phase, as well as the tests

of software improvements during the commissioning and operations phases (regression tests based on pipeline outputs and input truth). A common feature of Level 0 QA is the use of LSST simulations products, or any other dataset where the truth is sufficiently well known (e.g., the use of high-resolution observations from space telescopes to test resolved/unresolved object separation algorithms). The main goal of Level 0 QA is to quantify the software performance against these known expected outputs (e.g., to measure the completeness and false positive rate for an object finder; to measure the impact of blended sources on pipeline outputs; to measure the performance of calibration pipelines and MOPS), and to test for algorithm implementation problems (a.k.a. coding bugs).

- **Level 1 QA** assesses the system status and data quality in real time during commissioning and operations. Its main difference from other observatory, telescope, and camera status reporting tools will be heavy reliance on the massive science imaging data stream (in addition to various telemetry and metadata generated by the subsystems). This level is tasked with nightly reporting of the overall data quality, including the nightly data products (difference images and transient source event stream) and calibration products. Real-time information about observing conditions, such as sky brightness, transparency, seeing, and about the system performance, such as the achieved faint limit, will be delivered by Level 1 QA. Because the actual science data stream will be analyzed, Level 1 QA tools will be in a good position to discover and characterize subtle deterioration in system performance that might not be easily caught by tools employed by the telescope and the camera subsystems for self-reporting purposes.

- **Level 2 QA** assesses the quality of data products scheduled for the Data Releases, and provides quantitative details about data quality for each release (including the co-added image data products, and the properties of astrometrically and photometrically variable objects). This level also performs quality assessment for astrometric and photometric calibration, as well as for derived products, such as photometric redshifts for galaxies and various photometric estimators for stars. Subtle problems with the image processing pipelines and systematic problems with the instrument will be discovered with Level 2 QA.

- **Level 3 QA** quality assessment will be based on science analysis performed by the LSST user community. LSST will not develop Level 3 QA tools, but Level 0-2 visualization and data exploration tools will be made available to the community to form a basis on which Level 3 tools can be built. Common features expected for tools at this level are sensitivity to subtle systematic issues not recognized by Level 2 QA, and feedback about data quality to the project by external teams. It is envisioned that especially useful Level 3 QA tools would be migrated to Level 2 QA.

3 Shared Software Components

3.1 Applications Framework (WBS 02C.03.05, 02C.04.01)

3.1.1 Key Requirements

The ***LSST Applications Framework*** (`afw`) is to provide the basic functionality needed by an image processing system. In particular, it shall provide:

- Classes to represent and manipulate mappings between device and astronomical coordinates.
- Classes to represent and manipulate images and exposures[6]
- Classes to represent and estimate backgrounds on images
- Classes to represent the geometry of the camera
- Base classes to represent and manipulate the point spread function (PSF)
- Routines to perform detection of sources on images, and classes to represent these detections (*"footprints"*)
- Classes to represent astronomical objects
- Classes to represent and manipulate tables of astronomical objects
- Other low-level operations as needed by LSST science pipelines

3.1.2 Baseline design

The baseline design calls for a library of C++ classes and functions, exposed to Python callers as a Python module named `lsst.afw`. The classes required are captured in the UML Domain Model (LDM-133), with an example of the Exposure class shown in Figure 5.

This library will form the basis for all image processing pipelines and algorithms used for LSST so special attention will be paid to performance.

[6] images with associated metadata.

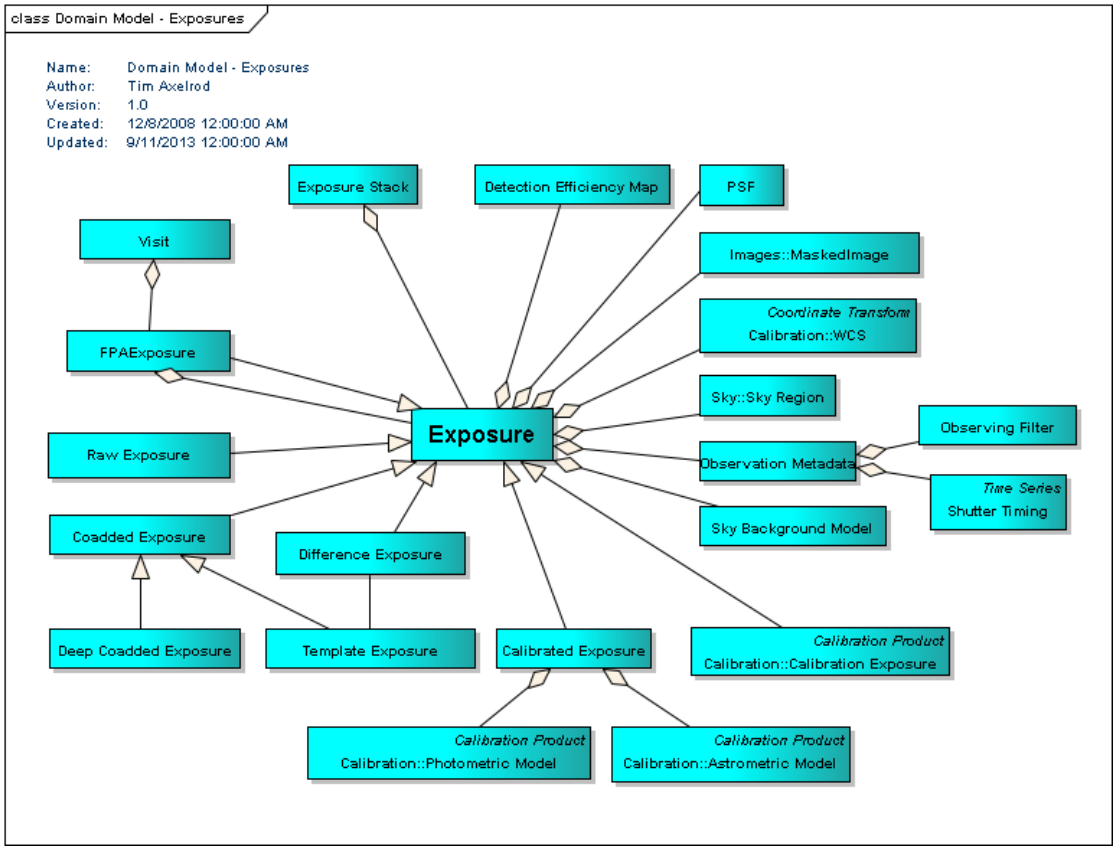

Figure 5: Exposure class diagram

3.1.3 Prototype Implementation

A prototype implementation of the baseline design has been completed in LSST Final Design Phase, including prototype GPU (CUDA) support for major image processing functions (e.g., warping). We expect it will be possible to transfer a significant fraction of the existing code into Construction.

Needs for future improvement have been identified in three areas:

- The current design of classes that represent the Camera geometry is suboptimal. Redesign will be needed in Construction.
- The current design of classes that represent the PSF does not allow for intensity-dependent PSF terms. These will need to be added in construction.
- Design of sky background and geometry classes was found to be insufficiently flexible. This will need to be rectified in Construction.

Prototype code is available at `https://github.com/lsst/afw/`. The documentation for the prototype is located at `http://ls.st/w3o` and `http://ls.st/6i0`.

4 Level 1 Pipelines

4.1 Single Frame Processing Pipeline (WBS 02C.03.01)

4.1.1 Key Requirements

Single Frame Processing (SFM) Pipeline is responsible for reducing raw image data to *calibrated exposures*, and detection and measurement of `Sources` (using the components functionally a part of the Object Characterization Pipeline).

SFM pipeline functions include:

- Assembly of per-amplifier images to an image of the entire CCD.
- Instrumental Signature Removal
- Cosmic ray rejection and snap combining
- Per-CCD Determination of zeropoint and aperture corrections
- Per-CCD PSF determination
- Per-CCD WCS determination and astrometric registration of images
- Per-CCD sky background determination
- Source detection

Calibrated exposure produced by the SFM pipeline must possess all information necessary for measurement of source properties by single-epoch Object Characterization algorithms.

It shall be possible to run this pipeline in two modes: a "fast" mode needed in nightly operations for Level 1 data reductions where no source characterization is done beyond what's required for zero-point, PSF, sky, and WCS determination (image reduction); and a "full" mode that will be run for Level 2 data reductions.

4.1.2 Baseline design

Single Frame Processing pipeline will be implemented as a flexible framework where different data can be easily treated differently, and new processing steps can be added without modifying the stack code.

It will consist of three primary components:

- A library of useful methods that wrap a small number of atomic operations (e.g., `interpolateFromMask`, `overscanCorrection`, `biasCorrection`, etc.)
- A set of classes (`Task`s) that perform higher level jobs (e.g., `AssembleCcdTask`, or `FringeTask`), and a top level class to apply corrections to the input data in the proper order. This top level class can be overridden in the instrument specific `obs_*` packages, making the core SFM pipeline camera agnostic.
- A top-level Task to run the SFM pipeline.

In the paragraphs to follow, we describe the adopted baseline for key SFM algorithms. If not discussed explicitly, the algorithmic baseline for all other functionallity is assumed to be the same as that used by SDSS *Photo* pipeline (Lupton et al.).

4.1.2.1 Instrumental Signature Removal: The adopted pipeline design allows for straightforward addition of correction for instrumental effects that will be discovered in the as-built Camera. The effects currently baselined to be corrected are:

- Bias: A master bias frame, created from a set of overscan corrected zero length exposures, is subtracted from the image to correct for 2D structure in the bias level. For each exposure, overscan columns will be averaged and fit with a 1D function and subtracted row by row to account for time variable bias level.
- Assembly: CCDs will be assembled by trimming the prescan and/or overscan rows and columns from amplifier images and storing them into a Image object.

- Dark current: A master dark frame created from a set of bias corrected exposures taken with the shutter closed is scaled to the science image exposure time and subtracted to correct for 2D structure in the dark current.
- Cross-talk: Cross talk is generated by interaction of fields produced by the current in physically proximate electronic components. This results in bright features from one amp showing up in other. Correction is to subtract each aggressor amp (possibly flipped) modulated by a measured coefficient from the victim amp. The implementation may assume the cross-talk is small enough to be correctable by first order correction only.
- Non-linearity: CCDs do not have perfectly linear response. At both almost empty and almost full well the response can become non-linear. Given a measurement of the linearity of the CCD response, along with any temperature dependence, the data values will be corrected to linear response by simple mapping and interpolation.
- Flat-field: The correction is a division by the normalized master flat. The master flat will be generated assuming a nominal flat spectrum for all sources. Photometric corrections will be applied downstream on a source by source basis given an SED for each source.
- Fringing: Fringe parerns are an interference effect that result from the sharp emission lines in the night sky spectrum. This effect is the strongest in redder bands. The best fit modeled fringe pattern, constructed from monochromatic flats assuming a low-dimensional parametrization of the night sky emission spectrum, will be subtracted from the image.
- Cosmic ray rejection and snap combining: Exposures will be taken in pairs separated by the readout time. The two images and the expected statistics on those images are used to reject pixels that are significant outliers. Once cosmic rays are flagged the two snaps will be added to produce an image with a longer effective exposure.

4.1.2.2 PSF determination: We will run separate algorithms to select candidate stars and determine the point-spread function (PSF, the light distribution for a point source, a critical ingredient to understanding the data

and measuring accurate fluxes and shapes). Both the star selector and PSF determiner algorithms will be pluggable Python modules, so that different algorithms can be run as desired for different analysis needs.

Three selectors will be implemented. The "second-moment" star selector will builds a histogram of the X and Y second moments of flux, search for a peak, and select sources in the peak as point source candidates. The "catalog" star selector, in contrast, will make use of an external catalog of point sources and use astrometric matching to select point source candidates. The "objectSize" star selector will identify point source candidates from the cluster of sources with similar sizes regardless of magnitude. When selecting point source candidates by size (i.e., for the "second-moment" and "objectSize" algorithms), the sizes will be corrected by the known optical distortion of the camera.

Given the irregularly sampled grid of PSFs represented by selected stars, the variation of the PSF across the CCD will be determined. The baselined "principal-components" PSF (pcaPsf) determiner performs a singular value decomposition (also known as a principal components analysis, or PCA) on the point-source candidates in pixel space to produce a set of eigen-images. Using the dominant eigen-images, it constructs polynomial interpolants for their relative weights. This produces a spatially-varying PSF model that captures the most important changes in the PSF over the CCD.

These algorithms are intended to be sufficient to enable Level 1 processing. More advanced PSF determination algorithms will be developed in the PSF Estimation Pipeline (WBS 02C.04.03).

4.1.2.3 Sky Background Determination: We will estimate the smooth sky background by measuring the background level in cells (typically 256 or 512 pixels square) using (by default) a clipped mean, and ignoring pixels that are part of detected sources. An interpolating spline (an Akima spline, by default) will be constructed to estimate the background level in each pixel. Backgrounds will be possible to estimate simultaneously over multiple sensors, including the full focal plane.

Background models will be saved, for later subtraction or restoration (e.g., in background matching, as implemented by the Coaddition Pipeline, WBS 02C.04.04).

4.1.2.4 WCS determination and image registration The absolute World Coordinate System (WCS) will be determined using an *astrometry.net* type algorithm (Lang et al. 2009), seeded with the approximate position of the boresight.

This module will also include the capability to perform relative registration of a set of images to enable coaddition and image differencing, using the *meas_mosaic* registration algorithm of Furusawa et al. (2013) as the baseline.

4.1.3 Constituent Use Cases and Diagrams

Assemble CCD; Determine Aperture Correction; Determine PSF; Remove Instrument Signature; Detect Sources; Determine Photometric Zeropoint; Measure Single Visit Sources; Determine WCS; Sum Exposures, Combine Raw Exposures, Remove Exposure Artifacts; Determine Sky Background Model; Calibrate Exposure; Process Raw Exposures to Calibrated Exposure; Perform Single Visit Processing;

4.1.4 Prototype Implementation

A prototype implementation of all major components of SFM baseline design has been completed in LSST Final Design Phase. The achieved accuracy is comparable to state-of-the art codes today (e.g., SDSS, SExtractor). We expect it will be possible to transfer a significant fraction of the existing code into Construction, with continued improvement to meet LSST accuracy requirements.

WCS determination and image registration modules are an exception, and will require extensive redesign and rewrite. The sky determination module will have to be enhanced to support multi-CCD fitting capability.

The prototype codes are available in the following repositories: `https://github.com/lsst/ip_isr`, `https://github.com/lsst/meas_algorithms`, `https://github.com/lsst/meas_astrom`, `https://github.com/lsst-dm/legacy-meas_mosaic`, `https://github.com/lsst/pipe_tasks`.

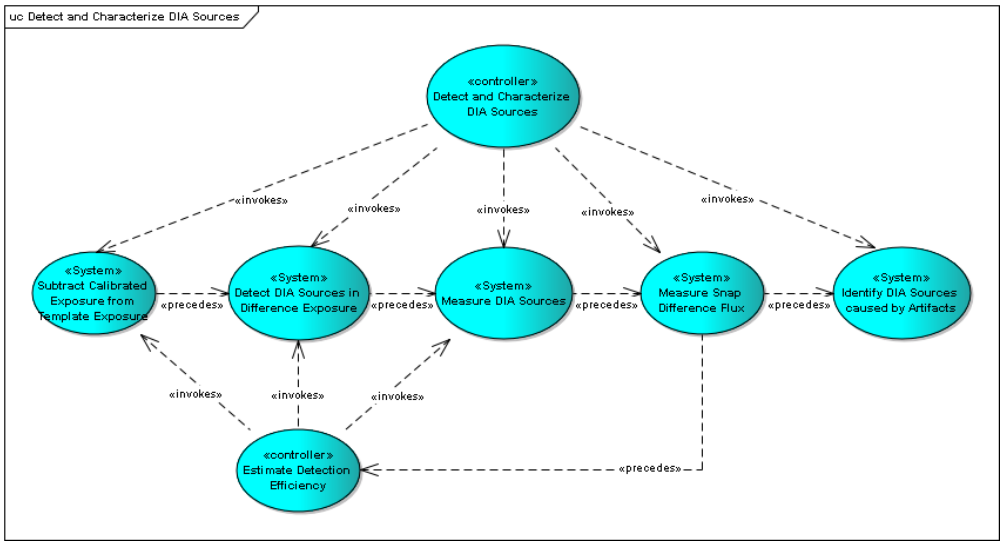

Figure 6: Image Differencing Pipeline Use Case Diagram

4.2 Image Differencing Pipeline (WBS 02C.03.04)

4.2.1 Key Requirements

The image differencing pipeline shall difference a visit image against a deeper template, and detect and characterize sources in the difference image in the time required to achieve the 60 second design goal for Level 1 alert processing (current timing allocation: 24 seconds). The algorithms employed by the pipeline shall result in purity and completeness of the sample as required by the DMSR. Image differencing shall perform as well in crowded as in uncrowded fields.

4.2.2 Baseline design

The Image Differencing pipeline will difference, detect, and deblend objects in the resulting image using the "preconvolution" algorithm as described in Becker et al. (`http://ls.st/x9f`). Differencing will be performed against a deeper template, and differential chromatic refraction (DCR) will be handled by having templates in several bins of airmass.

All `DIASource` measurements described in the DPDD, including post-processing such as variability characterization, will be performed for all sources detected in this manner. The measurements will be performed on the pre-

convolved likelihood image. The measurement code will reside in a separate module in the Object Characterization Pipeline (WBS 02C.04.06).

If necessary a *spuriousness metric* using machine-learning techniques (e.g., Bloom et al. 2010) will be developed to help in the discrimination of real sources from those caused by artifacts.

Details of this baseline design have been captured in the `Detect and Characterize DIA Sources` and related diagrams, presented in Figure 6.

4.2.3 Constituent Use Cases and Diagrams

Subtract Calibrated Exposure from Template Exposure; Identify DIA Sources caused by Artifacts; Perform Precovery Forced Photometry; Measure DIA Sources; Detect DIA Sources in Difference Exposure; Measure Snap Difference Flux; Perform Difference Image Forced Photometry; Calculate DIA Object Flux Variability Metrics; Fit DIA Object Position and Motion;

4.2.4 Prototype Implementation

A prototype implementation partially implementing the baseline design has been completed in the LSST Final Design Phase. It includes detection, centroiding, aperture and PSF photometry, and adaptive shape measurement. This implementation was used to benchmark the speed of the image differencing code and examine the expected levels of false positives. Deblending on difference images, fits to trailed sources, and dipole fits were not prototyped. The final report on prototype design and performance can be found in Becker et al. (`http://ls.st/x9f`).

The prototype code is available at `https://github.com/lsst/ip_diffim`. The current prototype, while functional, will require a partial redesign to be transfered to construction to address performance and extensibility concerns.

4.3 Association Pipeline (WBS 02C.03.02)

4.3.1 Key Requirements

The Association Pipeline has two key responsibilities: i) it must be able to associate newly discovered `DIASources` with previously known `DIAObjects` and `SSObjects`, and ii) it must be able to associate `DIAObjects` with known `Objects` from the Level 2 catalogs.

4.3.2 Baseline design

The baseline design for `DIASources` to `DIAObject` association and `DIAObject` to `Object` association is to use simple nearest-neighbor search while taking proper motions and positional errors ellipses into account.

For matches to `SSObjects`, the `SSObject`'s ephemeris are to be computed by NightMOPS (functionally a part of the Moving Object Pipeline, WBS 02C.03.06). Matching to the computed ephemeris is to be performed as if they were `DIAObjects`.

When Level 1 data is reprocessed, a more sophisticated clustering algorithm (OPTICS; Ankerst et al. 1999) will be employed.

4.3.3 Constituent Use Cases and Diagrams

Create Instance Catalog for Visit; Associate with Instance Catalog; Perform DIA Object Association; Perform DIA Source Association;

4.3.4 Prototype Implementation

Prototype implementation of the baseline design has been completed in LSST Final Design Phase. The nearest-neighbor matching has been implemented as a part of the Application Framework, while clustering using OPTICS resides in the database-related ingest modules.

The prototype code is available at `https://github.com/lsst/ap`. The current prototype, while functional, will require a partial redesign in Construction to address scalability and performance.

4.4 Alert Generation Pipeline (WBS 02C.03.03)

4.4.1 Key Requirements

Alert Generation Pipeline shall take the newly discovered `DIASources` and all associated metadata as described in the DPDD, and generate alert packets in `VOEvent` format. It will transmit these packets to VO Event Brokers, using standard IVOA protocols (eg., VOEvent Transport Protocol; VTP. End-users will primarily use these brokers to classify and filter events for subsets fitting their science goals.

To directly serve the end-users, the Alert Generation Pipeline shall provide a basic, limited capacity, alert filtering service. This service will run at the LSST U.S. Archive Center (at NCSA). It will let astronomers create simple filters that limit what alerts are ultimately forwarded to them. These *user defined filters* will be possible to specify using an SQL-like declarative language, or short snippets of (likely Python) code.

4.4.2 Baseline design

The baseline design is to adopt and upgrade for performance and functionallity the Skyalert package (`http://lib.skyalert.org/skyalert/`).

4.4.3 Constituent Use Cases and Diagrams

Distribute to Subscribed Brokers; Distribute to Subscribed Users; Generate Alerts; Generate and Distribute Alerts;

4.4.4 Prototype Implementation

No prototype implementation has been developed by LSST, as the Skyalert package (`http://lib.skyalert.org/skyalert/`) was found to be mature enough to baseline the architecture and estimate costs.

4.5 Moving Object Pipeline (WBS 02C.03.06)

4.5.1 Key Requirements

The Moving Object Pipeline System (MOPS) has two responsibilities within LSST Data Management:

- First, it is responsible for generating and managing the Solar System[7] data products. These are Solar System objects with associated Keplerian orbits, errors, and detected `DIASources`. Quantitatively, it shall be capable of detecting 95% of all Solar System objects that meet the findability criteria as defined in the OSS. The software components implementing this function are known as ***DayMOPS***.
- The second responsibility of the MOPS is to predict future locations of moving objects in incoming images so that their sources may be associated with known objects; this will reduce the number of spurious transient detections and appropriately flag alerts to detections of known Solar System objects. The software components implementing this function are known as ***NightMOPS***.

4.5.2 Baseline design

The baseline NightMOPS design is to adopt and adapt an existing ephemeris computation pipeline such as OrbFit (Milani et al.) or OpenOrb (Granvik et al.). The baseline DayMOPS design uses Kubica et al. (2005) algorithms to identify and link Solar System object candidates.

The design of these components are explained in detail in the MOPS Design Document (MOPSD).

4.5.3 Constituent Use Cases and Diagrams

Process Moving Objects; Fit Orbit; Prune Moving Object Catalog; Perform Precovery; Recalculate Solar System Object Properties; Link Tracklets into Tracks; Find Tracklets;

[7] Also sometimes referred to as 'Moving Object'

4.5.4 Prototype Implementation

A prototype implementation implementing the key components of DayMOPS baseline design has been completed in LSST Final Design Phase. NightMOPS has not been extensively prototyped, as it is understood not to be an area of significant uncertainty and risk. An extensive report on MOPS prototyping and performance is available as a part of the MOPS Design Document (MOPSD).

Prototype MOPS codes are available at `https://github.com/lsst/mops_daymops` and `https://github.com/lsst/mops_nightmops`. We expect it will be possible to transfer a significant fraction of the existing code into Construction. Current DayMOPS prototype already performs within the computational envelope envisioned for LSST Operations, though it does not yet reach the required completeness requirement.

5 Level 2 Pipelines

5.1 PSF Estimation Pipeline (WBS 02C.04.03)

5.1.1 Key Requirements

PSF Estimation pipeline must enable the estimation of the point spread function with residual ellipticity correlations at the levels required by LSR-REQ-0097.

5.1.2 Baseline design

LSST's PSF models will be implemented as a plugin. Simple PSF estimation algorithms (e.g., `pcaPsf`) will be implemented by the Single Frame Processing pipeline (WBS 02C.03.01). CoaddPsf (`stackfitPsf`) algorithm will be implemented in the Image Coaddition Pipeline (WBS 02C.04.04).

Current state-of-the-art PSF estimation methods typically use some basis functions (e.g., δ-functions, PCA) and a spatial model (e.g., polynomials) to estimate the PSF. This is not expected to be sufficient to reach LSST requirements. We've therefore adopted a baseline algorithm as follows:

- Decompose the PSF into a component due to the atmosphere and a component due to the telescope and camera system.
- Estimate the telescope+camera component using the wavefront sensor information (the reconstructed wavefront) and camera metrology (the laboratory or on-sky measurement of z offsets of individual sensors)
- Estimate the atmospheric contribution by modelling the PSF from bright, isolated, stars, and interpolating using Kriging.

This estimation will be performed on the full focal plane, and use the per-CCD estimates of the PSF and cutouts of bright stars (PSF estimation candidates) as inputs.

The PSF of thick deep depletion devices such as LSST's is known to be intensity dependent. We will estimate the zero-intensity PSF by iteratively forward-modeling the estimated zero-intensity PSF until convergence

is achieved. The same forward-modelling algorithm will allow us to estimate the PSF in undersampled imaging, as is expected for roughly the best quartile of the seeing distribution.

5.1.3 Constituent Use Cases and Diagrams

Perform Full Focal Plane PSF Estimation;

5.1.4 Prototype Implementation

Prototype code for wavefront reconstruction has been developed by the LSST Telescope and Site group. We expect this code will be rewritten in Construction to follow LSST Data Management standards and be straightforward to incorporate into the PSF Estimation pipeline.

The remaining components of the PSF Estimation pipeline have not been prototyped at this time.

5.2 Image Coaddition Pipeline (WBS 02C.04.04)

5.2.1 Key Requirements

Image Coaddition Pipeline shall produce PSF-matched and non-PSF-matched coadds given a stack of input calibrated exposures. Generated coadds must be characterized, at a pixel level, for validity and variance (i.e., have a Mask plane and a Variance plane).

When non-PSF-matched coadds are produced, the effective PSF and its variation across the full coadd must be determined and retained. Outliers appearing in a small subset of input images (e.g., transients or moving objects) must not appear in the final co-adds.

The Image Coaddition Pipeline shall make it possible to generate the coadds with astrophysical backgrounds removed or retained, depending on runtime configuration directives.

The pipeline must be capable of generating coadds in varying geometries and tesselations of the sky, plugged in at runtime via Python modules and configuration directives. To reduce its memory footprint, it shall be capable of generating the coadds in small patches set by the operator at runtime via configuration directives.

For performance reasons, the pipeline shall be able to simultaneously generate multiple coadds from the same input stack, avoiding unnecessary rewarping of inputs to generate each coadd.

5.2.2 Baseline design

5.2.2.1 Coaddition The Coaddition Pipeline will apply the results of the relative astrometric solution (developed as a part of Single Frame Pipeline, WBS 02C.03.01) to the input images, warp (§5.2.2.2) them to a common coordinate system ("sky map"; §5.2.2.4) and coadd the pixels. The warped images will be kept in memory. Using those images, multiple coadds will be created, each consisting of a subset of input images defined explicitly by the operator or algoritmically by quality cuts.

Production of multi-band coadds will be supported by implementing the Szalay, Connolly, Szokoly (1999) algorithm.

Determining the background from individual visits separately is problematic (because different choices can be made in each, especially at the edge of

a CCD; and because extended, faint astrophysical flux is misinterpreted as background), commonly manifesting as dark rings around very bright stars and the suppression of extended flux around galaxies. Therefore, the pipeline will include the capability to perform "background matching" (§5.2.2.3) to produce a coadd with a high signal-to-noise realization of the background in a single reference exposure. This background will be measured over large scales and be possible to subtract with a high degree of accuracy. Background matching will be the default mode of operation. Nevertheless, it will also be possible to produce coadds by subtracting the background first using Sky Background determination and substraction components developed in the Single Frame Processing Pipeline (WBS 02C.03.01).

If the seeing is not constant it becomes impossible to produce a coadd with a PSF varying in a continuous fashion over the field, unless the data is deliberately degraded by convolving the inputs to a common PSF (known as "PSF matching"). In order to enable dealing with the discontinuous PSF, the Coaddition Pipeline will construct a "CoaddPsf" (§5.2.2.5) PSF model, which is a sum of the PSFs of the input images at each point of interest (as proposed as part of 'StackFit' (Jee and Tyson, 2012)).

To accurately measure the colors of galaxies (e.g., for photometric redshifts) in the presence of intra-object color gradients, seeing changes and imperfect galaxy models, the baseline implementation will be capable of producing "PSF-matched coadds" using well known PSF-matching algorithms.

5.2.2.2 Warping To warp an image from the detector frame to the sky frame, a resampling kernel will be used. The kernel is set according to the sub-pixel position on the input image of the centre of the corresponding output pixel. The code will support using Lanczos (of configurable order), bilinear or nearest-neighbour kernels, with the default being a 3rd-order Lanczos (with 10^6 cache realizations), as a compromise between the infinite *sinc* function and the need for speed.

5.2.2.3 Background Matching Background matching will be implemented to enable reaching the maximum depth in the coadds and preserve the astrophysical backgrounds. We adopt as our baseline the algorithm of Huff et al. (2011), extended to two-dimensional data.

The common practice of subtracting the background from each input individually removes features on moderate scales (such as the outer haloes of galaxies, and Galactic cirrus) and can be unstable (especially when the features appear at the edge of a frame) causing increased noise.

Instead, the following algorithm (following Huff et al. 2011) will be implemented: the pipeline will choose one or more reference exposures, and match the backgrounds of each of the other exposures to the reference. This will be done by subtracting the reference from each of the other exposures (which mostly removes astrophysical sources, especially the extended sources which normally contaminate background measurement) and fitting a background model to the difference between the backgrounds[8]. These models will then be subtracted from the other inputs, so that all exposures have the same large-scale background as the reference exposure.

The coadd produced with the above algorithm will retain the extended astrophysical features at high signal-to-noise, and the background can be carefully removed over multiple patches at once. The subtracted image also provides an opportunity to identify sharp features such as optical ghosts, glints and other contaminants that can be masked.

5.2.2.4 Sky Tesselation and Coadd Projections The "skymap" is a tessellation of the sky, providing suitable pre-defined coordinate systems for operations on the sky such as coaddition. The sky map divides the sky into "tracts". For convenience and parallelism, each tract is sub-divided into "patches". Tracts and patches may overlap, so that sources are not lost in the gaps. Tessellations will be pluggable Python modules.

The baseline tessellation is one using a stereographic dodecahedron. The sky will be subdivided into 12 overlapping[9] *tracts*, spanning approximately 75×72 degrees. The sky will be stereographically projected onto the tracts[10], and pixelized into (logical) images 2.0 x 1.9 megapixels in size (3.8 terapixels in all). Physically, these large images will be subdivided into smaller, approximately 2k $\times$ 2k pixel, non-overlapping, *patches*, though that subdivision is

[8]It is anticipated this code will be shared with the Sky Background determination modules from the Single Frame Processing pipeline.

[9]We're planning for 3.5 degrees of overlap, roughly accommodating a full LSST focal plane.

[10]See `https://dev.lsstcorp.org/trac/wiki/DM/SAT/SkyMap` for details.

to be transparent to clients. Clients will be able to request arbitrarily chosen regions in each tract[11], and receive them back as afw `Exposure` objects.

5.2.2.5 CoaddPsf One of the main challenges in producing quality measurements from non-PSF-matched coadds is the complexity of the effective point-spread function on the coadd. Because the PSF has discontinuities at the location of chip boundaries, modeling approaches based on interpolating with smooth functions cannot be used. Instead, when creating a coadd image, we also combine the PSF models of all the input exposures, using an approach similar to that devised by Jee and Tyson (2011). This combination is lazy, in that we simply store the PSF models, their bounding boxes, and the coordinate transforms that relate them to the coadd pixel grid. Then, when measuring an object on the coadd, we obtain the coadd PSF model at that location by coadding the PSF models of all exposures that contributed to the relevant part of the coadd, after warping them by the appropriate coordinate transform. We approximate the PSF as spatially constant in the region of an individual object.

5.2.3 Constituent Use Cases and Diagrams

Create Deep Coadd Exposures; Create Short Period Coadd Exposures; Coadd Calibrated Exposures; Create Best Seeing Coadd Exposures; Create PSF-matched Coadd Exposures; Create Template Exposures;

5.2.4 Prototype Implementation

A prototype implementation of all major components of the Coaddition Pipeline baseline design has been completed in LSST Final Design Phase. We expect it will be possible to transfer a significant fraction of the existing code into Construction, for continued improvement to meet LSST performance and accuracy requirements.

The existing prototype has been extensively tested with image simulation inputs, as well as real data (SDSS Stripe 82). Using Stripe 82 data, it demonstrated the benefits of background matching and CoaddPsf approaches.

[11] Up to some reasonable upper limit, defined by available memory

The design and performance of the current prototype pipeline is described in the Summer 2012 Data Challenge (`http://ls.st/vho`) and Winter 2013 Data Challenge Report (`http://ls.st/ofk`). The prototype codes are available in the `coadd_*` git repositories browsable through the LSST git repository browser at `https://github.com/lsst`.

5.3 Object Detection and Deblending (WBS 02C.04.05)

5.3.1 Key Requirements

The Deep Detection Pipeline is responsible for detection of objects on coadds produced by the Coaddition Pipeline.

5.3.2 Baseline design

The baseline is the SDSS detection algorithm (Lupton et al.).

Object detection is performed by correlating the deep coadd image with the CoaddPsf PSF and searching for peaks above the preset threshold. Multiple adjacent peaks will be detected and merged to reduce spuriously detected objects due to noise or object substructure.

To enable detection of extended objects, the Pipeline will perform detection on recursively binned coadds.

5.3.3 Constituent Use Cases and Diagrams

Detect and Characterize AstroObjects; Detect Sources on Coadds;

5.3.4 Prototype Implementation

A fully functional prototype implementation of the Deep Detection Pipeline baseline design has been completed in LSST Final Design Phase and extensively tested with image simulation inputs as well as real data (SDSS Stripe 82). We expect it will be possible to transfer a significant fraction of the existing code into Construction, for continued improvement to meet LSST performance and accuracy requirements.

The detection functionallity is a part of the `afw` git repository browsable through the LSST git repository browser at `https://github.com/lsst`.

5.4 Object Characterization Pipeline (WBS 02C.04.06)

5.4.1 Key Requirements

Given one or more cutouts of a detected object, observed in one or more epochs, the Object Characterization Pipeline will perform all measurements required by the DPDD, within the computational budget allotted by the LSST sizing model.

Functionally, the components of this pipeline may be incorporated into or invoked by other pipelines.

5.4.2 Baseline design

5.4.2.1 Single-epoch Characterization Single-epoch (including coadd) object characterization will primarily rely on forward-modeling. Models will be convolved with the independently estimated PSF and compared with the pixel data until χ^2 is minimized.

5.4.2.2 Multifit For multi-epoch characterization and some science measurements — particularly shape measurement for weak lensing — measurement on a coadd image may not provide the necessary precision and control of systematics. To perform these measurements, we baseline the "MultiFit" approach (Bosch 2010, Bosch et al. 2013), in which a parametrized model for each astronomical object is fit simultaneously to all of the data in which that object appears. Rather than transform and combine the data, we instead transform the model to the coordinate system of each exposure, convolve it by the appropriate PSF, and compare it to the data from that exposure. Measurements from the coadd will be used as a starting point, so the multifit processing will proceed by iterating over the catalog generated from the coadd, loading postage stamps from the original images, and fitting to these data. Note that the output measurements will generally have the same form as the coadd-based measurements - one set of measurements per astronomical object. There is a single set of the parameters that describe each object, not a different set of parameters for each exposure in which it appears.

Because it does not involve transforming noisy data (which is usually lossy in some sense), and instead transforms analytic models, a multifit approach is theoretically optimal for measurements that can be framed as the results of model-based fits to the data (note that not all measurements can be framed

in such a way, but those that cannot usually do not properly account for the PSF). While it may be possible to construct an optimal coadd that would also be theoretically optimal, we expect that the difficulty in creating such a coadd (i.e., perfectly tracking the covariances between pixels introduced by resampling to a common pixel grid) would be prohibitively complex. On the other hand, for many measurements a non-optimal coadd may be practically sufficient, in the sense that the improvement produced by a multifit-based measurement of the same would be negligible. In these cases, the coadd measurement is likely to be far more efficient computationally. Whenever possible, then, we will do as much work as possible on the coadd first, and only use a multifit approach to "tune up" the final result. And when this final tuning is determined to be unnecessary, it can be skipped entirely.

To further make the multifit approach computationally tractable, we baseline use the multi-shapelet multifit algorithm implementation, as described in Bosch et al. 2013 (`http://ls.st/grd`).

5.4.2.3 Point source model fit To satisfy the point-source model fit requirements, we model each observed object as a point source with finite proper motion and parallax and with constant flux (allowed to be different in each band). This model is a good description for stars and other unresolved sources. Its 11 parameters will be simultaneously constrained using information from all available observations in all bands. The fitting procedure will account for differential chromatic refraction. Multifit will be used to perform the fit.

5.4.2.4 Bulge-disk model fit To satisfy the bulge-disk model fit requirements, the pipeline will be capable of modelling objects as a sum of a de Vaucouleurs (Sersic $n = 4$) and an exponential (Sersic $n = 1$) component. This model is a reasonable description of galaxies. The object is assumed not to move (i.e., have zero proper motion). The components share the same ellipticity and center. The model is independently fit to each band. There are a total of 8 free parameters, which will be simultaneously constrained using information from all available epochs for each band. Where there's insufficient data to constrain the likelihood (eg., small, poorly resolved, galaxies, or very few epochs), the pipeline will have the capability to take into account priors that limit the range of its sampling.

In addition to the maximum likelihood values of fitted parameters and

their covariance matrix, the pipeline shall be capable of sampling independent samples from the likelihood function (or posterior). Multifit will be used to perform the fit.

5.4.2.5 Trailed model fit The pipeline shall be capable of fitting a trailed object model, as described in the DPDD. The baseline algorithm is analogous to that employed by the bulge-disk model fit, but with the model being a line segment instead of a mixture of Sersic profiles.

5.4.2.6 Dipole model fit The pipeline shall be capable of fitting a dipole object model, as described in the DPDD. The baseline algorithm is analogous to that employed by the bulge-disk model fit, but with the model being a mixture of positive and negative point sources, instead of Sersic profiles.

5.4.2.7 Centroids Centroids will be computed independently for each band using an algorithm similar to that employed by SDSS (Lupton et al.). Information from all epochs will be used to derive the estimate. These centroids will be used for adaptive moment, Petrosian, Kron, standard color, and aperture measurements.

5.4.2.8 Adaptive moments Adaptive moments will be computed using information from all epochs, independently for each band, using the algorithm of Bernstein & Jarvis (2002).

5.4.2.9 Petrosian and Kron fluxes Petrosian and Kron radii and fluxes will be measured in standard seeing using self-similar elliptical apertures computed from adaptive moments. The apertures will be PSF-corrected and *homogenized*, convolved to a canonical circular PSF. This is in order to derive a definition of elliptical apertures that does not depend on seeing. For example, for a large galaxy, the correction to standard seeing will introduce little change to measured ellipticity. Corrected apertures for small galaxies will tend to be circular (due to smearing by the PSF). In the intermediate regime, this method results in derived apertures that are relatively seeing-independent. Note that this is only the case for *apertures*; the measured flux will still be seeing dependent and it is up to the user to take this into account.

The radii will be computed independently for each band. Fluxes will be computed in each band, by integrating the light within some multiple of

the radius measured in the canonical band. The shape of the aperture in all bands will be set by the profile of the galaxy in the canonical band alone. This procedure ensures that the color measured by comparing the flux in different bands is measured through a consistent aperture. See `http://www.sdss.org/dr7/algorithms/photometry.html` for details. The pipeline shall be capable of computing radii enclosing 50% and 90% of light.

The baseline for both Petrosian and Kron flux implementation is to derive these as an a mathematical transformation of aperture surface brightness measurements (see below).

5.4.2.10 Aperture surface brightness . Aperture surface brightness will be computed in a variable number, depending on the size of the source, of concentric, logarithmically spaced, PSF-homogenized, elliptical apertures, convolved to standard seeing.

5.4.2.11 Deblender At the depths probed by LSST images, many of the sources are superimposed on each other on the sky ("blended"), which makes detecting and measuring such sources difficult. Often this blending is not severe, but the footprints of reasonably well separated objects can overlap (and therefore merge together) slightly. In other cases, distinct objects will be superimposed directly on more extended objects (e.g., a star on a resolved galaxy).

In order to disentangle the multiple objects, we adopt a baseline for the "deblender" based on the algorithms used in SDSS (Lupton and Ivezic, 2005; `http://adsabs.harvard.edu/abs/2005ASPC..338..151L`).

The baseline deblender assumes that discrete sources generally have twofold rotational symmetry. When one side of a source is blended, we can recover its appearance by examining the symmetric side. The deblender begins by building these symmetric templates for each source. Next, the flux in each pixel is split among the blended source in proportion to their templates. The deblender produces cutouts of each source in a blended group, so that the measurement algorithms (fluxes, galaxy shapes, etc) need not know that the source was blended. This deblending algorithm works well in practice for moderately crowded fields (as demonstrated by SDSS).

This SDSS-style deblender will be run on the set of single-band coadds, then a set of PSF-convolved models will be fit to the deblended the chil-

dren. The initial templates with then be replaced by these models, convolved with the appropriate PSF, and flux assigned as before. The use of physically-motivated templates will help with identification and reduction of non-physical deblends, generalization to multi-epoch data, and data with very different image quality. Fitting these model templates will also allow us to improve centroids of objects whose positions were affected by their neighbors.

In areas of significant stellar crowding (i.e., Galactic plane, star clusters), this approach lends itself to imposition of appropriate template priors (i.e., the correct template being that of a point source convolved with the PSF). This, effectively, makes the deblender into a crowded field code, allowing this baseline to satisfy the requirements for crowded field photometry.

5.4.2.12 Resolved/Unresolved object separation We baseline the resolved/unresolved object separation algorithm based on the ratio of PSF and model fluxes. We use the extendedness criterion as defined by the HSC (Furusawa et al., 2013):

$$\text{extendedness} = (0.95 \times \text{flux.gaussian} < \text{flux.psf})\,?\,0.0 : 1.0$$

5.4.2.13 Variability Characterization Two groups of parameters are required to be provided (see `lcPeriodic` and `lcNonPeriodic` in the DPDD), designed to characterize periodic and aperiodic variability features. We baseline the metrics and algorithms described in Richards et al. (2011) for production of these data products.

5.4.3 Constituent Use Cases and Diagrams

Measure AstroObjects; Exposure Stack Measurements; Create Sky Coverage Maps; Perform Deblending and Association; Perform Forced Photometry; Characterize AstroObject Flux Variability;

5.4.4 Prototype Implementation

A prototype implementation of all major components of the Object Characterization Pipeline baseline design has been completed in LSST Final Design

Phase. The existing prototype has been extensively tested with image simulation inputs, as well as real data (SDSS Stripe 82). We expect it will be possible to transfer a significant fraction of the existing code into Construction, for continued improvement to meet LSST performance and accuracy requirements.

Missing from the current prototypes are the moving point source fit (implemented algorithms assume the source does not move), the trailed source fit, and the aperture, Kron and Petrosian magnitudes using elliptical apertures (implemented algorithms assume the apertures are circular).

The design and performance of the current prototype pipeline is described in the Summer 2012 Data Challenge (`http://ls.st/vho`), Winter 2013 Data Challenge Report (`http://ls.st/ofk`), and Summer 2013 Data Challenge Report (`http://ls.st/grd`). The codes are available in the `meas_*` git repositories browsable through the LSST git repository browser at `https://github.com/lsst`.

6 Calibration Pipelines

6.1 Calibration Products Pipeline (WBS 02C.04.02)

6.1.1 Key Requirements

The Calibration Products Pipeline shall generate the calibration data products as required by the Photometric Calibration Plan (PCP). This includes the telescope/camera calibration products derived from the in-dome apparatus of the LSST observatory, as well as the atmospheric models derived from the spectra taken by the Auxilliary Telescope.

6.1.2 Baseline design

The baseline Calibration Products Pipeline will implement all algorithms described in the Photometric Calibration Plan (PCP). Its logical design is further described in the Calibration Processing package of the UML model (LDM-134; see also Figure 7 for the high-level overview).

6.1.3 Constituent Use Cases and Diagrams

Produce Master Fringe Exposures; Produce Master Bias Exposure; Produce Master Dark Exposure; Calculate System Bandpasses; Calculate Telescope Bandpasses; Construct Defect Map; Produce Crosstalk Correction Matrix; Produce Optical Ghost Catalog; Produce Master Pupil Ghost Exposure; Determine CCOB-derived Illumination Correction; Determine Optical Model-derived Illumination Correction; Create Master Flat-Spectrum Flat; Determine Star Raster Photometry-derived Illumination Correction; Create Master Illumination Correction; Determine Self-calibration Correction-Derived Illumination Correction; Correct Monochromatic Flats; Reduce Spectrum Exposure; Prepare Nightly Flat Exposures;

6.1.4 Prototype Implementation

While parts of the Calibration Products Pipeline have been prototyped by the LSST Calibration Group (see the PCP for discussion), these have not been written using LSST Data Management software framework or coding standards. We therefore expect to transfer the know-how, and rewrite the implementation.

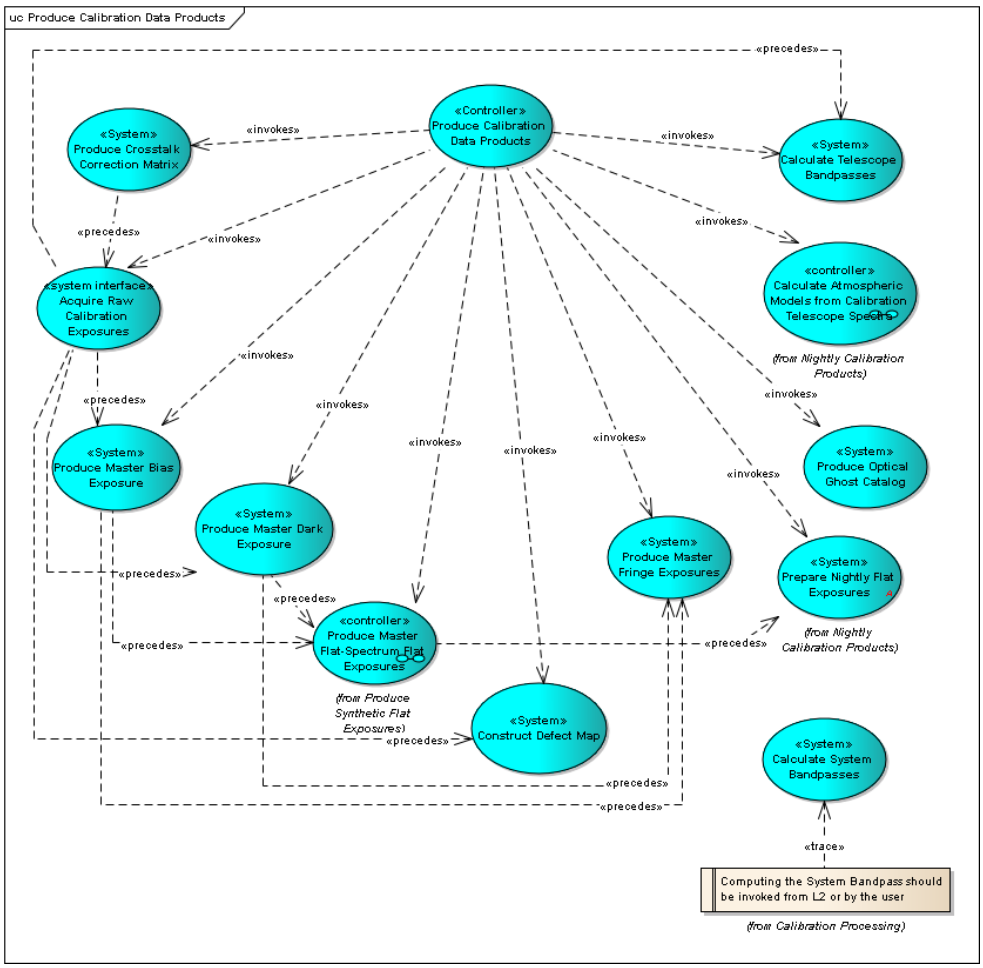

Figure 7: Calibration Products Pipeline Use Case Diagram

6.2 Photometric Calibration Pipeline (WBS 02C.03.07)

6.2.1 Key Requirements

The Photometric Calibration Pipeline is required to internally calibrate the relative photometric zero-points of every observation, enabling the Level 2 catalogs to reach the required SRD precision.

6.2.2 Baseline design

The adopted baseline algorithm is a variant of "ubercal", as described in Padmanabhan et al (2008) and Schlafly et al (2013). This baseline is described in detail in the Photometric Self Calibration Design and Prototype Document (UCAL).

6.2.3 Constituent Use Cases and Diagrams

Perform Global Photometric Calibration;

6.2.4 Prototype Implementation

Photometric Calibration Pipeline has been fully prototyped by the LSST Calibration Group to the required level of accuracy and performance (see the UCAL document for discussion).

As the prototype has not been written using LSST Data Management software framework or coding standards, we assume a non-negligible refactoring and coding effort will be needed to convert it to production code in LSST Construction.

6.3 Astrometric Calibration Pipeline (WBS 02C.03.08)

6.3.1 Key Requirements

The Astrometric Calibration Pipeline is required to calibrate the relative and absolute astrometry of the LSST survey, enabling the Level 2 catalogs to reach the required SRD precision.

6.3.2 Baseline design

Algorithms developed for the Photometric Calibration Pipeline (WBS 02C.03.07) will be repurposed for astrometric calibration by changing the relevant functions to minimize. This pipeline will further be aided by WCS and local astrometric registration modules developed as a component of the Single Frame Processing pipeline (WBS 02C.03.01).

Gaia standard stars will be used to fix the global astrometric system. It is likely that the existence of Gaia catalogs may make a separate Astrometric Calibration Pipeline unnecessary.

6.3.3 Constituent Use Cases and Diagrams

Perform Global Astrometric Calibration;

6.3.4 Prototype Implementation

The Astrometric Calibration Pipeline has been partially prototyped by the LSST Calibration Group, but outside of LSST Data Management software framework. We expect to transfer the know-how, and rewrite the implementation.

7 Level 3 Pipelines

7.1 Science Pipeline Toolkit (WBS 02C.01.02.03)

7.1.1 Key Requirements

The Science Pipeline Toolkit shall provide the software components, services, and documentation required to construct Level 3 science pipelines out of components built for Level 1 and 2 pipelines. These pipelines shall be executable on LSST computing resources or elsewhere.

7.1.2 Baseline design

The baseline design assumes that Level 3 pipelines will use the same `Tasks` infrastructure (see the Data Management Middleware Design document; DMMD) as Level 1 and 2 pipelines[12]. Therefore, Level 3 pipelines will largely be automatically constructible as a byproduct of the overall design.

The additional features unique to Level 3 involve the services to upload/download data to/from the LSST Data Access Center. The baseline for these is to build them on community standards (VOSpace).

7.1.3 Constituent Use Cases and Diagrams

Configure Pipeline Execution; Execute Pipeline; Incorporate User Code into Pipeline; Monitor Pipeline Execution; Science Pipeline Toolkit; Select Data to be Processed; Select Data to be Stored;

7.1.4 Prototype Implementation

While no explicit prototype implementation exists at this time, the majority of LSST pipeline prototypes have successfully been designed in modular and portable fashion. This has allowed a diverse set of users to customize and run the pipelines on platforms ranging from OS X laptops, to 10,000+ core clusters (e.g., BlueWaters), and to implement plugin algorithms (e.g., Kron photometry).

[12] Another way of looking at this is that, functionally, there will be no fundamental difference between Level 2 and 3 pipelines, except for the level of privileges and access to software or hardware resources.

8 Science Data Quality Analysis Pipelines

8.1 SDQA Pipeline (WBS 02C.01.02.02)

8.1.1 Key Requirements

SDQA Pipeline shall provide low-level data collection functionality for science data quality analysis of Level 1, 2, and Calibration Processing pipelines.

In addition, SDQA Pipeline shall provide low-level data collection functionality to support software development in Construction and Operations.

8.1.2 Baseline design

SDQA Pipeline implementation will monitor and harvest the outputs and logs of execution of other science pipelines, computing user-defined metrics.

The metrics will be defined by extending appropriate SDQA Pipeline base classes, and configuring them in the SDQA Pipeline configuration file and/or on the command line.

The outputs of SDQA Pipeline runs will be stored into a SDQA repository (RDBMS or filesystem based).

8.1.3 Constituent Use Cases and Diagrams

Assess Data Quality for Nightly Processing; Assess Data Quality for Calibration Products; Assess Data Quality for Data Release; Assess Data Quality for Nightly Processing at Archive;

8.1.4 Prototype Implementation

Prototype implementation of the SDQA Pipeline baseline design has been completed in LSST Final Design Phase. The existing prototype has been extensively tested with image simulation inputs, as well as real data (SDSS Stripe 82). The existing prototype will be refactored to enhance performance and flexibility in Construction.

The prototype code is available in the `https://github.com/lsst/testing_pipeQA` git repository.

8.2 SDQA Toolkit (WBS 02C.01.02.02)

8.2.1 Key Requirements

SDQA Toolkit shall.provides the visualization, analysis and monitoring capabilities for science quality data analysis. Its inputs will be provided by the SDQA Pipeline.

The toolkit capabilities shall be made flexible, to provide the analyst with the ability to easily construct custom tests and analyses, and "drill down" into various aspects of the data being analyzed.

The toolkit will enable automation of tests and monitoring, and issuance of warnings when alerting thresholds are met.

8.2.2 Baseline design

The core of the toolkit will be designed around a pluggable Python framework generating, a web-based, interactive, visualization interface. This framework has already been implemented in the final design phase as "pipeQA".

Standard data visualization *aspects* will be realized with predefined set of web pages with tests/analyses executed on a given SDQA repository. These aspects will allow the browsing and drill-down of data collected in Data Release Production runs, or monitoring of live data (for Level 1).

Data analysts and users will be able to create new QA tests to examine particular anomalies discovered in the data. It will be possible to add these tests to the library of predefined aspects, to be executed or monitored in Level 1 and Level 2 productions.

8.2.3 Constituent Use Cases and Diagrams

Analyze SDQA Metrics; Correlate SDQA metric with other data; Correlate SDQA metrics; Display SDQA Metrics;

8.2.4 Prototype Implementation

Prototype implementation of the SDQA Toolkit has been implemented in LSST Final Design Phase. The existing prototype has been extensively tested with image simulation inputs, as well as real data (SDSS Stripe 82).

The existing prototype uses a set of statically and dynamically generated pages (written in php) to display the results of data production runs. While proving invaluable for data analysis, the prototype design was found it to be difficult to extend with new analyst-developed tests. The current baseline has been defined based on this experience and will be implemented in Construction.

The prototype code is available in the `https://github.com/lsst/testing_displayQA` git repository.

9 Glossary

API Applications Programming Interface

DAC Data Access Center

DAQ Data Acquisition

DMS Data Management System

DR Data Release.

EPO Education and Public Outreach

Footprint The set of pixels that contains flux from an object. Footprints of multiple objects may have pixels in common.

FRS Functional Requirements Specification

MOPS Moving Object Pipeline System

OCS Observatory Control System

Production A coordinated set of pipelines

PSF Point Spread Function

RGB Red-Green-Blue image, suitable for color display.

SDS Science Array DAQ Subsystem. The system on the mountain which reads out the data from the camera, buffers it as necessary, and supplies it to data clients, including the DMS.

SDQA Science Data Quality Assessment.

SNR Signal-to-Noise Ratio

SQL Structured Query Language, the common language for querying relational databases.

TBD To Be Determined

Visit A pair of exposures of the same area of the sky taken in immediate succession. A Visit for LSST consists of a 15 second exposure, a 2 second readout time, and a second 15 second exposure.

VO Virtual Observatory

VOEvent A VO standard for disseminating information about transient events.

WCS World Coordinate System. A bidirectional mapping between pixel- and sky-coordinates.

References

[1] **Data Management System Design**, `http://ls.st/LDM-148`, 2013

[2] **Data Management Subsystem Requirements**, `http://ls.st/LSE-61`, 2013

[3] M. J. Jee and J. A. Tyson, **Toward Precision LSST Weak-Lensing Measurement. I. Impacts of Atmospheric Turbulence and Optical Aberration**, PASP 123, 596(2011)

[4] D. Lang, D. Hogg, S. Jester, and H-W Rix, **Measuring the undetectable: Proper motions and parallaxes of very faint sources**, ArXiv e-prints, 0808.4004 (2008)

[5] L. Denneau, J. Kubica, and R. Jedicke, **The Pan-STARRS Moving Object Pipeline**, Astronomical Data Analysis Software and Systems XVI ASP Conference Series, Vol. 376, proceedings of the conference held 15-18 October 2006 in Tucson, Arizona, USA. Edited by Richard A. Shaw, Frank Hill and David J. Bell., p.257

[6] **Science Requirements Document**, `http://ls.st/LPM-17`, 2011

[7] **LSST Science User Interface Conceptual Design**, `http://ls.st/LDM-131`, 2011

[8] J. A. Tyson, et al, **LSST and the Dark Sector: Image Processing Challenges**, Astronomical Data Analysis Software and Systems XVII O5.3, ASP Conference Series, Vol. 394 (2007)