PNNL-22455

Prepared for U.S. Department of Homeland Security
U.S. Customs and Border Protection and Domestic Nuclear Detection Office
under U.S. Department of Energy Contract HSHQDC-12-X-00376

ARES Software Test Plan

Rev. 0

CA Carlson
KE Dorow

May 2013

Proudly Operated by Battelle Since 1965

DISCLAIMER

PACIFIC NORTHWEST NATIONAL LABORATORY
operated by
BATTELLE
for the
UNITED STATES DEPARTMENT OF ENERGY
under Contract DE-AC05-76RL01830

Printed in the United States of America

Available to DOE and DOE contractors from the
Office of Scientific and Technical Information,
P.O. Box 62, Oak Ridge, TN 37831-0062;
ph: (865) 576-8401
fax: (865) 576-5728
email: reports@adonis.osti.gov

Available to the public from the National Technical Information Service,
U.S. Department of Commerce, 5285 Port Royal Rd., Springfield, VA 22161
ph: (800) 553-6847
fax: (703) 605-6900
email: orders@ntis.fedworld.gov
online ordering: http://www.ntis.gov/ordering.htm

This document was printed on recycled paper.
(9/2003)

Revision Log and Approvals

ARES Software Test Plan

Rev. No.	Date	Describe Changes	Pages Changed
0	5/03/13	Original Document Issued	NA

Name and Title	Approvals	Date
Kevin Dorow, Software Development Manager		
Carolyn Seifert, Project Manger		
Note: Document approvals that were received via email are retained in the project records. RADS ADC Identifier:		

Acronyms and Abbreviations

ARES	Aerial Radiological Enhanced Sensors
JIRA	(Project and Issue Tracking System)
LBNL	Lawrence Berkeley National Laboratory
PNNL	Pacific Northwest National Laboratory
TRIM	Total Records and Information Management

Contents

Figures and Tables

1 Purpose

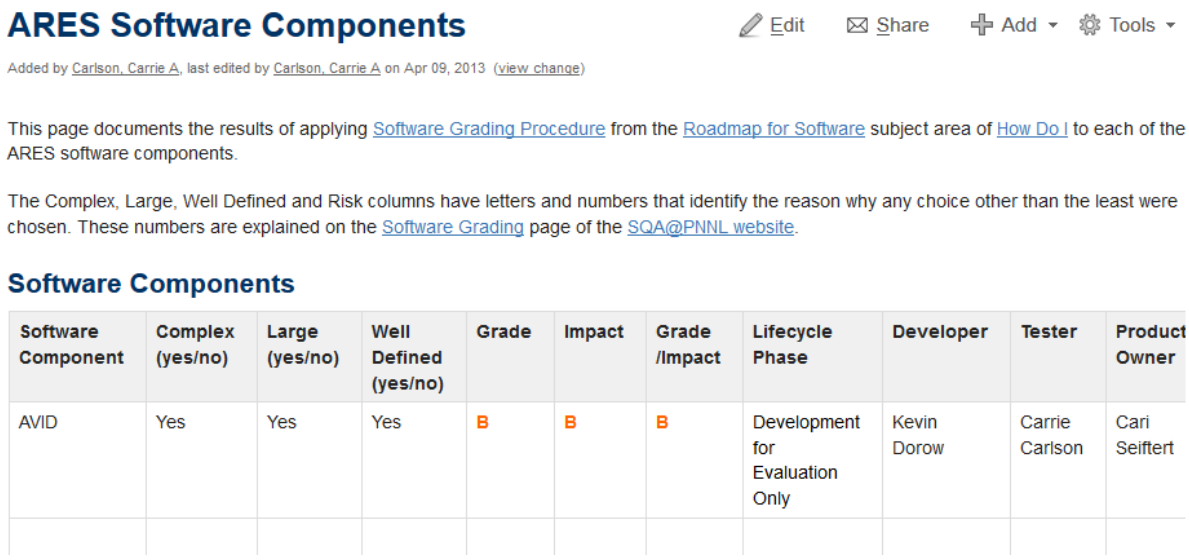
The purpose of this ARES Software Test Plan is to describe the scope, approach, resources, and schedule of testing activities that cover each type of software released or used on the ARES project. This document describes

- How testing will be performed for the ARES software applications generated by the Development and Algorithm teams
- Personnel conducting the testing based on the identified risk level

The software quality engineer may act as the test lead to manage testing activities to ensure compliance with the test plan and to provide a status report of testing activities to the rest of the project team.

1.1 Test Scope

The primary purpose of testing is to validate the software meets the defined requirements for each software item. Those software items are identified in the ARES Component list on the ARES Component Confluence website at:
https://confluence.pnnl.gov/confluence/display/ARES/ARES+Software+Components.



ARES Software Components

Edit Share Add Tools

Added by Carlson, Carrie A, last edited by Carlson, Carrie A on Apr 09, 2013 (view change)

This page documents the results of applying Software Grading Procedure from the Roadmap for Software subject area of How Do I to each of the ARES software components.

The Complex, Large, Well Defined and Risk columns have letters and numbers that identify the reason why any choice other than the least were chosen. These numbers are explained on the Software Grading page of the SQA@PNNL website.

Software Components

Software Component	Complex (yes/no)	Large (yes/no)	Well Defined (yes/no)	Grade	Impact	Grade /Impact	Lifecycle Phase	Developer	Tester	Product Owner
AVID	Yes	Yes	Yes	B	B	B	Development for Evaluation Only	Kevin Dorow	Carrie Carlson	Cari Seiftert

Figure 1.1. **An excerpt from the ARES Software Components List**

Each software component has its own confluence page linked from the components list. Each software item listed will be tested in accordance with a defined test scope specific to a release, associated test procedures, and cases. The test scope at a minimum will include the features to be tested and any test procedures that will not be covered.

Test case templates are available with the Zephyr plug-in for JIRA as shown below. The Zephyr plug-in is used to provide test planning, test case development and to record dynamic test results.

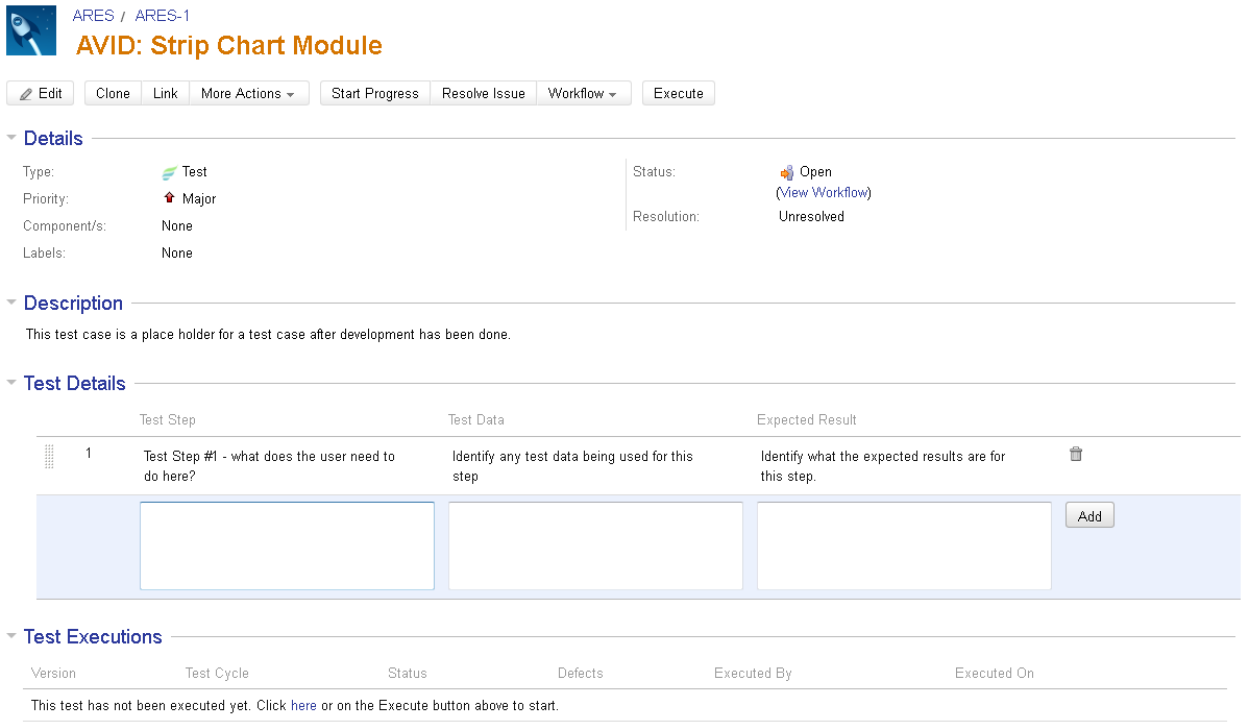


Figure 1.2. **Screen shot of Zephyr test case template**

Testers will document any specific test setup or environment requirements as part of the test case within Zephyr.

1.1 Test Environment

Testers are required to make sure they have current software on the computers being used to test with and the latest database build available (when needed). An Installation Guide (if available) provides specific information about the software setup required for testing. Although multiple builds may be available, testers with designated assignments via the test lead should not move to a new build unless the test lead has requested the entire test team move to the latest available build. This will allow for consistent testing and give testers enough time to test the available builds thoroughly. The most recent builds can be obtained from the Jenkins build server and installed in the test environment via the installation instructions provided by the developers, when needed. A standard test environment will be used whenever possible.

- Each tester will need the capability to complete a given test cycle of test cases on the Lawrence Berkeley National Laboratory (LBNL) testbed cluster (pnnl.ares.lbl.gov). Additionally all testers need to have access to use the supporting helicopter data which is accessible at the “/data” location on the LBNL cluster.
- Testers will need the permissions to access supporting helicopter data which is accessible at the “/data” location on the LBNL testbed cluster.

- All algorithms will have test data from specific tests that were run during the development phase of the algorithms. This test data will be used along with the results from the algorithm testing to compare to the results from the software.
- Test builds for installation are located on the Jenkins build server as shown below.

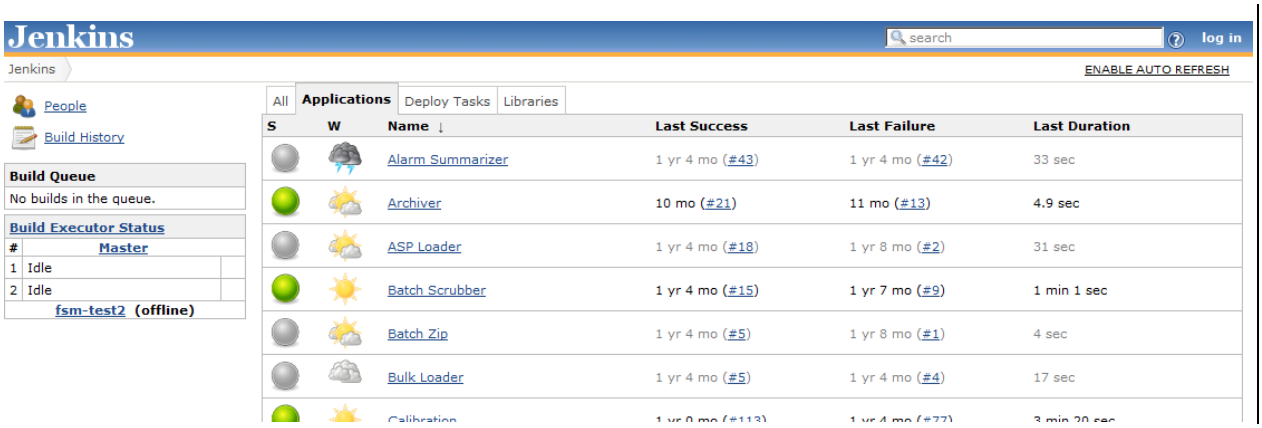


Figure 1.3. Screen shot of Jenkins Automated Build Server

1.2 Audience

The intended audience of this plan is the ARES software test team and DNDO and its evaluation team. The software test team verifies, tests, and reports on the functionality of software items. The test lead reviews the testing effort and all corresponding documentation to confirm that all procedures have been adequately tested.

1.3 Test Definitions

The following definitions are provided for information. These definitions are taken from the *IEEE Std 610.12-1990, IEEE Standard Glossary of Software Engineering Terminology*.

> A ***test procedure*** is defined as documentation specifying detailed instructions for the set-up and execution of a test. It is normally a set of general steps that completely test a component of a module of the system, or completely test one facet of that component. By one facet of a component, meaning that one procedure may test the mechanics of the component, while another may test the process of the component, and another may test how the component interacts with other components. The sum of the procedures for a component should completely test that component. Different modules may require more or less detailed treatment at the procedure level.
>
> A ***test case*** is defined as documentation specifying inputs, predicted results, and a set of execution conditions for a test item. It is normally a generalized detailed description of how to test a specific step that may occur in one or more procedures. This may be the details of how to test one control in a form, or it may be a description of how to test the

interaction of a group of controls, or it may be the detailed expectations after executing a function.

A ***test item*** is defined as a software item which is an object of testing.

2.0 Testing Approach

Testing will utilize JIRA and Zephyr for documenting all requirements traceability, test procedures and cases, and results. Zephyr is a test management tool used on various projects at Pacific Northwest National Laboratory (PNNL). Zephyr is a plug-in application and is used with JIRA. JIRA is an agile development tool used on many projects at PNNL to track User Stories and Acceptance Criteria (requirement) and tasks (work for developers to do) for a given sprint (a given amount of time to get a given amount of work completed).

JIRA with the Zephyr plug-in documents the project's software requirements for traceability to test cycles, test cases, and results. The test cycles and test cases together define "how" the software is tested, including expected results. The tool allows the testers to record actual results. These results are automatically traced to requirements and procedures. The tool also allows the test lead to manage the test process via its reporting capabilities, which include automatic requirements traceability linked to test results, and the status of test completion (such as the percentage of tests completed).

Confluence will serve as the repository for test documentation. Documentation is automatically created for items such as Software Requirements Documents, test reports, and progress reports and provided to ARES editors to be placed in the project format prior to being submitted to Total Records and Information Management (TRIM). This test plan will utilize the testing approach of independent testers. Each tester will be assigned a series of software items to test. The test lead is the point of contact for testing issues. The test lead will also be responsible for the test planning of a release which is documented in Zephyr that includes test cycles, test cases and reporting of the results. All testers will report any failures or anomalies using the JIRA application.

Algorithm developers will use the project defined standard cases to create test scenarios with documented results. The software testing team, which will include the algorithm developers, will use these cases and results in creating software tests to be executed on the software developed to integrate the algorithm functionality. The algorithm test lead will also define blind tests (and/or use blind test data from LBNL) with results that can be used for individual module testing as well as integration testing. Refer to the ARES Algorithm Test Plan for the project defined standard.

2.1 Requirements-Based Testing

Requirements will be documented for software items and entered into JIRA (as User Stories and Acceptance Criteria) for purposes of traceability. This allows the requirement to be identified and enables the test cycle or cases that exercise the requirement and the test result to be traced throughout the lifecycle of the software item. Appropriate tests that reflect the requirements will be included in the test cycle. These tests may be manual or automated. The requirements will be tracked using JIRA. The status of requirements-based testing will be updated through the testing period and will be included in the test results.

2.2 Manual Testing

For manual testing the approach utilized will be to track all tests generated and executed. Each tester will be assigned an area of the software to test.

If no test cases exist, the tester must develop them with the aid of the test lead and developer.

All manual tests will be entered into Zephyr as well as the results.

Testers may apply one or both of two testing methods. These methods are inspection and demonstration.

Testing by inspection is defined as running the software, then directly viewing the data structures.

Testing by demonstration is defined as running the software by the user interface, then viewing the results just as an operator of the system would.

In addition, each tester will be assigned any corresponding items to be retested for a software item.

Items will also be reviewed by the tester for incorporation into the current test suite for the purposes of regression testing. Unless specifically noted, manual testing requires demonstration.

All test procedures, cases, and related manual testing documentation will be maintained in Zephyr.

2.3 Automated Testing

Whenever possible, testing should be automated to ensure the consistency and efficiency of software testing. The approach for automated testing is to ensure that the existing automated test suite is accurate with the current build or release. It is possible to be out of synchrony, especially if new requirements are introduced prior to testing. The tester should run the test cycle against the current build to ensure that developer changes have not adversely affected the test suite. The tester will make any modifications needed to the test suite to ensure it will run. The results for this testing are automatically placed in the test results directory within the automated test suite at run time. A temporary results location will be used until it is determined that the test suite will run against the latest build. Once the tester determines the test suite is accurate then the tests will be run in the test results directory. The test results will indicate whether the test suite completed, if any failures were encountered, or if the test run was successful. The tester will report any failures or anomalies.

JIRA will be used to link the automated test to the requirements. Testers will be required to link the automated test suite so that requirements can be traced to test results. Automated tests generated are placed under configuration management and their location noted in the appropriate Zephyr test case.

2.4 Regression Testing

Testing is an iterative process like software development. Development often continues even though testing has started. This development often addresses problems or issues found by testing. Throughout the testing period, changes will be made and new builds will be issued that require rerunning and

modifying portions of the existing test suite. This is known as regression testing. The purpose of regression testing is to selectively test to ensure that modifications have not caused unintended adverse side effects and to verify that the modified system still meets the requirements. ARES software test suites will be designed to be consistent with the modular design of the software to better address regression testing. This allows the test lead to review the JIRA items fixed in each build prior to retesting. Based on the nature of the changes, the test lead can determine which regression suite to implement. For example, if a new build is released that has new functionality, but also contains major modifications to the software, it may not be necessary to rerun the entire test suite. Instead, the test lead may determine that only the one portion of the test suite is required.

2.5 Test Maintenance

Keeping the test suite up to date is important for future testing. All testers should ensure that the procedures they use are correct. If procedures are incorrect, the tester should take the time necessary to update the test suite. In some cases, such as with new functionality, there will be no procedures or cases existing. In this case, the tester should review any available documentation and work with the developer(s) to generate an adequate test suite. The following guidelines are given to testers for developing tests:

- Examine test procedures to ensure the completeness of the test against any existing documentation (i.e., requirements). Testers should use the standards identified below as a guide in developing and executing a test cycle.
- Testers should examine fixed items in builds to ensure that the test cycle accurately reflects the fix or change in functionality and update the test cycle as appropriate.
- In addition, the following standard should be used when developing test procedures and test cases. Cases are not directly linked to modules, but most will be associated with one.

For Case names, please use the following example format:
<ARES-XXX (automatically generated by JIRA> <Application Name and Short Title>
Example: ARES-20 Alarm Summary: Gamma Alarms.

2.6 Item Pass/Fail Criteria

Deviations or errors will be documented and reported using a JIRA ticket and noted in the test results.

Testers will mark the status of a completed test within Zephyr as follows:

- “Passed” is assigned if the testing effort demonstrates correct adherence to the expected result.
- “Failed” is assigned if the testing effort demonstrates a variance from the expected result that is deemed unacceptable.
- “WIP” is when the test case is under execution (not very often will this status be used but if the test will take some time a tester may use this status)
- “Blocked” may be assigned if there is something keeping the tester from running the test.

- "Unexecuted" may be assigned by the test lead should it be deemed necessary that only portions of the software need to be tested. Additionally this is the default status when a test case is scheduled but not yet executed.

All testers will submit a JIRA tickets for any defect or suggestion noted during testing. The tester can add notations, comments, and suggestions relating to items that are cosmetic in nature. Defects that significantly affect the quality or the goal of the release should be brought to the attention of the test lead immediately for resolution. Not all items will be approved for fix, and it is desirable that only items that are considered "Critical" or "High" priority be implemented; however, implementation of other items may be considered as well. Any item that is not approved for fix during the given testing cycle will be put on the backlog and will be reviewed by management for fixing during a later sprint.

Following the resolution of an item by the development team and approval of a new build, the testers will perform regression testing, retesting both the failed portions of the original test as well as any areas possibly affected by the item resolution.

2.7 Suspension Criteria and Resumption Requirements

Any problem found that results in a tester's inability to continue testing per their test assignment may require testing to be halted by the test lead. The test lead working with the development team should determine the severity of the issue and the impact it has on the overall execution of the test. Should it become necessary to discontinue testing, the test lead will work with the component owner to determine the necessary steps (for example, a schedule slip to accommodate a new build so the problem can be addressed).

3.0 Test Deliverables and Resources

The following test documentation will be maintained to ensure that the testing activities can be traced and defended. All test deliverables will be submitted to TRIM at the conclusion of the test cycle by the test lead.

Table 3.1**. Test Documentation List**

Deliverable	**Owner**
Test Cycle documents	test lead
Test Case Report	test lead
Requirements Traceability report	test lead
Test Summary Report	test lead

3.1 Test Resources

Testers run the formal test suites and are assigned software items to test. Each tester should run the test cycle that corresponds with the software item in Zephyr. Test resources are located in Confluence: https://confluence.pnnl.gov/confluence/display/ARES/ARES+Software+Components (as identified in figure 1.1).

4.0 Test Responsibilities

Overall responsibility for ensuring that testing occurs for each piece of software resides with the Test Lead, who may delegate the responsibility to a designated individual, referred to within this document as the test lead. All team members are accountable for implementing the required testing procedures and practices in their respective areas of responsibility. Team members will often have more than one role.

The key testing related responsibilities are listed below.

4.1 Software Test Lead

The test lead is responsible for:

- Generating and maintaining the software test plan.
- Ensuring that the test needs for all testers are addressed in a timely manner.
- Scheduling of testing activities, which includes resource assignments.
- Determining when builds are made available to the test team, and make build announcements to the test team to ensure the consistency and effectiveness of testing activities. Should it become necessary to rebuild the software, the test lead will work with the Project Manager to determine the impact on testing activities and resources.
- Ensuring that the results of testing are documented in the final test report to the software owner.
- Resolving any issues related to testing activities (for example, new builds, etc.).
- Tracking and reporting progress during testing.
- Determining if testing should be halted. Should it become necessary to discontinue testing, the test lead will work with the software owner to determine the necessary steps (such as a schedule slip).
- Ensuring that adequate system support is provided during the scheduled test period. This includes both System and Database Administration.

4.2 Testers

Each tester is responsible for:

- Working with the test lead to complete testing assignments.
- Ensuring that all problems identified during testing are documented in a JIRA ticket.
- Notifying the test lead of any issues that may affect testing, if applicable.
- Testing the software as identified by the test plan.
- Notifying the test lead or software owner of any items that require immediate attention during testing.
- Ensuring that test cases are up to date. This may require testers to rewrite or modify existing test suites.

- Ensuring that the test machine meets the specified configuration defined in the test setup section of the test plan. This includes using the latest approved build for testing as authorized by the test lead.

4.3 Staffing and Training Needs

The intent is to use experienced and trained independent PNNL testers who are familiar with JIRA and Zephyr for the majority of the system testing. Testers will also have algorithm domain knowledge as well as computer experience. No special training is required. JIRA and Zephyr training will be provided if necessary.

5.0 Test Schedule

The test lead will determine the appropriate test schedule for the test team. The test schedule may be included in other project scheduling activities.

6.0 Risks and Contingencies

Any software that has been identified as having a *high risk* "software risk" category will be required to document any high risk assumption for testing that could directly impact the quality of the software or testing activities. This should be included in the Test Plan within Zephyr. Use the following sample risk matrix as a starting point.

Table 6.1**. Sample Risk Matrix**

Risk 1:	This test plan assumes that all software deliveries will meet the schedule. Delayed delivery of software could result in schedule delay for integration and testing.		
Likelihood	**Difficulty**	**Potential Impact**	**Overall Risk**
Medium	Low	High	High
Mitigation Plan			
Owner	**Trigger**	**Corrective Action:**	
Test Lead	Missed software deliverables	Work with the project manager to determine new delivery dates and revise the test schedule as needed in the project plan.	
Preventive Actions:	Ensure that each of the software deliverables is delivered on schedule.		

7.0 Test Reporting Requirements

All test results will be documented to provide evidence that the software has been adequately tested. This approach will ensure that:

- A standard nomenclature is used to capture test results.
- Testers are using a consistent method to provide documented evidence of testing results.
- All testers document the test results in Zephyr.
- Any related JIRA tickets are written and noted with the test results.

7.1 Summary Test Report

The test lead creates a summary test report from Zephyr. This report is provided a PNNL number, edited for ARES format then reviewed and approved as appropriate. Once approved, the report will be submitted to TRIM as part of the testing package for the project records. Results can be reproduced in detailed format via Zephyr and the JIRA application at any time. At a minimum, the report will provide:

- A test case summary of the testing information on verification and regression activities.
- A description of the test suite and results.
- A summary of test cases related to requirements (as applicable).
- Statistical information on the breakdown of test procedure findings.

These reports will be submitted to DNDO if requested.