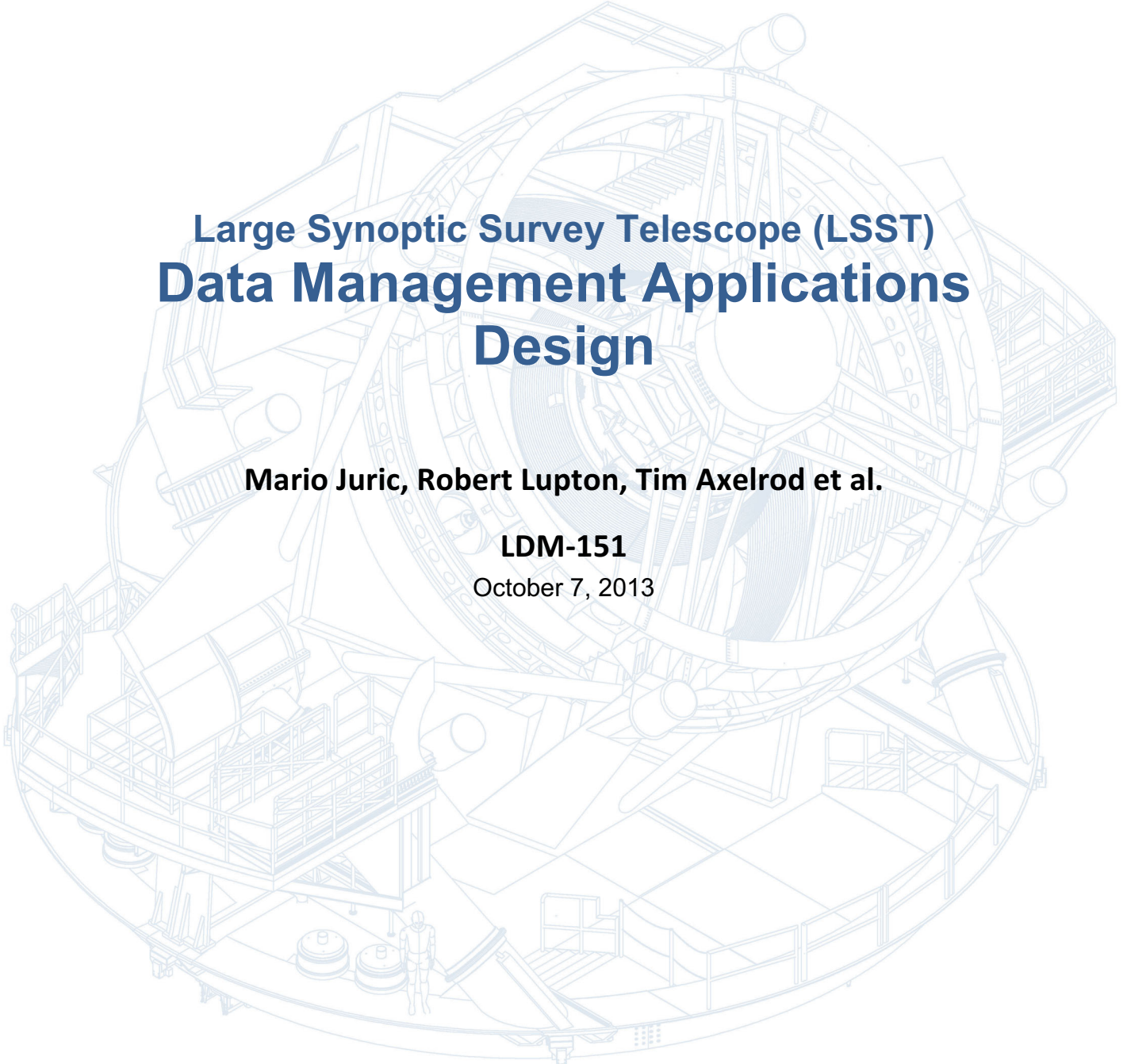




---

LARGE SYNOPTIC SURVEY TELESCOPE

---

A detailed white line-art cutaway diagram of the LSST dome, showing the internal structure, including the telescope's primary mirror, secondary mirror, and various support structures, stairs, and platforms. A small human figure is shown at the bottom for scale.

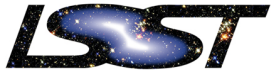
# Large Synoptic Survey Telescope (LSST) Data Management Applications Design

**Mario Juric, Robert Lupton, Tim Axelrod et al.**

**LDM-151**

October 7, 2013





## Change Record

Version	Date	Description	Owner name
1	3/26/2009	Initial version as Document-7396	Tim Axelrod, et. al
1.2	3/27/2009	Minor edits	Tim Axelrod
1.3	4/17/2009	General edits and updates	Tim Axelrod
1.4	5/8/2009	Explicit reference to multifit added to Section 6.1	Tim Axelrod
1.5	2/11/2010	General edits and updates; generated from SysML model	Jeff Kantor
2	8/4/2011	Elevated to LDM handle; general updates and edits	Tim Axelrod
3	10/7/2013	Updates for consistency with FDR baseline	Mario Juric
WIP	2015	Work in progress as part of 2015 replanning exercise	Lim, Bosch, Economou, Krughoff, Swinbank, et al.

# 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*

Wednesday 1<sup>st</sup> June, 2016, 19:18hrs

---

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

## 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 guidelines, 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](#)). The LSST Data Management subsystem's responsibilities include the 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.

# Contents

<b>1</b>	<b>Preface</b>	<b>8</b>
<b>2</b>	<b>Introduction</b>	<b>9</b>
2.1	LSST Data Management System . . . . .	9
2.2	Data Products . . . . .	12
2.3	Data Units . . . . .	13
2.4	Science Pipelines Organization . . . . .	14
<b>3</b>	<b>Level 1 Pipelines</b>	<b>16</b>
3.1	Single Frame Processing Pipeline (WBS 02C.03.01) . . . . .	16
3.1.1	Key Requirements . . . . .	16
3.1.2	Baseline Design . . . . .	17
3.1.2.1	Instrumental Signature Removal: . . . . .	17
3.1.2.2	PSF determination and background determination: . . . . .	18
3.1.2.3	Photometric and Astrometric calibration: . . . . .	18
3.1.3	Prototype Implementation . . . . .	19
3.2	Alert Detection (WBS 02C.03.04) . . . . .	20
3.2.1	Key Requirements . . . . .	20
3.2.2	Baseline Design . . . . .	20
3.2.2.1	Template Generation . . . . .	20
3.2.2.2	Image differencing . . . . .	20
3.2.2.3	Ephemeris Calculation . . . . .	21
3.2.2.4	Source Association . . . . .	21
3.2.3	Prototype Implementation . . . . .	22
3.3	Alert Generation Pipeline (WBS 02C.03.03) . . . . .	23
3.3.1	Key Requirements . . . . .	23
3.3.2	Baseline Design . . . . .	23
3.3.2.1	Alert generation . . . . .	23
3.3.2.2	Alert Distribution . . . . .	23
3.3.2.3	Forced Photometry on all DIAObjects . . . . .	24
3.3.3	Prototype Implementation . . . . .	24
3.4	Precovery Photometry Pipeline . . . . .	25
3.4.1	Key Requirements . . . . .	25
3.4.1.1	Precovery of new DIAObjects . . . . .	25
3.5	Moving Object Pipeline (WBS 02C.03.06) . . . . .	26

3.5.1	Key Requirements . . . . .	26
3.5.2	Baseline Design . . . . .	26
3.5.2.1	Generate Tracklets . . . . .	26
3.5.2.2	Attribution and precovery . . . . .	27
3.5.2.3	Fit Orbits . . . . .	27
3.5.2.4	Association and Precovery: New SSObjects . . . . .	27
3.5.2.5	Merge Orbits . . . . .	28
3.5.3	Prototype Implementation . . . . .	28
<b>4</b>	<b>Calibration Products Production</b>	<b>29</b>
4.1	Calibration Products Pipeline (WBS 02C.04.02) . . . . .	29
4.1.1	Key Requirements . . . . .	29
4.1.2	Baseline Design . . . . .	29
4.1.2.1	Instrumental sensitivity . . . . .	29
4.1.2.2	Atmospheric transmissivity . . . . .	30
4.1.2.3	Detector effects . . . . .	30
4.1.2.4	Ghost catalog . . . . .	32
4.1.3	Constituent Use Cases and Diagrams . . . . .	32
4.1.4	Prototype Implementation . . . . .	33
4.2	Photometric Calibration Pipeline (WBS 02C.03.07) . . . . .	34
4.2.1	Key Requirements . . . . .	34
4.2.2	Baseline Design . . . . .	34
4.2.3	Constituent Use Cases and Diagrams . . . . .	34
4.2.4	Prototype Implementation . . . . .	34
4.3	Astrometric Calibration Pipeline (WBS 02C.03.08) . . . . .	35
4.3.1	Key Requirements . . . . .	35
4.3.2	Baseline Design . . . . .	35
4.3.3	Constituent Use Cases and Diagrams . . . . .	35
4.3.4	Prototype Implementation . . . . .	35
<b>5</b>	<b>Data Release Production</b>	<b>36</b>
5.1	Single Visit Processing and Joint Calibration . . . . .	36
5.1.1	BootstrapImChar . . . . .	37
5.1.2	BootstrapJointCal . . . . .	37
5.1.3	RefineImChar . . . . .	37
5.1.4	RefineJointCal . . . . .	37
5.1.5	FinalImChar . . . . .	37
5.1.6	FinalJointCal . . . . .	37

5.2	Coaddition and Difference Image Processing . . . . .	37
5.2.1	WarpAndPsfMatch . . . . .	37
5.2.2	BackgroundMatchAndReject . . . . .	37
5.2.3	WarpAndPsfCorrelate . . . . .	37
5.2.4	CoaddTemplate . . . . .	37
5.2.5	DiffIm . . . . .	37
5.3	Detection, Association, and Deblending . . . . .	37
5.3.1	DecorrelateCoadds . . . . .	37
5.3.2	ProcessCoadds (Part 3) . . . . .	37
5.4	Object Characterization . . . . .	37
5.4.1	ProcessCoadds (Part 2) . . . . .	38
5.4.2	MultiFit . . . . .	38
5.4.3	ForcedPhotometry . . . . .	38
5.5	Postprocessing . . . . .	38
5.5.1	MOPS . . . . .	38
5.5.2	Classification . . . . .	38
5.5.3	MakeSelectionMaps . . . . .	38
5.5.4	GatherContributed . . . . .	38
<b>6</b>	<b>Science Data Quality Analysis Pipeline</b>	<b>39</b>
6.1	Key Requirements . . . . .	39
6.2	Key Tasks for Each Level of QA . . . . .	39
6.2.1	QA0 . . . . .	40
6.2.2	QA1 . . . . .	41
6.2.3	QA2 . . . . .	42
6.2.4	QA3 . . . . .	43
6.2.5	Prototype Implementation of PipeQA . . . . .	43
<b>7</b>	<b>Science User Interface and Toolkit</b>	<b>44</b>
7.1	Science Pipeline Toolkit (WBS 02C.01.02.03) . . . . .	44
7.1.1	Key Requirements . . . . .	44
7.1.2	Baseline Design . . . . .	44
7.1.3	Constituent Use Cases and Diagrams . . . . .	44
7.1.4	Prototype Implementation . . . . .	44
<b>8</b>	<b>Algorithmic Components</b>	<b>45</b>
8.1	Instrument Signature Removal . . . . .	45
8.1.1	AP: just skip some steps? . . . . .	45

8.1.2	DRP: do all the steps . . . . .	45
8.2	Artifact Detection . . . . .	45
8.2.1	Single-Exposure Morphology . . . . .	45
8.2.2	Snap Subtraction . . . . .	46
8.2.3	Warped Image Comparison . . . . .	46
8.3	Artifact Masking/Interpolation . . . . .	46
8.4	Source Detection . . . . .	46
8.5	Deblending . . . . .	46
8.5.1	Single Visit Deblending . . . . .	47
8.5.2	Multi-Coadd Deblending . . . . .	47
8.6	Measurement . . . . .	47
8.6.1	Variants . . . . .	47
8.6.2	Algorithms . . . . .	47
8.6.3	Blended Measurement . . . . .	48
8.7	Background Estimation . . . . .	48
8.8	Build Background Reference . . . . .	49
8.9	PSF Estimation . . . . .	49
8.9.1	Single CCD PSF Estimation . . . . .	49
8.9.2	Full Visit PSF Estimation . . . . .	49
8.10	Aperture Correction . . . . .	49
8.11	Astrometric Solutions . . . . .	50
8.11.1	Single CCD (for AP) . . . . .	50
8.11.2	Single Visit . . . . .	50
8.11.3	Joint Multi-Visit . . . . .	50
8.12	Photometric Solutions . . . . .	50
8.12.1	Single CCD (for AP) . . . . .	50
8.12.2	Single Visit . . . . .	50
8.12.3	Joint Multi-Visit . . . . .	50
8.13	Generate Diffim Template for a Visit . . . . .	51
8.14	PSF Matching . . . . .	51
8.14.1	Image Subtraction . . . . .	51
8.14.2	PSF Homogenization for Coaddition . . . . .	51
8.15	Image Warping . . . . .	51
8.15.1	Oversampled Images . . . . .	51
8.15.2	Undersampled Images . . . . .	51
8.15.3	Irregularly-Sampled Images . . . . .	51
8.16	Image Coaddition . . . . .	52
8.17	DCR-Corrected Template Generation . . . . .	52



8.18	Image Decorrelation . . . . .	52
8.18.1	Difference Image Decorrelation . . . . .	52
8.18.2	Coadd Decorrelation . . . . .	52
8.19	Star/Galaxy Classification . . . . .	53
8.19.1	Single Visit S/G, Pre-PSF . . . . .	53
8.19.2	Single Visit S/G, Post-PSF . . . . .	53
8.19.3	Multi-Source S/G . . . . .	53
8.19.4	Object Classification . . . . .	53
8.20	Variability Classification . . . . .	53
8.20.1	Using forced photometry . . . . .	53
8.20.2	Using DIASources . . . . .	53
8.21	Proper Motion and Parallax from DIASources . . . . .	53
8.22	Association and Matching . . . . .	53
8.22.1	Single CCD to Reference Catalog, Semi-Blind . . . . .	53
8.22.2	Single Visit to Reference Catalog, Semi-Blind . . . . .	54
8.22.3	Multiple Visits to Reference Catalog . . . . .	54
8.22.4	DIAObject Generation . . . . .	54
8.22.5	Object Generation . . . . .	54
8.22.6	Cross-Patch Merging . . . . .	55
8.22.7	Cross-Tract Merging . . . . .	55
8.23	Ephemeris Calculation . . . . .	55
8.24	Make Tracklets . . . . .	55
8.25	Attribution and precovery . . . . .	55
8.26	Orbit Fitting . . . . .	56
<b>9</b>	<b>Software Primitives</b>	<b>57</b>
9.1	Applications Framework (WBS 02C.03.05, 02C.04.01) . . . . .	57
9.1.1	Key Requirements . . . . .	57
9.1.2	Baseline Design . . . . .	58
9.1.3	Prototype Implementation . . . . .	58
<b>10</b>	<b>Glossary</b>	<b>59</b>

# 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<sup>1</sup>, 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.

---

<sup>1</sup>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 interfaces that utilize, 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](#)).

02C.01.02 SDQA System		
02C.05 Science User Interface and Analysis Tools	02C.03, 02C.04 Alert, Calibration, Data Release Productions	
02C.06.01 Science Data Archive (Images, Alerts, Catalogs)	Algorithmic Components	
02C.03.05, 02C.04.01 Shared Software Primitives		
02C.06.02 Data Access Services		
02C.07.01, 02C.06.03 Processing Middleware		
02C.07.02 Infrastructure Services (System Administration, Operations, Security)		
02C.07.04.01 Archive Site	02C.07.04.02 Base Site	02C.08.03 Long-Haul Communications
Physical Plant (included in above)		

Data Management System Design LDM-148

**Application Layer (LDM-151)**

- Scientific Layer
- Pipelines constructed from reusable Algorithmic Components
- Data Products represented by Shared Software Primitives
- Object-oriented, python, C++ Custom Software

**Middleware Layer (LDM-152)**

- Portability to clusters, grid, other
- Provide standard services so applications behave consistently (e.g. provenance)
- Preserve performance (<1% overhead)
- Custom Software on top of Open Source, Off-the-shelf Software

**Infrastructure Layer (LDM-129)**

- Distributed Platform
- Different sites specialized for real-time alerting vs peta-scale data access
- Off-the-shelf, Commercial Hardware & Software, Custom Integration

Figure 1: Architecture of the Data Management System

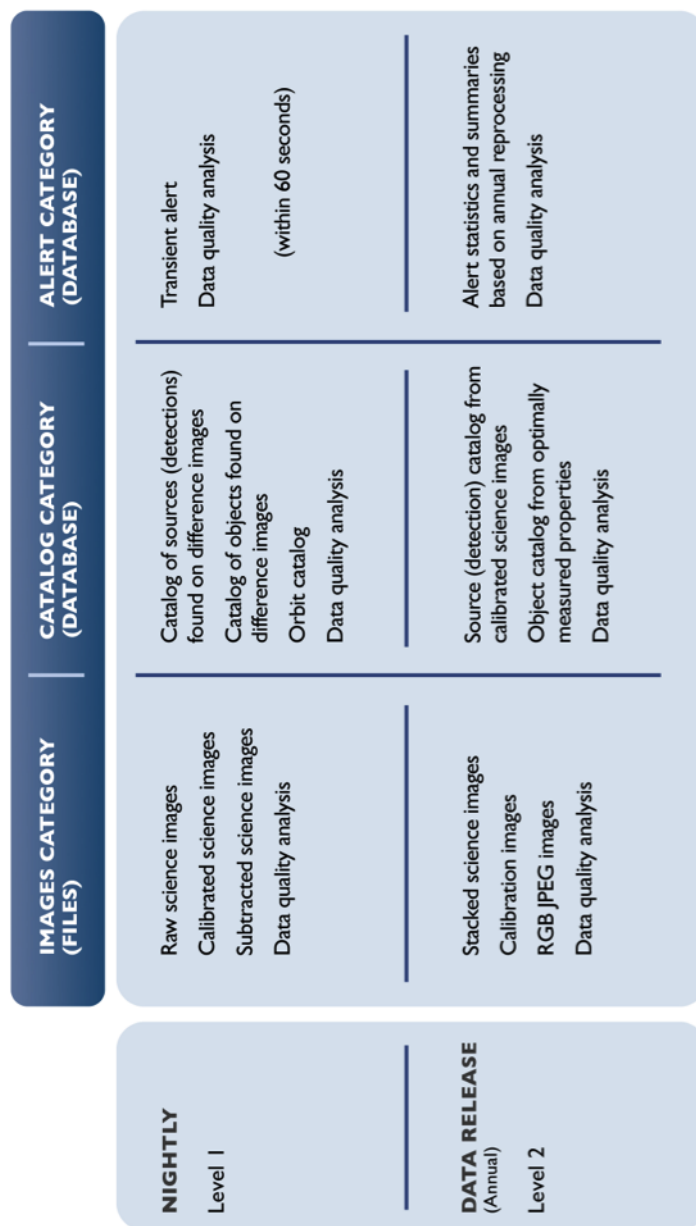


Figure 2: Organization of LSST Data Products

## 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”<sup>2</sup> 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 are generated on any computing resources anywhere and then stored in an LSST Data Access Center. Often, but not necessarily, they will be generated by 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 products. 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.

---

<sup>2</sup>Science Data Quality Analysis

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

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.

The pipelines used to produce these public data products will also produce many intermediate data products that may not be made publically available (generally because they are fully superseded in quality by a public data product). Intermediate products may be important for QA, however, and their specification is an important part of describing the pipelines themselves.

### 2.3 Data Units

In order to describe the components of our processing pipelines, we first need standard nomenclature for the units of data the pipeline will process.

The smallest data units are those corresponding to individual astrophysical entities. In keeping with LSST conventions, we use “object” to refer to the astrophysical entity itself (which typically implies aggregation of some sort over all exposures), and “source” to refer to the realization of an object on a particular exposure. In the case of blending, of course, these are just our best attempts to define distinct astrophysical objects, and hence it is also useful to define terms that represent this process. We use “family” to refer to group of blended objects (or, more rarely, sources), and “child” to refer to a particular deblended object within a family. A “parent” is also created for each family, representing the alternate hypothesis that the blend is actually a single object. Blends may be hierarchical; a child at one level may be a parent at the level below.

LSST observations are taken as a pair of 15-second “snaps”; together these constitute a “visit”. Because snaps are typically combined early in the processing (and some special programs and survey modes may take only a single snap), visit is much more frequently used as a unit for processing and data products. The image data for to a visit is a set of 189 “CCD” or “sensor” images. CCD-level data from the camera is further data divided across the 16 amplifiers within a CCD, but these are also combined at an early stage, and the  $3\times 3$  CCD “rafts” that play an important role in the hardware design are relatively unimportant for the pipeline. This leaves visit and CCD the main identifiers of most exposure-level data products and pipelines.

Our convention for defining regions on the sky is deliberately vague; we

hope to build a codebase capable of working with virtually any pixelization or projection scheme (though different schemes may have different performance or storage implications). Our approach involves two region concepts: “tracts” and “patches”. A tract is a large region with a single Cartesian coordinate system; we assume it is larger than the LSST field of view, but its maximum size is essentially set by the point at which distortion in the projection becomes significant enough to affect the processing (by e.g. breaking the assumption that the PSF is well-sampled on the pixel grid). Tracts are divided into patches, all of which share the tract coordinate system. Most image processing is performed at the patch level, and hence patch sizes are chosen largely to ensure that patch-level data products and processing fit in memory. Both tracts and patches are defined such that each region overlaps with its neighbors, and these overlap regions must be large enough that any individual astronomical object is wholly contained in at least one tract and patch. In a patch overlap region, we expect pixel values to be numerically equivalent (i.e. equal up to floating point round-off errors) on both sides; in tract overlaps, this is impossible, but we expect the results to be scientifically consistent. Selecting larger tracts and patches thus reduces the overall fraction of the area that falls in overlap regions and must be processed multiple times, while increasing the computational load for processing individual tracts and patches.

## 2.4 Science Pipelines Organization

As shown in Figure 1, the Applications Layer is itself split into three levels. In sections 3, 4, and 5, we describe the Alert Production, Calibration Products Production, and Data Release Production (respectively), breaking them down into *pipelines*. In this document, a pipeline is a high-level combination of algorithms that is intrinsically tied to its role in the production in which it is run. For instance, while both Alert Production and Data Release Production will include a pipeline for single-visit processing, these two pipelines are *distinct*, because the details of their design depend very much on the context in which they are run. Section 6 describes the Science Data Quality Analysis System, a collection of pipelines and mini-productions designed to assess and continuously validate the quality of both the data and the processing system. The SDQA System is not a single production; its components are either directly integrated into other productions or part of a set of multiple mini-productions run on different cadences.



Pipelines are largely composed of Algorithmic Components: mid-level algorithmic code that we expect to reuse (possibly with different configuration) across different productions. These components constitute the bulk of the new code and algorithms to be developed for Alert Production and Data Release Production, and are discussed in section 8. Most algorithmic components are applicable to any sort of astronomical imaging data, but some will be customized for LSST.

The lowest level in the Applications Layer is made up of our shared software primitives: libraries that provide important data structures and low-level algorithms, such as images, tables, coordinate transformations, and nonlinear optimizers. Much (but not all) of this content is astronomy-related, but essentially none of it is specific to LSST, and hence we can and will make use of third-party libraries whenever possible. These primitives also play an important role in connecting the Science User Interface Toolkit and Level 3 processing environment with Level 1 and Level 2 data products, as they constitute the programmatic representation of those data products. Shared software primitives are discussed in section 9.

## 3 Level 1 Pipelines

### 3.1 Single Frame Processing Pipeline (WBS 02C.03.01)

#### 3.1.1 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.

### 3.1.2 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 (Tasks) 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 functionality is assumed to be the same as that used by SDSS *Photo* pipeline [15].

Output information for OCS telemetry: ACTION clarify OCS interactions

#### 3.1.2.1 Instrumental Signature Removal: Clarify interaction with butler

**Input Data?**

**Output Data?**

**Anscillary Products?**

**Actions in case of failure?**

**Alternative procedures?**

**Subtasks:**

- Mask defects and saturation
- Assembly

- Full frame corrections: Dark, Flats (includes fringing)
- Pixel level corrections: Brighter fatter, static pixel size effects
- Interpolation of defects and saturation
- CR rejection
- Generate snap difference?
- Snap combination

### **3.1.2.2 PSF determination and background determination:**

**Input Data?**

**Output Data?**

**Anscillary Products?**

**Actions in case of failure?**

**Alternative procedures?**

**Subtasks:**

- Low order background estimation
- Source detection
- Selection of PSF candidate stars
- PSF determination
- Mask detected source
- Re-estimate background and re-detect: iterate to convergence
- Determine final PSF

### **3.1.2.3 Photometric and Astrometric calibration:**

**Input Data?**

**Output Data?**

**Anscillary Products?**

**Actions in case of failure?**

**Alternative procedures?**

**Subtasks:**

- Source measurement (Aperture Corrections)
- Source association
- Calculate zeropoint
- Remove known astrometric distortions
- Fit remaining residual
- Produce composed astrometric solution
- Output information for OCS telemetry: ACTION clarify OCS interactions

## **OUTPUT: Calibrated Exposure and Calibrated Catalog**

### **3.1.3 Prototype Implementation**

The prototype codes are available in the following repositories: [https://github.com/lsst/ip\\_isr](https://github.com/lsst/ip_isr), [https://github.com/lsst/meas\\_algorithms](https://github.com/lsst/meas_algorithms), [https://github.com/lsst/meas\\_astrom](https://github.com/lsst/meas_astrom), [https://github.com/lsst-dm/legacy-meas\\_mosaic](https://github.com/lsst-dm/legacy-meas_mosaic), [https://github.com/lsst/pipe\\_tasks](https://github.com/lsst/pipe_tasks).

## 3.2 Alert Detection (WBS 02C.03.04)

### 3.2.1 Key Requirements

The alert detection 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.

### 3.2.2 Baseline Design

#### 3.2.2.1 Template Generation

Input Data?

Output Data?

Anscillary Products?

Actions in case of failure?

Alternative procedures?

Subtasks:

- Determine appropriate template to use
- Generate template for observation

#### 3.2.2.2 Image differencing

Input Data?

Output Data?

Anscillary Products?

Actions in case of failure?

Alternative procedures?

Subtasks:

- Obtain measurements of coadd sources
- Determine relative astrometric solution
- Warp template and measurements to science image frame

- Obtain science image PSF
- Correlate science image with science PSF
- Determine appropriate PSF matching sources
- Compute PSF matching model
- Difference science and template images
- Apply correction for correlated noise
- Difference image source detection
- Difference image source measurement: dipole fit, trailed source measurement
- Measure flux on snap difference for all DIASources?
- Spuriousness calculation

### 3.2.2.3 Ephemeris Calculation

**Input Data?**

**Output Data?**

**Anscillary Products?**

**Actions in case of failure?**

**Alternative procedures?**

**Subtasks:**

- Calculate positions for all solar system objects that may overlap the current exposure.

### 3.2.2.4 Source Association

**Input Data?**

**Output Data?**

**Anscillary Products?**

**Actions in case of failure?**

**Alternative procedures?**

**Subtasks:**

- Match all DIASources to predicted Solar System object positions and DIAObject catalog positions
- Perform forced photometry of un-associated DIAObjects.(Maybe not if we force photometer all DIAObjects?). SSOjects will not be force photometered because the precision of the prediction will not be good enough. Force photometry for external DIAObjects?
- Update associated DIAObjects with aggregate quantities: e.g. parallax, proper motion, and variability metrics
- New spuriousness calculation?

### **3.2.3 Prototype Implementation**

The prototype code is available at [https://github.com/lsst/ip\\_diffim](https://github.com/lsst/ip_diffim). The current prototype, while functional, will require a partial redesign to be transferred to construction to address performance and extensibility concerns.



### 3.3 Alert Generation Pipeline (WBS 02C.03.03)

#### 3.3.1 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.

#### 3.3.2 Baseline Design

##### 3.3.2.1 Alert generation

Input Data?

Output Data?

Anscillary Products?

Actions in case of failure?

Alternative procedures?

Subtasks:

- Generate postage stamps for all `DIASources`: direct image and difference image
- Push alert records to alert database

##### 3.3.2.2 Alert Distribution

Input Data?

Output Data?

Anscillary Products?

Actions in case of failure?

Alternative procedures?

**Subtasks:**

- Filter event records (for content as well as for events)
- Author VOEvent
- Push to messaging queue

**3.3.2.3 Forced Photometry on all DIAObjects**

**Input Data?**

**Output Data?**

**Anscillary Products?**

**Actions in case of failure?**

**Alternative procedures?**

**Subtasks:**

- Compute forced photometry on all DIAObjects in the field. This does not end up in the alerts.

**3.3.3 Prototype Implementation**

## **3.4 Precovery Photometry Pipeline**

### **3.4.1 Key Requirements**

Within 24 hrs.

#### **3.4.1.1 Precovery of new DIAObjects**

**Input Data?**

**Output Data?**

**Anscillary Products?**

**Actions in case of failure?**

**Alternative procedures?**

**Subtasks:**

- Force photometer in difference images for all new DIAObjects for the past 30 days.

## 3.5 Moving Object Pipeline (WBS 02C.03.06)

### 3.5.1 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<sup>3</sup> 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*.

### 3.5.2 Baseline Design

#### 3.5.2.1 Generate Tracklets

Output Data?

Anscillary Products?

Actions in case of failure?

Alternative procedures?

Subtasks:

- Make all tracklet pairs
- Merge multiple chained observation into single longer tracklets
- Purge any tracklets inconsistent with the merged tracklets

---

<sup>3</sup>Also sometimes referred to as ‘Moving Object’

### **3.5.2.2 Attribution and precovery**

**Output Data?**

**Anscillary Products?**

**Actions in case of failure?**

**Alternative procedures?**

**Subtasks:**

- Predict locations of known Solar System objects
- Match tracklet observation to predicted ephemerides taking into account velocity
- Update SSObjects
- Possibly iterate

### **3.5.2.3 Fit Orbits**

**Output Data?**

**Anscillary Products?**

**Actions in case of failure?**

**Alternative procedures?**

**Subtasks:**

- Merge unassociated tracklets into tracks.
- Fit orbits to all tracks.
- Purge unphysical tracks.
- Update SSObjects
- Possibly iterate

### **3.5.2.4 Association and Precovery: New SSObjects**

**Output Data?**

**Anscillary Products?**

**Actions in case of failure?**

**Alternative procedures?**

**Subtasks:**

- Do association and precovery just for SSOjects just found
- Update SSOjects

### 3.5.2.5 Merge Orbits

**Output Data?**

**Anscillary Products?**

**Actions in case of failure?**

**Alternative procedures?**

#### **Subtasks:**

- Merge orbits with high probability of being the same orbit into a single SSOject

### 3.5.3 Prototype Implementation

Prototype MOPS codes are available at [https://github.com/lsst/mops\\_daymops](https://github.com/lsst/mops_daymops) and [https://github.com/lsst/mops\\_nightmops](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.

## 4 Calibration Products Production

### 4.1 Calibration Products Pipeline (WBS 02C.04.02)

#### 4.1.1 Key Requirements

The work performed in this WBS serves two complementary roles:

- It will enable the production of calibration data products as required by the Level 2 Photometric Calibration Plan ([LSE-180](#)) and other planning documents [17]<sup>4</sup>. This includes both characterization of the sensitivity of the LSST system (optics, filters and detector) and the transmissivity of the atmosphere.
- It will characterize of detector anomalies in such a way that they can be corrected either by the instrument signature removal routines in the Single Frame Processing Pipeline (WBS 02C.03.01) or, if appropriate, elsewhere in the system;
- It will manage and provide a catalog of optical ghosts and glints to other parts of the system upon demand.

#### 4.1.2 Baseline Design

**4.1.2.1 Instrumental sensitivity** We expect laboratory measurements of the filter profiles. We further baseline the development of a procedure for measuring the filter response at 1 nm resolution using the approach described in [17].

We baseline the following procedure for creating flat fields:

1. Record bias/dark frames;
2. Use “monochromatic” (1 nm) flat field screen flats with no filter in the beam to measure the per-pixel sensitivity;
3. Use a collimated beam projector (CBP) to measure the quantum efficiency (QE) at a set of points in the focal plane, dithering those points to tie them together;

---

<sup>4</sup>Resolving contradictions between these documents is out of scope here.

4. Combine the screen and CBP data to determine the broad band (10–100 nm) QE of all pixels;
5. Fold in the filter response to determine the 1 nm resolution effective QE of all pixels.

This WBS is responsible for the development of the data analysis algorithms and software required and the ultimate delivery of the flat fields. Development and commissioning of the CBP itself, together with any other infrastructure required to perform the above procedure, lies outwith Data Management (see 04C.08 *Calibration System*).

**4.1.2.2 Atmospheric transmissivity** Measurements from the auxiliary instrumentation—to include the 1.2 m “Calypso” telescope, a bore-sight mounted radiometer and satellite-based measurement of atmospheric parameters such as pressure and ozone—will be used to determine the atmospheric absorption along the line of sight to standard stars. The atmospheric transmission will be decomposed into a set of basis functions and interpolated in space in time to any position in the LSST focal plane.

This WBS will develop a pipeline for accurate spectrophotometric measurement of stars with the auxiliary telescope. We expect to repurpose and build upon publicly available code e.g. from the PFS<sup>5</sup> project for this purpose.

This WBS will construct the atmospheric model, which may be based either on MODTRAN (as per LSE-180) or a PCA-like decomposition of the data (suggested by [17]).

This WBS will define and develop the routine for fitting the atmospheric model to each exposure from the calibration telescope and providing estimates of the atmospheric transmission at any point in the focal plane upon request.

**4.1.2.3 Detector effects** An initial cross-talk correction matrix will be determined by laboratory measurements on the Camera Calibration Optical Bench (CCOB). However, to account for possible instabilities, this WBS will develop an on-telescope method. We baseline this as being based on measurement with the CBP, but we note the alternative approach based on cosmic rays adopted by HSC [10].

Multiple reflections between the layers of the CCD give rise to spatial variability with fine scale structure in images which may vary with time [17,

---

<sup>5</sup>Subaru’s Prime Focus Spectrograph; <http://sumire.ipmu.jp/pfs/>.



§2.5.1]. These can be characterized by white light flat-fields. Preliminary analysis indicates that these effects may be insignificant in LSST [20]; however, the baseline calls for a routine developed in this WBS to analyse the flat field data and generate fringe frames on demand. This requirement may be relaxed if further analysis (outside the scope of this WBS) demonstrates it to be unnecessary.

This WBS will develop algorithms to characterize and mitigate anomalies due to the nature of the camera’s CCDs.

**Note:**

There’s a complex inter-WBS situation here: the actual mitigation of CCD anomalies will generally be performed in SFM (WBS 02C.03.01), based on products provided by this WBS which, in turn, may rely on laboratory based research which is broadly outside the scope of DM. We baseline the work required to develop the corrective algorithms here. We consider moving it to WBS 02C.03.01 in future.

The effects we anticipate include:

- QE variation between pixels;
- Static non-uniform pixel sizes (e.g. “tree rings” [23]);
- Dynamic electric fields (e.g. “brighter-fatter” [2]);
- Time dependent effects in the camera (e.g. hot pixels, changing cross-talk coefficients);
- Charge transfer (in)efficiency (CTE).

Laboratory work required to understand these effects is outwith the scope of this WBS. In some cases, this work may establish that the impact of the effect may be neglected in LSST. The baseline plan addresses these issues through the following steps:

- Separate QE from pixel size variations<sup>6</sup> and model both as a function of position (and possibly time);

---

<sup>6</sup>Refer to work by Rudman.

- Learn how to account for pixel size variation over the scale of objects (e.g. by redistributing charge);
- Develop a correction for the brighter-fatter effect and develop models for any features which cannot be removed;
- Handle edge/bloom using masking or charge redistribution;
- Track defects (hot pixels);
- Handle CTE, including when interpolating over bleed trails.

**4.1.2.4 Ghost catalog** The Calibration Products Pipeline must provide a catalog of optical ghosts and glints which is available for use in other parts of the system. Detailed characterization of ghosts in the LSST system will only be possible when the system is operational. Our baseline design therefore calls for this system to be prototyped using data from precursor instrumentation; we note that ghosts in e.g. HSC are well known and more significant than are expected in LSST.

**Note:**

It is not currently clear where the responsibility for characterizing ghosts and glints in the system lies. We assume it is outwith this WBS.

### 4.1.3 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;

#### 4.1.4 Prototype Implementation

While parts of the Calibration Products Pipeline have been prototyped by the LSST Calibration Group (see the [LSE-180](#) 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.

## 4.2 Photometric Calibration Pipeline (WBS 02C.03.07)

### 4.2.1 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.

### 4.2.2 Baseline Design

The adopted baseline algorithm is a variant of “ubercal” [19, 22]. This baseline is described in detail in the Photometric Self Calibration Design and Prototype Document ([UCAL](#)).

### 4.2.3 Constituent Use Cases and Diagrams

Perform Global Photometric Calibration;

### 4.2.4 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.

### **4.3 Astrometric Calibration Pipeline (WBS 02C.03.08)**

#### **4.3.1 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.

#### **4.3.2 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.

#### **4.3.3 Constituent Use Cases and Diagrams**

Perform Global Astrometric Calibration;

#### **4.3.4 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.

## 5 Data Release Production

**Overview Diagram:**

Collapse and summarize “DRP Top-Level Overview” on confluence along the lines of this outline, mostly by dropping the “Task/Process” boxes and merging data products with the same data units.

**Data Products Table:**

Table of data products with brief descriptions (including intermediates, so not just DPDD), expanding the groups defined in Overview Diagram

Brief summary of all Pipelines:

- Start with iteration between single-visit processing and joint calibration; iteration necessary to get consistent WCS/PSF, but also helpful to identify stars for PSF modeling.
- Iteratively build coadds and detect differences between images. Improve backgrounds, find artifacts, and define “static sky”, then use this to detect transient/variable/moving astrophysical sources.
- Detect on coadds, associate detections (inc. DIASources), and deblend on coadds.
- Measure objects on coadds and individual epochs.
- Measure selection functions, compute classifications, absorb level-3 contributions we depend on. Highlight uncertainty.

### 5.1 Single Visit Processing and Joint Calibration

**ImChar/JointCal Diagram:**

Extract ImChar/JointCal pipelines from “DRP Top-Level Overview” on confluence and expand detail to show data flow and ordering of “Task/Process” boxes.

### 5.1.1 BootstrapImChar

### 5.1.2 BootstrapJointCal

### 5.1.3 RefineImChar

### 5.1.4 RefineJointCal

### 5.1.5 FinalImChar

### 5.1.6 FinalJointCal

## 5.2 Coaddition and Difference Image Processing

**Coaddition, DiffIm Diagram:**

Extract Coaddition and DiffIm pipelines from “DRP Top-Level Overview” on confluence and expand detail to show data flow and ordering of “Task/Process” boxes.

### 5.2.1 WarpAndPsfMatch

### 5.2.2 BackgroundMatchAndReject

### 5.2.3 WarpAndPsfCorrelate

### 5.2.4 CoaddTemplate

### 5.2.5 DiffIm

## 5.3 Detection, Association, and Deblending

**Detection/Association/Deblending Diagram:**

Extract process\_coadds pipeline from “DRP Top-Level Overview” on confluence and expand detail to show data flow and ordering of “Task/Process” boxes.

### 5.3.1 DecorrelateCoadds

### 5.3.2 ProcessCoadds (Part 3)

## 5.4 Object Characterization

**Object Characterization Diagram:**

Extract multifit/forced\_photometry pipelines from “DRP Top-Level Overview” on confluence and expand detail to show data flow and ordering of “Task/Process” boxes.

#### 5.4.1 ProcessCoadds (Part 2)

#### 5.4.2 MultiFit

#### 5.4.3 ForcedPhotometry

### 5.5 Postprocessing

**Postprocessing Diagram:**

Extract Afterburner pipelines from “DRP Top-Level Overview” on confluence and expand detail to show data flow and ordering of “Task/Process” boxes.

#### 5.5.1 MOPS

#### 5.5.2 Classification

#### 5.5.3 MakeSelectionMaps

#### 5.5.4 GatherContributed



## 6 Science Data Quality Analysis Pipeline

### 6.1 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.
- SDQA Pipeline shall provide 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.
- SDQA Pipeline implementation will monitor and harvest the outputs and logs of execution of other science pipelines, computing user-defined metrics.
- The outputs of SDQA Pipeline runs will be stored into a SDQA repository (RDBMS or filesystem based).

### 6.2 Key Tasks for Each Level of QA

The SDQA system will be an integrated framework that is capable of providing useful information at four different levels of the data system.

- QA Level 0 - Testing and Validation of DM sub-system in pre-commissioning
- QA Level 1 - Real-time data quality and system assesment during commissioning + operations
- QA Level 2 - Quality assessment of Data Releases

- QA Level 3 - Tools for the community to evaluate the data quality of their own analyses. These will be made available as well-documented and packaged versions of QA Levels 0–2.

### 6.2.1 QA0

Test the DM software during pre-commissioning as well as test software improvements during commissioning and operations, quantifying the software performance against known expected outputs. Validating the software and its performance on (selected) data.

(“Make me a three-color diagram, compute the width of point sources in the blue part of the locus”)

(“I have 20 visits all over the sky, I want to match up the results”)

The main components:

1. CI system that compiles code
2. Test execution harness – that runs test up to “weekly” scale?
3. Library of validation metrics codes – some has to come from Science Pipelines, but KPMs are delivered by SQuaRE
4. Instrumentation capability for computational performance metrics
5. Library of “instrumentations”
6. Interface to data products and QA metrics (including visualization)
  - (a) Tabular query result interface
  - (b) Visualizer for images
  - (c) Plotter
  - (d) SuperTask execution on selected data
7. Curated datasets to use in tests
8. Toolkit for analysis of QA outputs (drilldown into existing tests, ad hoc tests, ad hoc afterburners) – some has to come from Science Pipelines or SUIT but SQuaRE provides examples [move to Shared Software Components section]

- (a) Tools that perform computations
  - (b) Tools that perform visualization (using Butler if astronomical, maybe direct database if not)
9. Connection from analysis toolkit to validation metrics (attach common interactive plots to validation metrics)
  10. QA database including ingestion
  11. Notification system for threshold crossings

Prototypes for all of these exist except “Toolkit for analysis of QA outputs” and “Connection from analysis toolkit to validation metrics”

“Toolkit for analysis of QA outputs” will take more resources than the others listed above, but some may be already scheduled in other teams

### 6.2.2 QA1

Data quality in real time during commissioning and operations. Analyzes the data stream in real time, information about observing conditions (sky brightness, transparency, seeing, magnitude limit) as well as characterize subtle deterioration in system performance.

Validating the operational system.

Main components from above:

1. Library of validation metrics codes
2. Instrumentation capability for computational performance metrics
3. Library of “instrumentations”
4. Interface to results (including visualization)
5. Curated datasets to use in tests
6. Toolkit for analysis of failures (drilldown into existing tests, ad hoc tests, ad hoc afterburners) – some has to come from Science Pipelines or SUIT but SQuaRE provides examples
7. Connection from analysis toolkit to validation metrics
8. QA database including ingestion

New main components:

1. Harness for analyzing alert contents (and perhaps format)
2. Faster metrics codes to meet overall 60 second performance requirement for alert publication (but not necessarily for all QA processing, which must meet only throughput requirements)
3. Additional metrics/instrumentation codes (that must not disturb production system, including its performance, when dynamically inserted)
4. Output interface to “comfort” display (aggregation, trending, etc.)
5. Output interface to automated systems (drop alerts, reschedule field, etc.)
6. Correlator between telemetry streams and metrics
7. Input interface from sources of data not already present in Prompt Processing system
8. Fake source injection and analysis
9. Metrics codes specific for calibration/engineering/special- purpose images

### 6.2.3 QA2

Assess the quality of data releases (including the co-added image data products) performing quality assessment for astrometric and photometric calibration and derived products, looking for problems with the image processing pipelines and systematic problems with the instrument. Validating the Data Release products. All components from QA0 New main components:

1. DRP-specific dataset
2. Release data product editing tools (including provenance tracking)
3. Output interface to workflow system based on QA results and provenance
4. Provenance analysis tools
5. Output interface to Science Pipelines, including from QA database

6. Comparison tools for overlap areas due to satellite processing
7. Metrics/products for science users to understand quality of science data products (depth mask/selection function, etc.)
8. Characterization report for Data Release

#### 6.2.4 QA3

Data quality based on science analysis performed by the LSST Science Collaborations and the community. Level 0-2 visualization and data exploration tools will be made available to the community. Make all results from the above available. Make all of the above components available to some part of the community (could be just affiliated data centers or could be individual scientists) as a supported product. Ingest external science analysis data as Level 3 data products; ingest useful external science analysis tools.

#### 6.2.5 Prototype Implementation of PipeQA

The pipeQA prototype is a useful reference for exploring ideas and we mention it here to capture this prototype work.

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

The prototype used 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 prototype code is available in the [https://github.com/lsst/testing\\_displayQA](https://github.com/lsst/testing_displayQA) git repository.

## 7 Science User Interface and Toolkit

### 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<sup>7</sup>. 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).

---

<sup>7</sup>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 Algorithmic Components

### 8.1 Instrument Signature Removal

AUTHOR: Merlin

- Mask defects and saturation
- Assembly
- Overscan
- Linearity
- Crosstalk
- Full frame corrections: Dark, Flats (includes fringing)
- Pixel level corrections: Brighter fatter, static pixel size effects
- Interpolation of defects and saturation
- CR rejection
- Generate snap difference
- Snap combination

#### 8.1.1 AP: just skip some steps?

AUTHOR: Simon

#### 8.1.2 DRP: do all the steps

AUTHOR: Merlin

## 8.2 Artifact Detection

### 8.2.1 Single-Exposure Morphology

AUTHOR: Simon

- Find CRs via morphology.

- Find satellites via Hough transform.
- Find some optical ghosts (etc?) from bright star catalog and optics predictions.

### 8.2.2 Snap Subtraction

AUTHOR: Simon

- All of the above, but improve by looking at both snaps.

### 8.2.3 Warped Image Comparison

AUTHOR: Jim

- Find more optical artifacts by looking at differences between warped images (this is run during background matching).
- Find transient astronomical sources we don't want to include in coadds.

## 8.3 Artifact Masking/Interpolation

AUTHOR: Jim

- Set mask planes for all artifacts.
- Eliminate small artifacts by interpolating them.

## 8.4 Source Detection

AUTHOR: Jim

- Detect above-threshold regions and peaks in direct or difference images.
- Needs to work on preconvolved and unconvolved images.

## 8.5 Deblending

AUTHOR: Jim



### 8.5.1 Single Visit Deblending

- Generate HeavyFootprint deblends using only a single image.

### 8.5.2 Multi-Coadd Deblending

- Generate consistent HeavyFootprint deblends from coadds over multiple bands and possibly epoch ranges.

## 8.6 Measurement

AUTHOR: Jim

[ **NOTE:**  
Each bullet here is really a subsubsection, but Latex doesn't like that. Could we bump up Sections to Chapters or Parts to get another level, or do something equivalent? ]

[ **Measurement Algorithms Table:**  
Matrix of measurement algorithms and the contexts in which they're run, indicating which combinations are supported ]

### 8.6.1 Variants

- Single Visit Measurement
- Multi-Coadd Measurement
- Forced Measurement
- Difference Image Measurement
- Multi-Epoch Measurement

### 8.6.2 Algorithms

- Centroids
- Second-Moment Shapes

- Aperture Photometry
- Kron Apertures
- Petrosian Apertures
- Galaxy Models
- Moving Star Models
- Trailed Point Source Models
- Dipole Models
- Spuriousness

### **8.6.3 Blended Measurement**

- Deblend Template Projection
- Neighbor Noise Replacement
- Simultaneous Fitting
- Hybrid Models

## **8.7 Background Estimation**

AUTHOR: Simon

- Fit or interpolate large-scale variations while masking out detections.
- Needs to work in crowded fields.
- Needs to work on both difference images and direct images.
- Need to be able to compose backgrounds measured in different coordinate systems on different scales.
- Needs to work on single CCDs for AP even if we use full FoV in DRP.

## 8.8 Build Background Reference

AUTHOR: Simon

- Given multiple overlapping visit images (already warped to a common coordinate system), synthesize a continuous single-epoch image that can be used as a reference for background matching.

## 8.9 PSF Estimation

### 8.9.1 Single CCD PSF Estimation

AUTHOR: Simon

- Fit simple empirical PSF model to stars from a single exposure.
- No chromaticity.
- May use external star catalog, but doesn't rely on one.
- Used in Alert Production and BootstrapImChar in DRP.

### 8.9.2 Full Visit PSF Estimation

AUTHOR: Jim

- Decompose PSF into optical + atmosphere.
- Constrain model with stars, telemetry, and wavefront data.
- Wavelength-dependent.
- Used in RefineImChar in DRP.
- Must include some approach to dealing with wings of bright stars.

## 8.10 Aperture Correction

AUTHOR: Jim

- Measure curves of growth from bright stars.
- Correct various flux measurements to infinite.
- Propagate uncertainty in aperture correction to corrected fluxes; covariance is tricky.

## 8.11 Astrometric Solutions

AUTHOR: Simon

### 8.11.1 Single CCD (for AP)

- If this uses DRP's internal reference catalog, this does all we need. THIS IS A NEW DEPENDENCY BETWEEN DRP AND AP.

### 8.11.2 Single Visit

- Fit multi-component WCS to all CCDs in a single visit simultaneously after matching to reference catalog.

### 8.11.3 Joint Multi-Visit

- Fit multi-component WCS to all CCDs from multiple visits simultaneously after matching to reference catalog.

## 8.12 Photometric Solutions

AUTHOR: Simon (and Merlin?)

### 8.12.1 Single CCD (for AP)

### 8.12.2 Single Visit

- Fit zeropoint (and some small spatial variation?) to all CCDs simultaneously after matching to reference catalog.
- Need for chromatic dependence unclear; probably driven by AP.

### 8.12.3 Joint Multi-Visit

- Derive SEDs for calibration stars from colors and reference catalog classifications.
- Utilize additional information from wavelenth dependent photometric calibration built by calibration products production.
- Fit zeropoint and possibly perturbations to all CCDs on multiple visits simultaneously after matching to reference catalog.

### 8.13 Generate Diffim Template for a Visit

AUTHOR: Simon

- Determine appropriate template to use
- Generate template for observation (may include DCR correction)

### 8.14 PSF Matching

AUTHOR: Simon

#### 8.14.1 Image Subtraction

- Match template image to science image, as in Alert Production and DRP Difference Image processing.
- Includes identifying sources to use to determine matching kernel, fitting the kernel, and convolving by it.

#### 8.14.2 PSF Homogenization for Coaddition

- Match science image to predetermined analytic PSF, as in PSF-matched coaddition.

### 8.15 Image Warping

AUTHOR: Jim

#### 8.15.1 Oversampled Images

- Just use Lanczos.

#### 8.15.2 Undersampled Images

- Can use PSF model as interpolant if we also want to convolve with PSF (as in likelihood coadds). Otherwise impossible?

#### 8.15.3 Irregularly-Sampled Images

- Approximate procedure for fixing small-scale distortions in pixel grid.

## 8.16 Image Coaddition

AUTHOR: Jim

- Can do outlier rejection (but usually doesn't).
- Needs to propagate full uncertainty somehow.
- May need to propagate larger-scale per-exposure masks to get right PSF model or other coadded quantities.
- Should be capable of combining coadds from different bands and/or epoch ranges as well as combining individual exposures.

## 8.17 DCR-Corrected Template Generation

AUTHOR: Simon

- Somewhat like coaddition, but may need to add dimensions for wavelength or airmass, and may involve solving an inverse problem instead of just compute means.

## 8.18 Image Decorrelation

### 8.18.1 Difference Image Decorrelation

AUTHOR: Simon

- Fourier-space (?) deconvolution of preconvolved difference images before measurement - ZOGY as reinterpreted by Lupton (could apply correction in real space, too)
- Need to test with small-scale research before committing to this approach.

### 8.18.2 Coadd Decorrelation

AUTHOR: Jim

- Fourier-space/iterative deconvolution of likelihood coadds, as in DMTN-15.
- Need to test with small-scale research before committing to this approach.

## 8.19 Star/Galaxy Classification

AUTHOR: Jim

### 8.19.1 Single Visit S/G, Pre-PSF

- Select stars to be used in PSF estimation (mostly from moments).

### 8.19.2 Single Visit S/G, Post-PSF

- Extendedness or trace radius difference that classifies sources based on single frame measurements that can utilize the PSF model. Used to select single-frame calibration stars, and probably aperture correction stars.

### 8.19.3 Multi-Source S/G

- Aggregate of single-visit S/G post-PSF numbers in jointcal.

### 8.19.4 Object Classification

- Best classification derived from multifit and possibly variability.

## 8.20 Variability Classification

AUTHOR: John

### 8.20.1 Using forced photometry

### 8.20.2 Using DIASources

## 8.21 Proper Motion and Parallax from DIASources

AUTHOR: Simon

## 8.22 Association and Matching

### 8.22.1 Single CCD to Reference Catalog, Semi-Blind

AUTHOR: Simon

- Want to match in image coordinates, so also needs to transform reference catalog.
- Run prior to single-visit WCS fitting, with only telescope's best guess as a starting WCS.
- Single CCD form needed by AP.

### 8.22.2 Single Visit to Reference Catalog, Semi-Blind

AUTHOR: Simon

- Want to match in focal plane coordinates, so also needs to transform reference catalog.
- Run prior to single-visit WCS fitting, with only telescope's best guess as a starting WCS.

### 8.22.3 Multiple Visits to Reference Catalog

AUTHOR: Jim

- Match sources from multiple visits to a single reference catalog, assuming good WCSs.

### 8.22.4 DIAObject Generation

AUTHOR: Simon

- Match all DIASources to predicted Solar System object positions and existing DIAObjects and generate new DIAObjects. Definitely run in AP, maybe run in DRP.

### 8.22.5 Object Generation

AUTHOR: Jim

- Match coadd detections from different bands/SEDs/epoch-ranges, merging Footprints and associating peaks.
- Also merge in DIASources or (if already self-associated) DIAObjects.



### 8.22.6 Cross-Patch Merging

AUTHOR: Jim

- Resolve duplicates in patch overlap regions by flagging “primary” objects. Difficult due to blending.

### 8.22.7 Cross-Tract Merging

AUTHOR: Jim

- Resolve duplicates in tract overlap regions by flagging “primary” objects. Difficult due to blending.

## 8.23 Ephemeris Calculation

AUTHOR: Simon

- Calculate positions for all solar system objects in a region at a given time.

## 8.24 Make Tracklets

AUTHOR: Simon

- Make all tracklet pairs
- Merge multiple chained observation into single longer tracklets
- Purge any tracklets inconsistent with the merged tracklets

## 8.25 Attribution and precovery

AUTHOR: Simon

- Predict locations of known Solar System objects
- Match tracklet observation to predicted ephemerides taking into account velocity
- Update SSOjects
- Possibly iterate

## **8.26 Orbit Fitting**

AUTHOR: Simon

- Merge unassociated tracklets into tracks.
- Fit orbits to all tracks.
- Purge unphysical tracks.
- Update SSObjects
- Possibly iterate

## 9 Software Primitives

NEW STUFF:

- Convolution AND correlation
- Simple statistics on arrays

### 9.1 Applications Framework (WBS 02C.03.05, 02C.04.01)

#### 9.1.1 Key Requirements

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

- Classes to represent and manipulate mappings between device and astronomical coordinates;
- Classes to represent and manipulate images and exposures;<sup>8</sup>
- 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.

---

<sup>8</sup>images with associated metadata.

### 9.1.2 Baseline Design

This library will form the basis for all image processing pipelines and algorithms used for LSST, so special attention will be paid to performance. For that reason, this baseline design calls for a library of C++ classes and functions. Throughout construction these low level routines will be continually upgraded and refined to meet the performance and algorithmic fidelity requirements driven by the algorithmic requirements in other WBSs. If it proves impossible to meet performance goals based on pure C++ code, GPU support for some functions has been prototyped.

We expect that LSST stack developers, Level 3 pipeline developers, and a substantial group of end users will need to interact directly with the `afw` functionality. For that reason, it is exposed to Python callers through a Python module named `lsst.afw`. Throughout the construction period, we expect to devote effort to refining this interface layer to provide an idiomatic system which adheres to community norms and expectations.

### 9.1.3 Prototype Implementation

A prototype version of the required classes was described in the UML Domain Model ([LDM-133](#)) and implemented in LSST Final Design Phase, including prototype GPU (CUDA) support for major image processing functions (e.g., warping). A significant fraction of this code will be transferred into construction.

Work-in-progress code is available at <https://github.com/lsst/afw/>. The documentation for this code is located at <http://ls.st/w3o> and <http://ls.st/6i0>.

## 10 Glossary

**API** Applications Programming Interface

**CBP** Collimated Beam Projector

**CCOB** Camera Calibration Optical Bench

**CTE** Charge Transfer Efficiency

**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

**PFS** Prime Focus Spectrograph. An instrument under development for the Subaru Telescope.

**PSF** Point Spread Function

**QE** Quantum Efficiency

**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] M. Ankerst, M. M. Breunig, H.-P. Kriegel and J. Sander, **OPTICS: Ordering Points To Identify the Clustering Structure**, Proc ACM SIGMOD (1999).
- [2] P. Antilogus, P. Astier, P. Doherty, A. Guyonnet and N. Regnault **The brighter-fatter effect and pixel correlations in CCD sensors** Journal of Instrumentation, Volume 9, Issue 3, article id. C03048 (2014).
- [3] A. Becker et al, **Report on Late Winter 2013 Production: Image Differencing** <http://ls.st/x9f>.
- [4] A. Becker, **Report on Summer 2014 Production: Analysis of DCR**, [https://github.com/lsst-dm/S14DCR/blob/master/report/S14report\\_V0-00.pdf](https://github.com/lsst-dm/S14DCR/blob/master/report/S14report_V0-00.pdf).
- [5] J. S. Bloom et al, **Automating discovery and classification of transients and variable stars in the synoptic survey era**, PASP 124, 1175–1196 (2012).
- [6] J. F. Bosch, **Modeling Techniques for Measuring Galaxy Properties in Multi-Epoch Surveys**, PhD Thesis, University of California, Davis (2011). <http://adsabs.harvard.edu/abs/2011PhDT.....226B>
- [7] J. Bosch, P. Gee, R. Owen, M. Juric and the LSST DM team, **LSST DM S13 Report: Shape measurement plans and prototypes**, <https://docushare.lsstcorp.org/docushare/dsweb/ImageStoreViewer/Document-15298>
- [8] J. Bosch, **Measurement of Blended Objects in LSST**.
- [9] E. M. Huff et al, **Seeing in the dark – I. Multi-epoch alchemy**, <http://arxiv.org/abs/1111.6958>.
- [10] H. Furusawa et al, **Hyper Suprime-Cam Survey Pipeline Description**, [http://hsca.ipmu.jp/pipeline\\_outputs.pdf](http://hsca.ipmu.jp/pipeline_outputs.pdf).
- [11] 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).

- [12] M. J. Jee, J. A. Tyson, M. D. Schneider, D. Wittman, S. Schmidt and S. Hilbert, **Cosmic shear results from the Deep Lens Survey. I. Joint constraints on  $\Omega_M$  and  $\sigma_8$  with a two-dimensional analysis**, ApJ 765 74 (2013).
- [13] J. Kubica et al, **Efficiently Tracking Moving Sources in the LSST**, Bulletin of the American Astronomical Society, 37, 1207 (2005).
- [14] D. Lang, D. Hogg, S. Jester and H.-W. Rix, **Measuring the undetectable: Proper motions and parallaxes of very faint sources**, AJ 137 4400–4111 (2009).
- [15] R. H. Lupton et al, **SDSS Image Processing II: The *Photo* Pipelines**. <http://www.astro.princeton.edu/~rhl/photo-lite.pdf>
- [16] R. Lupton and Ž. Ivezić, **Experience with SDSS: the Promise and Perils of Large Surveys**, Astrometry in the Age of the Next Generation of Large Telescopes, ASP Conferences Series, Vol 338 (2005). <http://adsabs.harvard.edu/abs/2005ASPC..338..151L>
- [17] R. Lupton, M. Jurić and C. Stubbs, **LSST’s Plans for Calibration Photometry**, July 2015.
- [18] 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.
- [19] N. Padmanabhan et al, **An Improved Photometric Calibration of the Sloan Digital Sky Survey Imaging Data**, ApJ 674 1217–1233 (2008).
- [20] A. Rasmussen, **Sensor Modeling for the LSST Camera Focal Plane: Current Status of SLAC Originated Code** July 2015. <https://docushare.lsstcorp.org/docushare/dsweb/Get/Document-8590>.
- [21] J. Richards et al, **On Machine-learned Classification of Variable Stars with Sparse and Noisy Time-series Data**, ApJ 733 10 (2011).



- [22] E. F. Schlafly et al, **Photometric Calibration of the First 1.5 Years of the Pan-STARRS1 Survey**, ApJ 756 158 (2012).
- [23] C. W. Stubbs, **Precision Astronomy with Imperfect Fully Depleted CCDs – An Introduction and a Suggested Lexicon**, Journal of Instrumentation, Volume 9, Issue 3, article id. C03032 (2014).
- [24] A. S. Szalay, A. J. Connolly and G. P. Szokoly, **Simultaneous Multicolor Detection of Faint Galaxies in the Hubble Deep Field**, AJ 117 68–74 (1999).