LARGE SYNOPTIC SURVEY TELESCOPE

Large Synoptic Survey Telescope (LSST)

Data Management Test Plan

William O'Mullane, Mario Juric, Frossie Economou

LDM-503

Latest Revision: 2017-04-26

Abstract

This is the Test Plan for Data Management. In it we define terms associated with testing and further test specifications for specific items.

Change Record

Version	Date	Description	Owner name
D	1	2017-01-13	WOM
First draft			

Contents

1 Introduction **vi**

1.1 Objectives vi

1.2 Scope vi

1.3 Assumptions vi

1.4 Applicable Documents vi

1.5 Reference Documents vi

1.6 Definitions, acronyms, and abbreviations vii

2 Test Items **vii**

3 Roles and Reporting **vii**

3.1 Pass/Fail Criteria viii

4 Constraints and Limitations **ix**

4.1 Requirements Traceability Constraints ix

4.1.1 Scientific ix

4.1.2 Computational ix

4.1.3 KPMs x

4.2 Functional Requirements x

4.3 Interfaces xi

5 Master Schedule xi

6 Validation Tools xii

6.1 Introduction xii

6.2 Data Comparison Tools xii

6.3 Data Transformation Tools xiii

6.4 Analysis Tools xiii

7 Unit and Integration Tests xiii

7.1 Approach xiii

7.2 Test Coverage xiv

7.3 Unit and Integration Test Specification xiv

8 Validation Tests xiv

8.1 General strategy xiv

8.2 Test Designs xv

8.2.1 Test Design DM-Data Management-SYS-X xv

8.3 Test Case Specification xv

8.3.1 Test Case DM-Data Management-SYS-X-1 xv

9 Science Validation xvi

1 Introduction

THERE IS ONLY ONE OF THESE YOU PROBABLY WANT AN STS

1.1 Objectives

The Software Test Plan describes the system being tested, summarising the system context and decomposition. It sets out the test and verification approach for the system and describes constraints and limitations in the testing to be performed. The STP describes the unit and integration tests for the component modules of the system and describes the validation tests to be performed on the fully integrated system.

1.2 Scope

The Software Test Plan is to be executed by the CU prior to delivery to the DPC where the system will be operated. The DPC will execute integration and acceptance test involving this system within the context of the DPC processing systems. This document will be updated during the different Gaia cycle phases, according to the requirements updates.

1.3 Assumptions

This paragraph is optional. It describes the preliminary assumptions on which the overall testing strategy are based.

1.4 Applicable Documents

When applicable documents change a change may be required in this document.

? DPAC Product Assurance Plan
? Software Development Plan for DM
? Software Requirements Specification for Data Management,

1.5 Reference Documents

1.6 Definitions, acronyms, and abbreviations

The following table has been generated from the on-line Gaia acronym list:

Acronym	Description
CU	Coordination Unit (in DPAC)
DPAC	Data Processing and Analysis Consortium
DPC	Data Processing Centre
OF	Object Feature (source packet)
SP	Software Product
SPR	Software Problem Report
SRS	Software Requirements Specification
STP	Software Test Plan
STS	Star Tracker System
SVN	SubVersioN

2 Test Items

The test items covered in this test plan are Data Management and its consituent components:

- All the produsct - from KT diagrams
- Interfaces
- Procedures like Data release

3 Roles and Reporting

Tester report issues through Jira, but what other mechanisms will be used?

WHo directs OPS rehersals .. ?

Reports on rehersals .. issues and

Handeling failures - time ines for fix.

Not my section but a thought: System Engineering intends to capture commissioning tests through JIRA testing addons/plugins such as Kanoah. We could use some of these to capture functional tests for repeatability between operation rehearsals (they might be overkill for most tests though). –FE

Note that downstream text refers to "Software Review Board". We don't have such an entity so we need to either identify an existing entity or define the constitution of that Board

3.1 Pass/Fail Criteria

The Software Review Board will meet once a full run of all Test Cases has been performed, and subsequently after a complete run of all outstanding Test Cases.

A Test Case will be considered "Passed" when:

- All of the test steps of the Test Case are completed and
- All open SPRs from this Test Case agreed in Software Review Board are considered non-critical.

A Test Case will be considered "Partially Passed" when:

- Only a subset of all of the test steps in the Test Case are completed but the overall purpose of the test has been met and
- Any critical SPRs from this Test Case agreed in Software Review Board are still not closed.

A Test Case will be considered "Failed" when:

- Only a subset of all of the test steps in the Test Case are completed and the overall purpose of the test has not been met and
- Any critical SPRs from this Test Case agreed in Software Review Board are still not closed.

4 Constraints and Limitations

Wil: Describes the limitations and the constraints which apply to CU level tests of the system. lack of computing resources may mean that datasets are smaller or that full accuracy cannot be achieved. Explain what must be validated in the DPC tests

- Verification is being done on the basis of precursor data sets such as HSC, and eventually with engineering data from the LSST arrays. These are just a proxy for full-focal-plane on-site LSST data.
- Metric measurements and operational rehearsals during construction may not involve critical operational systems that are still in development. For example, while computational performance is being measured, computationally dominant algorithmic steps such as deblending and multi-fit are only modeled, since they have not yet been implemented; operational rehearsals are done without the factory LSST workflow system; etc.

4.1 Requirements Traceability Constraints

I felt a summary of the current state of play being verified could be useful to Wil. We don't have to leave it in the final document –FE

4.1.1 Scientific

Some science requirements are captured in LSE-29 (aka LSR) and flow down to LSE-30 (aka OSS) ; some also exist in LSE-163 (aka DPDD) and will flow down in LSE-61 (aka DMSR) *Flow-down is not complete, TJ is working on this*

4.1.2 Computational

There are requirements in LSE-61 (aka DMSR) which captures the LSE-30 (OSS) requirements that DM is responsible for. *In practice LSE-63 (the QA document) has not been flown down to LSE-61*. These are:

- The primary computational performance flown down from LSE-29 (LSR) is OTT1 which is the requirement to issue an alert within 60 seconds of exposure end.
- Another requirement flows down from LSE-29 is calculation of orbits within 24 hours of the end of the observing night
- There is a new (not yet baselined?) requirement for the calibration pipeline to reduce calibration observations within 1200 seconds
- A nightly report on data quality, data management system performance and a calibration report have to be generated with 4 hours of the end of the night

Note that there are no computational requirements on individual technical components eg. data processing cluster availability, database data retrieval speeds, etc. There is an upper limit on acceptable data loss, and there is a network availability requirement.

4.1.3 KPMs

As a proxy for validating the DM system, LDM-240 (aka "the spreadsheet") defined a set of Key Performance Metrics that the system could be verified against. KPMs were not formally flowed down from LSE-29 (LSR) through LSE-30 (OSS) although there is some overlap with LSE-29 requirements. [TJ is working on this]. In particular, the non-science KPMs only exist in LDM-240 *(spreadsheet/old plan)*.

[While verification was part of the SQuaRE WBS we prepared a KPM verification plan at the request of System Engineering - LDM-502. This work is now being led by Wil now I guess?]

4.2 Functional Requirements

Functional requirement are not explicitly called out as such. They are captured in LSE-61 (DMSR). *[When SQuaRE prepared the verification plan for SysEng, we were directed not to include functional requirements and limit ourselves to KPM. In general functional requirements are easy to verify by simply undertaking to perform the required functions in eg. operational rehearsals so maybe we could just say that?]*

4.3 Interfaces

There is an implicit, but not explicit, need to verify interfaces to other subsystems. The ICDs describing external interfaces are curated in Docushare Collection 5201. *[Integration used to be a Tucson role; I believe this is being led by the currently vacant Integration Scientist role? or whoever conducts the operation rehearsals?]*

Internal interfaces: currently we have no definitions and hence they are not verifiable presumably. If we did, I would propose: I

5 Master Schedule

The schedule for testing the system until launch. If some modules are scheduled for development after other, explain dependencies and impact on integration and validation tests.

Nightly Tests

Weekly Integration test with data ..

Interface tests (2by 2 and integrated E2E, Internal and External))

End to End Tests ?? Freeze software for Ops ..

WISE data to PDAC - ...

HSC reprocessing - yes see the data and also validate the ops platform . Validate some procedures like install some procedures etc ..

ZTF Alerts processing to valiate ALerts pipe ..

2018 Specrograph Data Acquasiitong Test..

2018 - Ops rehsal for comissioning - with a weeks comissioning say - pick which parts of plan we could reherase.

2019 - Ops rehsal #2 for comissioning - more complete .

2020 - Ops Rehersal Data Release (Comisisoning Data)
2021 - Ops Rehersal Data Release (Regular Data)

6 Validation Tools

6.1 Introduction

To evaluate the correctness of the generated data and the systems performances a set of tools may be developed or used. These tools will provide the means to facilitate the validation tasks. Following subsections describe the various tools that can used in the Data Managementvalidation (e.g. data comparison tools, analysis tools, etc).

6.2 Data Comparison Tools

This type of test tools are used to manage products in terms of:

- Comparison of a product generated during a test execution w.r.t. the relevant reference product
- Non regression verification comparing output products generated by different versions of the same system
- Measurement of quality degradation due to perturbed inputs

It allows:

- Product analysis
- Decoding of generated product allowing to read the most significant data of the product itself
- Visualisation of the values of a single selected field
- Apply an accuracy to the comparison
- Comparing specific parts of the products
- Filtering using flags values

6.3 Data Transformation Tools

These tools allow the data to be transformed to other formatted data.

6.4 Analysis Tools

Descriptions of the performance monitoring tools, profilers, test coverage programs... used in the Performance evaluation tests.
...

7 Unit and Integration Tests

7.1 Approach

Unit and Integration Tests will be automatically executed through the JUnit test framework. The descriptions of the test below are extracted from the test cases code and documentation.The results of Unit and Integration Test to be included in the Sofwtare Test Report will be generated automatically from the output of the execution of the tests by JUnit. A script will be provided to perform thes processing steps.

Module identification? (module tag in class header? mapping file?)

7.2 Test Coverage

Test coverage goal for unit and integration testing. Each class and public method shall have a JUnit test harness that may be labelled according to their purpose (e.g. I/O, individual class/method tests, software integration, data model integration etc.). Nominal and contingency tests should be clearly identified.

Interface coverage...

The tool [insert name of unit test coverage tool here] will be used to provide metrics on the code coverage by Unit Tests for Data Management and this metric will be provide in the Test Report.

7.3 Unit and Integration Test Specification

This is a example test plan record; this should be generated automatically.

Class	Unit Test Name	Purpose
Unit Test Class	Unit Test Method	Purpose of Unit Test from method header

8 Validation Tests

Validation of the system though Oerations Rehersals (and or end to end tests)

8.1 General strategy

Description of the general verification and validation strategy, decomposition into verification testing categories (e.g. science tests, SP external interface tests, algorithms interrelation and sequence). Assessed validation tests results shall be available over the software development duration: they are stored into SVN repository along with related input data, property-file, etc.

A subset of tests are run at DPC during software release qualification process, the results of DPC runs are compared with corresponding test outputs. During DPC integration tests, these assessed outputs will also allow to verify software non-regression.

8.2 Test Designs

8.2.1 Test Design DM-Data Management-SYS-X

8.2.1.1 Objective Explain the objective of this test design

8.2.1.2 Features to be tested

- Component A
- Component B

8.2.1.3 Features not to be tested

- Component C
- Component D

8.2.1.4 Approach Description of the approach to writing this test design

8.2.1.5 Test Cases List of test cases to be specified

Test Case	Description
DM-Data Management-SYS-X-1	Description of Validation Test

8.3 Test Case Specification

8.3.1 Test Case DM-Data Management-SYS-X-1

8.3.1.1 Testable Items List the components to be tested in this test case

8.3.1.2 Purpose Explain the purpose of this test case

8.3.1.3 Input Specification Describe the inputs to this test (data, written procedures, etc.)

8.3.1.4 Output Specification Describe the outputs of this test

8.3.1.5 Environment Describe the environment (computing resources etc) required for this test.

8.3.1.6 Inter-case dependencies If this test in dependent on another test having been completed successfully (for input data for example), state that here.

8.3.1.7 Test Procedure Describe the procedure to be performed

8.3.1.8 Test Verification Describe how to verify if the test has been successful.

9 Science Validation