SIEMENS

Algorithm Validation Toolkit
AVT2EXT

Algorithm Validation Toolkit

AVT2EXT

Test Plan
Revision 0.1

Last Change: April 30, 2010

Robert W. Schwanke

History

Document History

Version/ Status	Date of Issue	Author	Change and Reason of Change
0.1 Draft		Robert W. Schwanke	Final draft prior to AVT2EXT release
1.0 Released	10-May-2010	Robert W. Schwanke	Released

History of released Versions

Version	Release date	Product Version
1.0	10-May-2010	AVT2EXT

Table of Contents

1. Introduction

1.1. Purpose of the document

The purpose of this document is to describe how system-level testing is done in the AVT2EXT project..

1.2. Definitions and abbreviations

See *AVT2EXT Definitions and Abbreviations,* a separate document.

1.3. References

[1] AVT2EXT Administration Guide , version 1.0
[2] AVT2EXT Definitions and Abbreviations, version 1.0
[3] AVT2EXT Design Specification, version 1.0
[2] AVT2EXT Functional Specification, version 1.0
[4] AVT2EXT Image Annotation User Manual, version 1.0
[5] AVT2EXT Implementation Plan, version 1.0
[6] AVT2EXT Installation Guide, version 1.0
[7] AVT2EXT Measurement Variability Tool User Manual, version 1.0
[8] AVT2EXT Programming Guide, version 1.0
[9] AVT2EXT Requirement Specification, version 1.0
[10] AVT2EXT Vision, Scope, and Technical Overview, version 1.0
www.openxip.org

2. Test-First Development

This test plan is formed around the concept of "Test-First Development", in which the software requirements and design are investigated and stabilized by documenting the tests that the software will be required to pass. These tests are written at a high level, first, but made concrete and detailed as soon as practical to do so. In Test-First Development, the tests and the software are developed concurrently, and with synergy, rather than one always being written first and the other designed to match it.

The test scripts for AVT serve several purposes at the same time:

- Introductory tour
- Integration and system testing
- Requirements coverage testing (acceptance testing)
- Supplement to the User Manual

These instructions give a tour of every feature of AVT, in a logical order that helps the user learn AVT and that also builds each step upon the previous ones. The test script is divided into numbered sections, each of which is considered to be one ***test case***. Each test case is marked with requirement keys that trace back to, and cover, the requirement specification. Each test case is also a leaf-level step in a use case, as documented in the Functional Specification. The title of the test case exactly matches the text of the use case step. Therefore, successfully executing the test scripts, with correct results, assures that every use case is implemented and every requirement that can be tested by software execution, has been tested.

2.1. Test Scripts

The test scripts begin at https://collab01a.scr.siemens.com/avtwiki/index.php/AVTWiki:Test_AVT_2_Extension and include several other pages referenced there, including, as of March 5, 2010:

https://collab01a.scr.siemens.com/avtwiki/index.php/AVT_Installation
https://collab01a.scr.siemens.com/avtwiki/index.php/Customize_Summary_Statistic
https://collab01a.scr.siemens.com/avtwiki/index.php/Outlier_Designer_dialog
https://collab01a.scr.siemens.com/avtwiki/index.php/Plotting_Designer_dialog
https://collab01a.scr.siemens.com/avtwiki/index.php/Statistic_Designer_dialog

C:\home\caBIG\AVT\
A snapshot of the main script is attached here: Docs\Test Plan\TestA
However, the current version supersedes the one above, and can always be found by following the link provided above.

2.2. Informal Test Procedure

The informal test procedure may be used whenever it is expected that the build being tested will NOT be released outside of the team, typically because the functionality is incomplete or the bug density is known to be high.

The procedure consists of the following steps, each of which is executed as many times as necessary:

- Execute each test case of the test script in order, with additional variations of order if desired.
- Each time a deviation between the expected and actual results occurs,
    - If the actual results are acceptable, edit the script to make the actual results the expected results.
    - If the actual results indicate that a prior bug has been resolved, adjust the script and update the bug report accordingly, taking care to leave a test in place to check whether the bug has reappeared.
    - If the deviation is a new manifestation of a bug already mentioned in the script, update the bug documentation as appropriate.
    - Otherwise
        1. File a bug report in the GForge bug tracker. Include a reference to the script section where it was uncovered.
        2. Describe the bug briefly in the script, with a URL pointing to the bug report. Include your wiki signature at the beginning of the description.
        3. Put a work-around in the script, if practical, and carefully mark it as such.
        4. Put the bug report title in the script change log.
- If a test case cannot be meaningfully executed because of a bug somewhere else, insert a comment at the beginning of the test case, listing the build id and explaining this.

2.3. Formal Test Procedure

This formal test procedure is used on Stable Builds that are being considered for external release. It may also be used on other builds.

1. Insert a line before the first heading of the main test script saying,
   “ <<wiki signature>> Currently testing Stable Build x.y.z.”
2. In the change log, say “Beginning to test Stable Build x.y.z.”
3. Carry out the Informal Test Procedure.
4. Replace the line inserted in step 1 with text of the following form:
   - o <<wiki signature>> Finished testing Stable Build x.y.z
   - o [[Testing Reports#Section | Test Report]]
5. In the change log, say “Finished testing Stable Build x.y.z

2.4. Bug Severity Scale

Blocker	This bug is preventing others on the team from making progress.
Critical	If this bug is not fixed quickly, major project damage could occur.
Major	Bug prevents execution of many test cases
Normal	Bug has primarily a localized impact.
Minor	Annoying but livable. No work-around needed.
Trivial	Cosmetic, or only noticeable to those who are sensitized to it.
Enhancement	Not wrong, but there is a better way.

2.5. Acceptance Criteria

We define the following levels of acceptance, each of which includes the ones above it:

- No Blockers
- No Criticals
- No Majors.
- Engineering: Every test case can be executed. All Normals have documented work-arounds.
- Release: Qualifies for Engineering acceptance and none of the bugs prevents end-to-end analysis of the Pivotal (40) cases.

2.6. Testing Report

A testing report consists of:

- Build identification
- Starting version ID of the test script
- Ending version ID of the test script
- List of bug reports created or modified
- Acceptance criterion achieved (see above)
- Comments

Testing reports are collected in the Wiki.