# SED Library

## Project Test Plan

| | |
|---|---|
| **Project ID:** | 1.3 |
| **Revision:** | 1.0 |
| **Project Lead:** | Janet Evans |
| **QA&T Lead:** | Randy Thompson |
| **Document Status:** | In Review |
| **Software Version:** | 1.0 - Year 1 Release |
| **Reviewed by:** | Brian Thomas |
| **Review Date:** | Sep-20-2011 |

## 1 Goal

This document describes the details of how the Spectral Energy Distribution (SED) Java Library is to be tested through its life cycle for requirements to be implemented in Year 1 of the project.

## 2 References

- SED Library Derived Requirements. See Section "SED Library Requirements" near the end for concise listing.
- SED PDD
- Main SED testing page. Contains links to other SED project test plans.

## 3 Project Description

The SED Library is a component of the SED project and is a library designed to facilitate access and manipulation of Spectral Energy Distribution files in support of the Virtual Astronomical Observatory (VAO). From the VAO Statement of Work, a "library to allow software to handle (create, read, interpret, modify) SED data objects". It is implemented in Java.

## 4 Test Environment, Tools and Execution Considerations

This is a Java library, and shall be tested on a Fedora Core 14 platform which implements the SUN/Oracle JVM 1.6u25.

Because this is a library, testing is limited to static analysis, code coverage, unit and (limited) acceptance tests.

This document does not cover component integration testing of this product beyond the requirement 8. Extensibility; that testing should be described under the Test Plan for the component(s) which utilize this library. However testing in this fashion is expected, and the artifact(s) from this project must be deployed to a deploy repository to facilitate this type of testing.

All tests should be automated using VAO supported software. For unit tests, this means Java unit test

framework and build systems (JUnit and Maven 2.2.1). Static analysis using both PMD and FindBugs? software is to be utilized. Code coverage is to be achieved using Cobertura.

# 5 Additional Information

To faciliate reporting, unit test cases are to be implemented in files which are clearly labelled as that test case, ex. "TestCase3.java" and aggregate all of the unit tests for that test case within them. Alternatively, in the test cases listed below, the individual files which execute the (automated) tests for each test case may be indicated.

# 6 Test Cases

## 6.1 Static Analysis Tests

**TestCase0 : Static Analysis of Code**

*Procedure*: Incorporate PMD and FindBugs? software into build.

The build process shall incorporate Static Code Analysis software in order to catch defects. Both PMD and FindBugs? software shall be incorporated, and no defects of "High priority" or greater shall be allowed.

## 6.2 Acceptance Tests

**TestCase1 : Artifact Completeness**

*Requirement Identifier*: 1.5
*Procedure*: Manual inspection of project artifacts.

The system shall be checked for deliverable artifact completeness as specified by Requirements 1.5.1-1.5.3. Documentation will be checked for completeness, and comprehensibility.

## 6.3 Unit tests

**TestCase2 : VAO Platform Compliance**

*Requirement Identifier*: 1.1, 1.2., 1.3, 1.4
*Procedure*: Automated Java Unit test using JUnit

The platform shall be test built, packaged and all unit tests run on VAO platform(s; see above) using the Sun/Oracle? 1.6u25 SDK and maven 2.

**Test Case 3 : Validation of data product validators**

*Requirement Identifier*: 2
*Procedure*: Automated Java Unit test using JUnit. Manual validation of test data products for use in this test.
*Intercase Dependencies* : TestCase2, tests must be executed on all supported VAO Platforms.

This test shall test data product validators which will be created and incorporated ,as indicated below, to insure the SED library meets compliance with all supported data product standards.

Known example valid data products of types indicated in Requirement 2 shall be made part of the library, and tested with unit tests (below) which run the validators of the unit test suite. Data product types shall include any which the library is supporting for backward compatability.

**The following document(s) defines these file type standards ???, not in the requirements...**

**Test Case 4 *: General Programatic Library Interface Unit tests***

*Requirement Identifier*: 3, 4, 5, 6, 7, 9
*Procedure*: Automated Java Unit test using JUnit
*Intercase Dependencies* : TestCase2, tests must be executed on all supported VAO Platforms.

Interfaces for the key programatic functions of the library shall be created, documented and tested. These key functions, as indicated by the requirements, are "Req 3. create", "Req 4. read", "Req 5. write" and "Req 6. manipulate" and "Requirement 7. Operate". All implementing classes need to have unit tests which exercise the functional aspects of these interfaces. Code coverage will be used to quantify the implementation of unit testing and for classes which implement public interfaces. The code coverage must be greater than 50% for class and method coverage. For year 1, there is no set threshold for conditional coverage.

**Test Case 4.1; TestSpectrumIO** : Create SED data

*Requirement Identifier*: 3, 9

Unit tests shall exercise the creation API to serialize SED data to local disk. All supported data product types **(see document(s) for input data product specification doc)** shall be tested and validated. All created SED data products are to be validated using appropriate data product validators.

Operator requests which result in error states shall be reported by exceptions and tested for (as described in Requirement 9. Error Handling).

**Test Case 4.2** : Read SED data

*Requirement Identifier*: 4, 9

Unit tests shall exercise the reading API for SED data which are held locally or retrieved from an SED service. All input data products are to be validated as part of this test case.

Operator requests which result in error states shall be reported by Exceptions and tested (as described in Requirement 9. Error Handling). The test suite must test the reading of badly formatted/incomplete data products shall also be tested to insure graceful failure of the library when parsing of input data product types.

**Test Case 4.3** : Write SED data

*Requirement Identifier*: 5, 9

Unit tests shall exercise the writing API for SED data to local disk or to FileStream?. All serialized data products are to be validated as part of this test case.

Operator requests which result in error states shall be reported by Exceptions and tested (as described in Requirement 9. Error Handling). Common error states which must be tested include the inability to write data to disk (denied permission).

**Test Case 4.4** : Manipulate SED data

*Requirement Identifier*: 6, 9

Unit tests shall exercise the manipulation API of the library (as defined in Requirement 6. Manipulate) . In addition to checking for runtime errors and sanity checks on manipulated data products, all manipulated data are to be written out and checked for validity to insure conformance to SED standard (as defined in **data product format document**) above.

Operator requests which result in error states shall be reported by Exceptions and tested (as described in Requirement 9. Error Handling).

**Test Case 4.5** ***: Operate on SED data***

*Requirement Identifier*: 7, 9

Unit tests shall exercise the operation API of the library (as defined in Requirement 7. Operations). In addition to checking for runtime errors and sanity checks on manipulated data products, all manipulated data are to be written out and checked for validity to insure conformance to SED standard (as defined in **data product format document**) above.

Operator requests which result in error states shall be reported by Exceptions and tested (as described in Requirement 9. Error Handling).

**Test Case 5 : Extensibility and Sample Application**

*Requirement Identifier* : 8
*Procedure*: Automated Java Unit test using JUnit/TestNG.
*Intercase Dependencies* : TestCase2, tests must be executed on all supported VAO Platforms.

Unit tests shall be created to test the sample application which utilizes the SED library. Mock classes shall be created to test any extended functionality which is not implemented in the sample application.

# 7 Schedule

| Activity | Task | Dependencies | Responsible party | Start Date | End Date | Comments |
|---|---|---|---|---|---|---|
| Static Testing | implementation | VAO CI/Build system | QA&T Lead and SED Library Team | Feb 2011 | March 2011 | Implemented by Beta Test 1 |
| Unit Tests | implementation | | SED Library PD team | Feb 2011 | March 2011 | Implemented by Beta Test 1, includes coverage |
| Acceptance Test | execution | Passed prior testing | QA&T team | April 2011 | Jul 2011 | Verify deliverables during beta test 1,2 and at delivery |