LARGE SYNOPTIC SURVEY TELESCOPE

Large Synoptic Survey Telescope (LSST)

DM Science Acceptance Test Specification

L. Guy

LDM-639

Latest Revision: 2018-06-13

Abstract

This document describes the detailed test specification for the DM Science Acceptance.

Change Record

Version	Date	Description	Owner name
1.0	2018-06-12	Initial release of draft version.	gcm

Document curator: Leanny Guy
Document source location: `https://github.com/lsst/ldm-639`
Version from source repository: b46cb67

Contents

DM Science Acceptance Test Specification

1 Introduction

This document specifies the test procedure for the DM Science Acceptance.

1.1 Objectives

This document describes the test cases applicable to science requirements defined in LSE-61.

1.2 Scope

The scope of this document is to provide the test procedures required to validate the LSE-61 Data Management requirements.

1.3 Applicable Documents

LDM-148	LSST DM System Architecture
LDM-151	LSST DM Science Pipelines Design
LDM-294	LSST DM Organization & Management
LDM-502	The Measurement and Verification of DM Key Performance Metrics
LDM-503	LSST DM Test Plan
LSE-61	LSST DM Subsystem Requirements
LSE-163	LSST Data Products Definition Document
LSE-180	Level 2 Photometric Calibration for the LSST Survey

1.4 References

[1] **[LSE-61]**, Dubois-Felsmann, G., Jenness, T., 2017, *LSST Data Management Subsystem Requirements*, LSE-61, URL `https://ls.st/LSE-61`

[2] **[LSE-180]**, Jones, L., 2013, *Level 2 Photometric Calibration for the LSST Survey*, LSE-180, URL `https://ls.st/LSE-180`

[3] **[LSE-163]**, Jurić, M., et al., 2017, *LSST Data Products Definition Document*, LSE-163, URL https://ls.st/LSE-163

[4] **[LDM-148]**, Lim, K.T., Bosch, J., Dubois-Felsmann, G., et al., 2017, *Data Management System Design*, LDM-148, URL https://ls.st/LDM-148

[5] **[LDM-502]**, Nidever, D., Economou, F., 2016, *The Measurement and Verification of DM Key Performance Metrics*, LDM-502, URL https://ls.st/LDM-502

[6] **[LDM-503]**, O'Mullane, W., Jurić, M., Economou, F., 2017, *Data Management Test Plan*, LDM-503, URL https://ls.st/LDM-503

[7] **[LDM-294]**, O'Mullane, W., Swinbank, J., Jurić, M., DMLT, 2017, *Data Management Organization and Management*, LDM-294, URL https://ls.st/LDM-294

[8] **[LDM-151]**, Swinbank, J.D., et al., 2017, *Data Management Science Pipelines Design*, LDM-151, URL https://ls.st/LDM-151

2 Approach

To be completed.

2.1 Tasks and criteria

To be completed.

2.2 Features to be tested

To be completed.

2.3 Features not to be tested

To be completed.

2.4 Pass/fail criteria

The results of all tests will be assessed using the criteria described in LDM-503 §4.

2.5 Suspension criteria and resumption requirements

Refer to individual test cases where applicable.

2.6 Naming convention

The naming convention applied is derived from Jira Test Management Plugin.

3 Test Case Summary

Jira Id	Test Name
LVV-T61	Verify Associate Sources to Objects (DMS-REQ-0034) implementation
LVV-T77	Verify Best Seeing Coadds (DMS-REQ-0330) implementation
LVV-T72	Verify Coadd Image Method Constraints (DMS-REQ-0278) implementation
LVV-T64	Verify Coadded Image Provenance (DMS-REQ-0106) implementation
LVV-T73	Verify Deep Detection Coadds (DMS-REQ-0279) implementation
LVV-T71	Verify Detecting extended low surface brightness objects (DMS-REQ-0349) implementation
LVV-T66	Verify Forced-Source Catalog (DMS-REQ-0268) implementation
LVV-T75	Verify Multi-band Coadds (DMS-REQ-0281) implementation
LVV-T78	Verify Persisting Data Products (DMS-REQ-0334) implementation
LVV-T62	Verify Provide PSF for Coadded Images (DMS-REQ-0047) implementation
LVV-T68	Verify Provide Photometric Redshifts of Galaxies (DMS-REQ-0046) implementation
LVV-T74	Verify Template Coadds (DMS-REQ-0280) implementation
LVV-T80	Verify implementation of Detecting faint variable objects (DMS-REQ-0337)
LVV-T79	Verify implementation of PSF-Matched Coadds (DMS-REQ-0335)
LVV-T82	Verify implementation of Tracking Characterization Changes Between Data Releases (DMS-REQ-0

4 Test Cases

4.1 LVV-T61 - Verify Associate Sources to Objects (DMS-REQ-0034) implementation

Version	Status	Priority	Verification Type	Critical Event	Owner
1	Draft	Normal	Test	False	ktl

4.1.1 Test Items

4.1.2 Requirements

4.1.3 Requirements

- LVV-16 - DMS-REQ-0034-V-01: Associate Sources to Objects

4.1.4 Test Script

Step 1
The DM Stack shall be initialized using the loadLSST script (as described in DRP-00-00).

Step 2
A “Data Butler” will be initialized to access the repository.

Step 3
For each of the expected data products types (listed in Test Items section §4.3.2) and each of the expected units (PVIs, coadds, etc), the data product will be retrieved from the Butler and verified to be non-empty.

Step 4
Verify that sources have objects

Step 5
Verify that objects list sources that seem reasonably near them.

4.2 LVV-T77 - Verify Best Seeing Coadds (DMS-REQ-0330) implementation

Version	Status	Priority	Verification Type	Critical Event	Owner
1	Draft	Normal	Test	False	ktl

4.2.1 Test Items

4.2.2 Requirements

4.2.3 Requirements

- LVV-161 - DMS-REQ-0330-V-01: Best Seeing Coadds

4.2.4 Test Script

Step 1

Step 2
The DM Stack shall be initialized using the loadLSST script (as described in DRP-00-00).

Step 3
A "Data Butler" will be initialized to access the repository.

Step 4
For each of the expected data products types (listed in Test Items section §4.3.2) and each of the expected units (PVIs, coadds, etc), the data product will be retrieved from the Butler and verified to be non-empty.

Step 5
Explicitly create a coadd for a specified seeing range in each filter.

Step 6
Verify that these coadds exist.

4.3 LVV-T72 - Verify Coadd Image Method Constraints (DMS-REQ-0278) implementation

Version	Status	Priority	Verification Type	Critical Event	Owner
1	Draft	Normal	Test	False	ktl

4.3.1 Test Items

4.3.2 Requirements

4.3.3 Requirements

- LVV-109 - DMS-REQ-0278-V-01: Coadd Image Method Constraints

4.3.4 Test Script

Step 1
The DM Stack shall be initialized using the loadLSST script (as described in DRP-00-00).

Step 2
A “Data Butler” will be initialized to access the repository.

Step 3
For each of the expected data products types (listed in Test Items section §4.3.2) and each of the expected units (PVIs, coadds, etc), the data product will be retrieved from the Butler and verified to be non-empty.

Step 4
Verify that coadds were created following specification

4.4 LVV-T64 - Verify Coadded Image Provenance (DMS-REQ-0106) implementation

Version	Status	Priority	Verification Type	Critical Event	Owner
1	Draft	Normal	Test	False	ktl

4.4.1 Test Items

4.4.2 Requirements

4.4.3 Requirements

- LVV-46 - DMS-REQ-0106-V-01: Coadded Image Provenance

4.4.4 Test Script

Step 1

Step 2
The DM Stack shall be initialized using the loadLSST script (as described in DRP-00-00).

Step 3
A “Data Butler” will be initialized to access the repository.

Step 4
For each of the expected data products types (listed in Test Items section §4.3.2) and each of the expected units (PVIs, coadds, etc), the data product will be retrieved from the Butler and verified to be non-empty.

Step 5
Query and verify provenance of input images, and software versions that went into producing stack.

Step 6
Test re-generating 10 different coadds tract+patches based on the provenance image given

4.5 LVV-T73 - Verify Deep Detection Coadds (DMS-REQ-0279) implementation

Version	Status	Priority	Verification Type	Critical Event	Owner
1	Draft	Normal	Test	False	ktl

4.5.1 Test Items

4.5.2 Requirements

4.5.3 Requirements

- LVV-110 - DMS-REQ-0279-V-01: Deep Detection Coadds

4.5.4 Test Script

Step 1
The DM Stack shall be initialized using the loadLSST script (as described in DRP-00-00).

Step 2
A “Data Butler” will be initialized to access the repository.

Step 3
For each of the expected data products types (listed in Test Items section §4.3.2) and each of the expected units (PVIs, coadds, etc), the data product will be retrieved from the Butler and verified to be non-empty.

Step 4
Verify through inspection that per-filter coadds exist for each tract+patch possible

Step 5

Verify through inspection that the images used to generate those coadds met specified conditions

Step 6
Visually inspect a subset of the coadds to verify that they visually appear reasonable and to be from good quality data.

4.6 LVV-T71 - Verify Detecting extended low surface brightness objects (DMS-REQ-0349) implementation

Version	Status	Priority	Verification Type	Critical Event	Owner
1	Draft	Normal	Test	False	jbosch

4.6.1 Test Items

4.6.2 Requirements

4.6.3 Requirements

- LVV-180 - DMS-REQ-0349-V-01: Detecting extended low surface brightness objects

4.6.4 Precondition

Input Data
-- HSC Public Data Release

4.6.5 Test Script

Step 1
Verify that low surface brightness objects exist. I.e., objects at SNR ~ 10 with clear extendedness.

Step 2

The DM Stack shall be initialized using the loadLSST script (as described in DRP-00-00).

Step 3
A "Data Butler" will be initialized to access the repository.

Step 4
For each of the expected data products types (listed in Test Items section §4.3.2) and each of the expected units (PVIs, coadds, etc), the data product will be retrieved from the Butler and verified to be non-empty.

4.7 LVV-T66 - Verify Forced-Source Catalog (DMS-REQ-0268) implementation

Version	Status	Priority	Verification Type	Critical Event	Owner
1	Draft	Normal	Test	False	jbosch

4.7.1 Test Items

4.7.2 Requirements

4.7.3 Requirements

- LVV-99 - DMS-REQ-0268-V-01: Forced-Source Catalog

4.7.4 Test Script

Step 1
The DM Stack and Alert Processing packaged shall be initialized as described in LVT-T17 (AG-00-00).

Step 2
The alert generation processing will be executed using the verification cluster:

```
python ap_verify/bin/prepare_demo_slurm_files.py
# At present we must run a single ccd+visit to handle ingestion before
# parallel processing can begin
./ap_verify/bin/exec_demo_run_1ccd.sh 410915 25
ln -s ap_verify/bin/demo_run.sl
ln -s ap_verify/bin/demo_cmds.conf
sbatch demo_run.sl
```

and any errors or failures reported.

Step 3
A "Data Butler" will be initialized to access the repository.

Step 4
For each of the expected data products types (listed in §4.2.2) and each of the expected units (PVIs, catalogs, etc.), the data product will be retrieved from the Butler and verified to be non-empty.

Step 5
DIAObjects are currently only stored in a database, without shims to the Butler, so the existence of the database table and its non-empty contents will be verified by directly accessing it using sqlite3 and executing appropriate SQL queries.

Step 6
The DM Stack shall be initialized using the loadLSST script (as described in DRP-00-00).

Step 7
A "Data Butler" will be initialized to access the repository.

Step 8
For each of the expected data products types (listed in Test Items section §4.3.2) and each of the expected units (PVIs, coadds, etc), the data product will be retrieved from the Butler and verified to be non-empty.

Step 9

Verify that there exist entries in the forced-photometry table for all coadd objects for the PVIs on which the object should appear.

Step 10
Verify that there exist entries in a forced-photometry table for each image for all DIAObjects.

4.8 LVV-T75 - Verify Multi-band Coadds (DMS-REQ-0281) implementation

Version	Status	Priority	Verification Type	Critical Event	Owner
1	Draft	Normal	Test	False	ktl

4.8.1 Test Items

4.8.2 Requirements

4.8.3 Requirements

- LVV-112 - DMS-REQ-0281-V-01: Multi-band Coadds

4.8.4 Precondition

4.8.5 Test Script

Step 1

Step 2
The DM Stack shall be initialized using the loadLSST script (as described in DRP-00-00).

Step 3
A “Data Butler” will be initialized to access the repository.

Step 4

For each of the expected data products types (listed in Test Items section §4.3.2) and each of the expected units (PVIs, coadds, etc), the data product will be retrieved from the Butler and verified to be non-empty.

Step 5

The DM Stack shall be initialized using the loadLSST script (as described in LVV-T10 - DRP-00-00)

Step 6

A “Data Butler” will be initialized to access the repository.

Step 7

For each combination of tract/patch/filter, the PVI will be retrieved from the Butler, and the existence of all components described in Test items section §4.6.2 will be verified.

Step 8

Scripts from the pipe_analysis package will be run on every visit to check for the presence of data products and make plots

Step 9

Ten patches will be chosen at random and inspected by eye for unmasked artifacts.

Step 10

Verify that deep detection coadds exist based on all filters.

4.9 LVV-T78 - Verify Persisting Data Products (DMS-REQ-0334) implementation

Version	Status	Priority	Verification Type	Critical Event	Owner
1	Draft	Normal	Test	False	ktl

4.9.1 Test Items

4.9.2 Requirements

4.9.3 Requirements

- LVV-165 - DMS-REQ-0334-V-01: Persisting Data Products

4.9.4 Precondition

Precursor data from HSC PDR.

4.9.5 Test Script

Step 1

4.10 LVV-T62 - Verify Provide PSF for Coadded Images (DMS-REQ-0047) implementation

Version	Status	Priority	Verification Type	Critical Event	Owner
1	Draft	Normal	Test	False	ktl

4.10.1 Test Items

4.10.2 Requirements

4.10.3 Requirements

- LVV-20 - DMS-REQ-0047-V-01: Provide PSF for Coadded Images

4.10.4 Test Script

Step 1

Step 2
The DM Stack shall be initialized using the loadLSST script (as described in LVV-T10 - DRP-00-00)

Step 3
A "Data Butler" will be initialized to access the repository.

Step 4
For each combination of tract/patch/filter, the PVI will be retrieved from the Butler, and the existence of all components described in Test items section §4.6.2 will be verified.

Step 5
Scripts from the pipe_analysis package will be run on every visit to check for the presence of data products and make plots

Step 6
Ten patches will be chosen at random and inspected by eye for unmasked artifacts.

Step 7
Query range of positions on 10 different coadd images. Verify that reasonable PSFs are returned.

4.11 LVV-T68 - Verify Provide Photometric Redshifts of Galaxies (DMS-REQ-0046) implementation

Version	Status	Priority	Verification Type	Critical Event	Owner
1	Draft	Normal	Test	False	ktl

4.11.1 Test Items

4.11.2 Requirements

4.11.3 Requirements

- LVV-19 - DMS-REQ-0046-V-01: Provide Photometric Redshifts of Galaxies

4.11.4 Precondition

Input Data
HSC Public Data Release

4.11.5 Test Script

Step 1
load LSST stack

Step 2
Ingest HSC PDR data

Step 3
processCcd.py
multiBandDriver.py

Step 4
Estimate Photometric Redshifts

Step 5
Load into DRP Database

Step 6

Inspect database to verify that photometric redshifts are present for all objects

4.12 LVV-T74 - Verify Template Coadds (DMS-REQ-0280) implementation

Version	Status	Priority	Verification Type	Critical Event	Owner
1	Draft	Normal	Test	False	ktl

4.12.1 Test Items

4.12.2 Requirements

4.12.3 Requirements

- LVV-111 - DMS-REQ-0280-V-01: Template Coadds

4.12.4 Test Script

Step 1

Step 2
The DM Stack and Alert Processing packaged shall be initialized as described in LVT-T17 (AG-00-00).

Step 3
The alert generation processing will be executed using the verification cluster:

```
python ap_verify/bin/prepare_demo_slurm_files.py
# At present we must run a single ccd+visit to handle ingestion before
# parallel processing can begin
./ap_verify/bin/exec_demo_run_1ccd.sh 410915 25
ln -s ap_verify/bin/demo_run.sl
ln -s ap_verify/bin/demo_cmds.conf
sbatch demo_run.sl
```

and any errors or failures reported.

Step 4
A “Data Butler” will be initialized to access the repository.

Step 5
For each of the expected data products types (listed in §4.2.2) and each of the expected units (PVIs, catalogs, etc.), the data product will be retrieved from the Butler and verified to be non-empty.

Step 6
DIAObjects are currently only stored in a database, without shims to the Butler, so the existence of the database table and its non-empty contents will be verified by directly accessing it using sqlite3 and executing appropriate SQL queries.

4.13 LVV-T80 - Verify implementation of Detecting faint variable objects (DMS-REQ-0337)

Version	Status	Priority	Verification Type	Critical Event	Owner
1	Draft	Normal	Test	False	ebellm

4.13.1 Test Items

4.13.2 Requirements

4.13.3 Requirements

- LVV-168 - DMS-REQ-0337-V-01: Detecting faint variable objects

4.13.4 Precondition

Input Data
DECam HiTS data.

4.13.5 Test Script

Step 1
The DM Stack and Alert Processing packaged shall be initialized as described in LVT-T17 (AG-00-00).

Step 2
The alert generation processing will be executed using the verification cluster:

```
python ap_verify/bin/prepare_demo_slurm_files.py
# At present we must run a single ccd+visit to handle ingestion before
# parallel processing can begin
./ap_verify/bin/exec_demo_run_1ccd.sh 410915 25
ln -s ap_verify/bin/demo_run.sl
ln -s ap_verify/bin/demo_cmds.conf
sbatch demo_run.sl
```

and any errors or failures reported.

Step 3
A "Data Butler" will be initialized to access the repository.

Step 4
For each of the expected data products types (listed in §4.2.2) and each of the expected units (PVIs, catalogs, etc.), the data product will be retrieved from the Butler and verified to be non-empty.

Step 5
DIAObjects are currently only stored in a database, without shims to the Butler, so the existence of the database table and its non-empty contents will be verified by directly accessing it using sqlite3 and executing appropriate SQL queries.

Step 6
Identify 100 objects from Gaia with proper motions high enough to have detectably moved during HSC observations.

Step 7
Measure reported proper motion of these objects in DM Stack processing. Verify that it is

consistent with Gaia objects.

Step 8
Identify 100 quasars from color-space or existing extragalactic spectroscopic catalog.

Step 9
Measure lightcurves of these quasars. Determine if structure function is reasonable.

4.14 LVV-T79 - Verify implementation of PSF-Matched Coadds (DMS-REQ-0335)

Version	Status	Priority	Verification Type	Critical Event	Owner
1	Draft	Normal	Test	False	ktl

4.14.1 Test Items

4.14.2 Requirements

4.14.3 Requirements

- LVV-166 - DMS-REQ-0335-V-01: PSF-Matched Coadds

4.14.4 Test Script

Step 1

Step 2
The DM Stack shall be initialized using the loadLSST script (as described in DRP-00-00).

Step 3
A “Data Butler” will be initialized to access the repository.

Step 4

For each of the expected data products types (listed in Test Items section §4.3.2) and each of the expected units (PVIs, coadds, etc), the data product will be retrieved from the Butler and verified to be non-empty.

Step 5
Verify that PSF-matched coadds were created.

4.15 LVV-T82 - Verify implementation of Tracking Characterization Changes Between Data Releases (DMS-REQ-0339)

Version	Status	Priority	Verification Type	Critical Event	Owner
1	Draft	Normal	Test	False	jbosch

4.15.1 Test Items

4.15.2 Requirements

4.15.3 Requirements

- LVV-170 - DMS-REQ-0339-V-01: Tracking Characterization Changes Between Data Releases

4.15.4 Test Script

Step 1
The DM Stack shall be initialized using the loadLSST script (as described in DRP-00-00).

Step 2
A “Data Butler” will be initialized to access the repository.

Step 3

For each of the expected data products types (listed in Test Items section §4.3.2) and each of the expected units (PVIs, coadds, etc), the data product will be retrieved from the Butler and verified to be non-empty.

Step 4
The DM Stack shall be initialized using the loadLSST script (as described in LVV-T10 - DRP-00-00).

Step 5
A "Data Butler" will be initialized to access the repository.

Step 6
Scripts from the pipe_analysis package will be run on every visit to check for the presence of data products and make plots.

Step 7
The DM Stack shall be initialized using the loadLSST script (as described in LVV-T10 - DRP-00-00).

Step 8
A "Data Butler" will be initialized to access the repository.

Step 9
Scripts from the pipe_analysis package will be run on every tract to check for the presence of data products and make plots

Step 10
The DM Stack shall be initialized using the loadLSST script (as described in LVV-T10 - DRP-00-00).

Step 11
A "Data Butler" will be initialized to access the repository.

Step 12
For each processed CCD, the PVI will be retrieved from the Butler, and the existence of all components described in section Test Items (§4.6.2) will be verified.

Step 13
Scripts from the pipe_analysis package will be run on every visit to check for the presence of data products and make plots

Step 14
Five sensors will be chosen at random from each of two visits and inspected by eye for unmasked artifacts.

Step 15
The DM Stack shall be initialized using the loadLSST script (as described in LVV-T10 - DRP-00-00)

Step 16
A “Data Butler” will be initialized to access the repository.

Step 17
For each combination of tract/patch/filter, the PVI will be retrieved from the Butler, and the existence of all components described in Test items section §4.6.2 will be verified.

Step 18
Scripts from the pipe_analysis package will be run on every visit to check for the presence of data products and make plots

Step 19
Ten patches will be chosen at random and inspected by eye for unmasked artifacts.

Step 20
Prepare a second DRP run -> DPDD with different configuration parameters for this second test Data Release.

Step 21

Stage subset of products from first test Data Release to separate storage.

Step 22

Scientifically compare the results of the subset of that region of sky to those in the second test Data Release comparing the results of the DRP Scientific Verification tests.