

Vector Software

WHITEPAPER

Using VectorCAST to Satisfy FDA Software Testing Requirements

Introduction

The VectorCAST family of products automates testing activities across the software development lifecycle. In this white paper, we provide a high-level view of how they provide automation to fully meet the software testing requirements as specified in *“General Principles of Software Validation; Final Guidance for Industry and FDA Staff”*, Section 5.2.5 *“Testing by the Software Developer”*.

The FDA guideline specifically addresses:

- > Unit Testing
- > Integration Testing
- > System Testing

Section 5.2.5 states:

“Testing entails running software products under known conditions with defined inputs and documented outcomes”

“Essential element of test is the expected result”

“Testing starts with unit test and concludes with system testing”

“The amount of structural coverage is commensurate with the level of risk posed”

“Regression analysis and testing are employed to provide assurance that a change has not created a problem”

“A software products testing can be organized into unit, integration, and system testing”

Using VectorCAST to Satisfy FDA Software Testing Requirements

VectorCAST supports all of the testing activities mentioned above, including the generation of all test artifacts required for FDA audit.

The VectorCAST product family consists of four complementary technologies:

VectorCAST/C++	<i>This tool automates the process of testing source modules written in C or C++; both unit testing and integration testing. Exchanges data with VectorCAST/Cover to provide 100% test coverage.</i>
VectorCAST/Cover	<i>Generates code-coverage artifacts during functional or system test.</i>
VectorCAST/RSP (Runtime Support Package)	<i>Extends VectorCAST/C++ to enable test execution for real-time applications on an embedded-target or in a simulator environment.</i>
VectorCAST/Manage	<i>Is used to automate the regression testing activities for all unit and integration tests.</i>

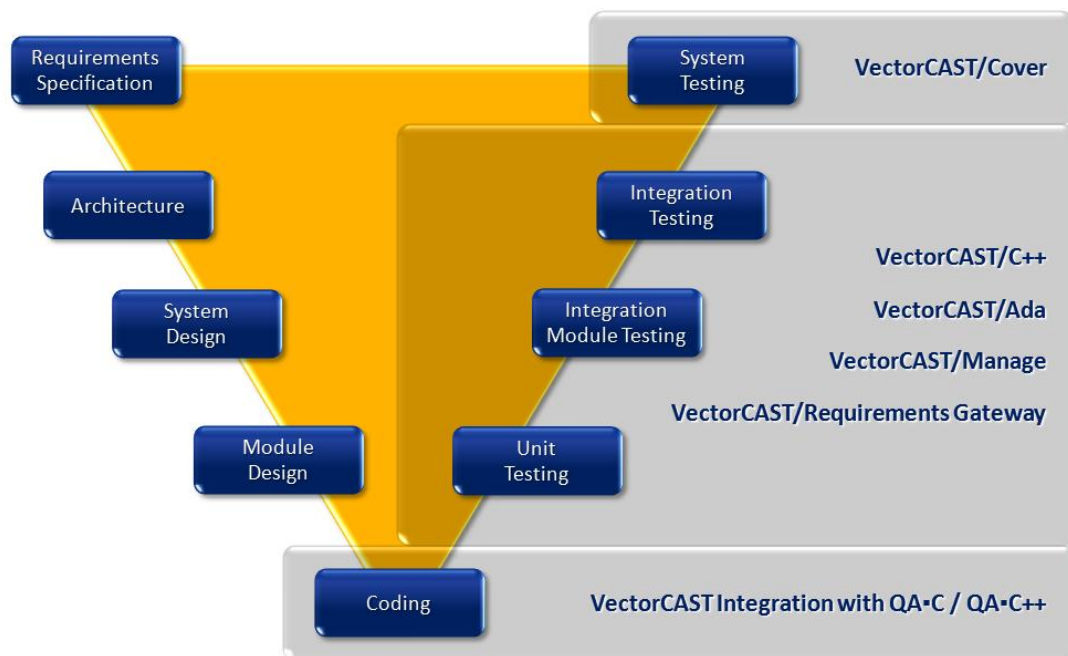


Figure 1 – Where VectorCAST is used in the software development process

VectorCAST/C++

VectorCAST/C++ automates key software testing activities, including:

Automatic generation of executable test harness - based on one or more files of source code

As illustrated in Figure 2, the executable harness generated by VectorCAST/C++ consists of a test driver, the source file(s) under test, any user-designated stubs for dependent functions, and all unstubbed dependents.

Data Driven Test Harness

Test data is read by the harness during execution. This approach precludes having to compile and link a new executable harness when the same tests are performed with different test cases.

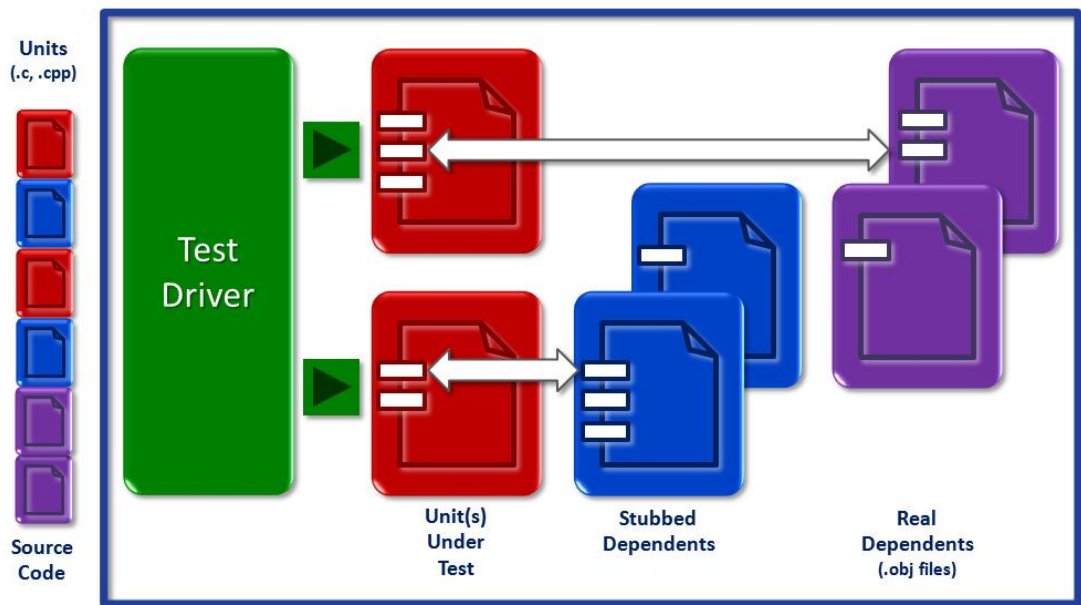


Figure 2 - Automated Test Harness Generation

Builds test cases for stimulating the source code under tests

VectorCAST/C++ will generate complete test cases automatically, or you can specify the input data and expected results for these. The test cases used (input data and expected results) are automatically preserved for subsequent regression testing.

Executes Tests

VectorCAST/C++ can execute tests in any of three ways:

1. On a host machine running natively on a PC or UNIX box; and
2. with a simulator or emulator (in conjunction with VectorCAST/RSP); or
3. directly on an embedded target (in conjunction with VectorCAST/RSP)

You can execute tests from either a GUI or a command-line-interface (CLI).

Reports pass/fail results and code coverage

VectorCAST/C++ generates pass/fail reports on the tests performed as well as coverage reports on the associated code. VectorCAST/C++ can provide coverage information at the statement, branch, and MC/DC levels.

VectorCAST/C++ can exchange coverage data with VectorCAST/Cover to ensure 100% coverage.

Facilitates regression testing

VectorCAST/C++ maintains all the necessary information for automated regression testing over the course of a development project. VectorCAST/C++ works in conjunction with VectorCAST/Manage to automate the entire regression testing process.

VectorCAST/Cover

VectorCAST/Cover gauges the effectiveness of system testing by determining which areas of an application have been exercised (or covered). You can perform coverage analysis on a portion of an application or on the entire application.

The coverage information generated by VectorCAST/Cover derives from system or functional testing. These tests are usually application specific and constructed to test high-level requirements. In this way, coverage information generated by VectorCAST/Cover differs from the information generated by VectorCAST/C++.

The coverage information generated by VectorCAST/C++ derives from unit or integration testing. These tests are typically constructed to determine whether algorithms are behaving as expected, whether anomalous conditions are being tested, and whether all paths are being tested.

Since it is usually it is not possible to test an entire application during system testing, a combination of unit, integration, and system test is necessary to achieve 100% coverage.

Ensuring 100% Coverage

Figure 3 illustrates two methods of using VectorCAST/Cover and VectorCAST/C++ to ensure 100% coverage.

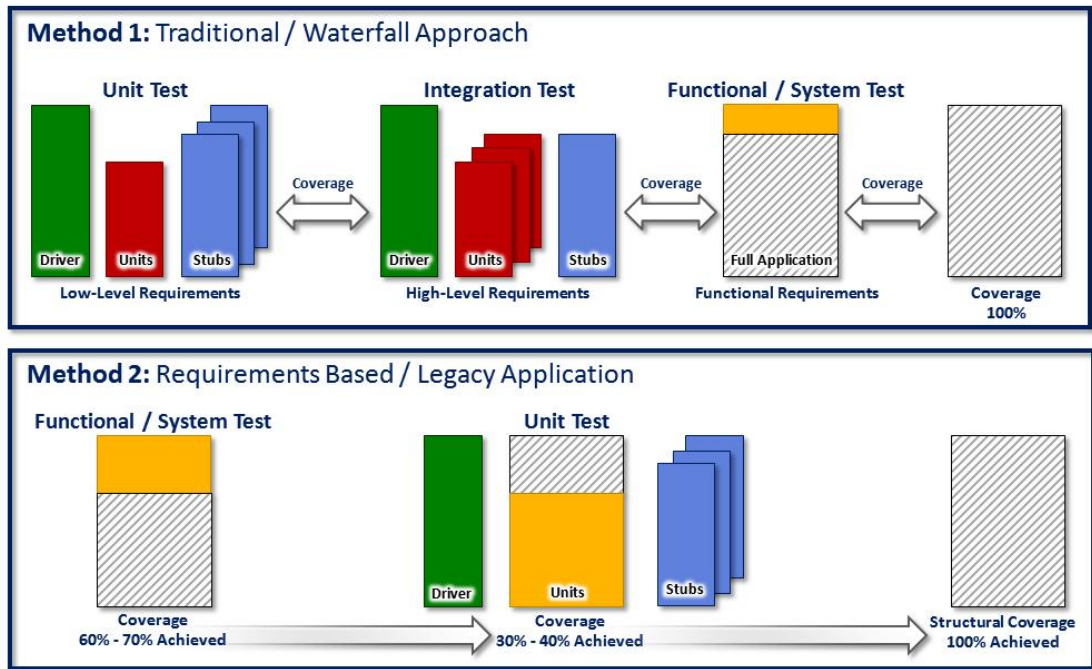


Figure 3 – Two Methods of Ensuring 100% Coverage

Using VectorCAST to Satisfy FDA Software Testing Requirements

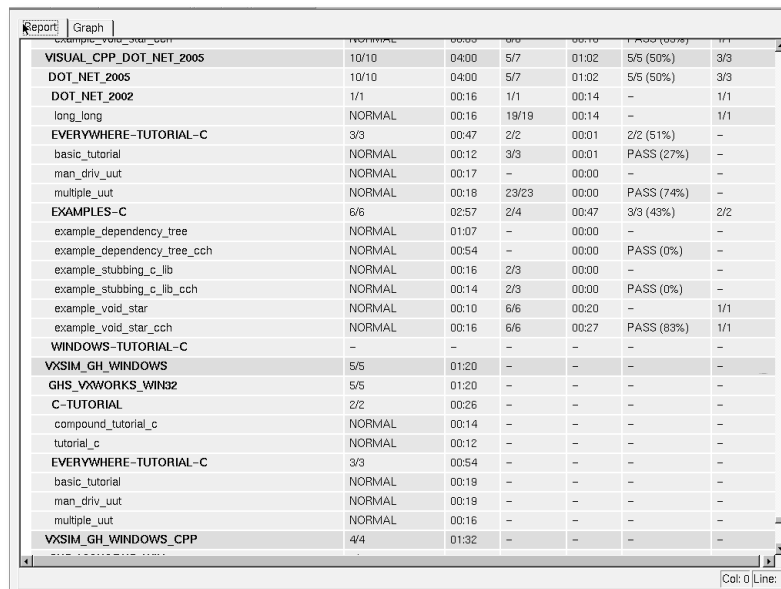
Method 1 applies to the traditional testing approach in which the testing progression is from individual components to aggregates to the full system. Method 2 applies to requirements-based testing, in which high-level system requirements are tested first, followed by low-level unit or integration tests.

Under the Method 1, VectorCAST/C++ conducts unit and integration testing on components and aggregates. In conjunction with this testing, you would use the built-in coverage utility to determine and report on source coverage. As you perform functional/system test, the VectorCAST/Cover tool exchanges coverage data from VectorCAST/C++ to ensure 100% coverage.

Under Method 2 (in conjunction with system testing) you use VectorCAST/Cover to identify any 'holes' in source coverage. You augment the coverage by using VectorCAST/C++ to run test cases on the files of source code containing the 'holes'. Again, VectorCAST/C++'s coverage-analysis utility exchanges coverage data with VectorCAST/Cover) to ensure 100% coverage.

VectorCAST/Manage

VectorCAST/Manage is an extension of the VectorCAST family of unit and integration testing tools. VectorCAST/Manage allows you to import previously developed VectorCAST test environments into regression test suites, providing a single point-of-control for all unit and integration test activities. At-a-glance logs, summary reports, and color-coded pass/fail criteria highlight the status of each test within the regression suite.



Test Name	Status	Start	End	Time	Pass	Fail	
VISUAL_CPP_DOT_NET_2005	NORMAL	10/10	04:00	5/7	01:02	5/5 (50%)	3/3
DOT_NET_2005	NORMAL	10/10	04:00	5/7	01:02	5/5 (50%)	3/3
DOT_NET_2002	NORMAL	1/1	00:16	1/1	00:14	-	1/1
long_long	NORMAL	10/10	00:16	19/19	00:14	-	1/1
EVERYWHERE-TUTORIAL-C	NORMAL	3/3	00:47	2/2	00:01	2/2 (51%)	-
basic_tutorial	NORMAL	00:12	3/3	00:01	PASS (27%)	-	-
man_driv_uut	NORMAL	00:17	-	00:00	-	-	-
multiple_uut	NORMAL	00:18	23/23	00:00	PASS (74%)	-	-
EXAMPLES-C	NORMAL	6/6	02:57	2/4	00:47	3/3 (43%)	2/2
example_dependency_tree	NORMAL	01:07	-	00:00	-	-	-
example_dependency_tree_cch	NORMAL	00:54	-	00:00	PASS (0%)	-	-
example_stubbing_c_lib	NORMAL	00:16	2/3	00:00	-	-	-
example_stubbing_c_lib_cch	NORMAL	00:14	2/3	00:00	PASS (0%)	-	-
example_void_star	NORMAL	00:10	6/6	00:20	-	-	1/1
example_void_star_cch	NORMAL	00:16	6/6	00:27	PASS (83%)	1/1	-
WINDOWS-TUTORIAL-C	-	-	-	-	-	-	-
VXSIM_GH_WINDOWS	NORMAL	5/5	01:20	-	-	-	-
GHS_VXWORKS_WIN32	NORMAL	5/5	01:20	-	-	-	-
C-TUTORIAL	NORMAL	2/2	00:26	-	-	-	-
compound_tutorial_c	NORMAL	00:14	-	-	-	-	-
tutorial_c	NORMAL	00:12	-	-	-	-	-
EVERYWHERE-TUTORIAL-C	NORMAL	3/3	00:54	-	-	-	-
basic_tutorial	NORMAL	00:19	-	-	-	-	-
man_driv_uut	NORMAL	00:19	-	-	-	-	-
multiple_uut	NORMAL	00:16	-	-	-	-	-
VXSIM_GH_WINDOWS_CPP	NORMAL	4/4	01:32	-	-	-	-

Example VectorCAST/Manage Report

VectorCAST for Embedded Testing

VectorCAST/C++ when used in conjunction with the VectorCAST/Runtime Support Package (RSP), allows seamless unit and integration testing of real-time applications - on the target itself or in a simulator. The VectorCAST/RSP integrates with the target CLI to invoke the cross compiler and to establish an I/O conduit between the target environment and VectorCAST/C++. Test execution is transparent to the user.

In reference to Figure 4, VectorCAST/C++ (on a host platform with RSP enabled) builds a test harness for one or more units of application code. The RSP ensures proper compilation and automatically downloads the executable harness onto the target environment. VectorCAST/C++ runs test cases (on the host platform harness) against the application modules residing on the target.

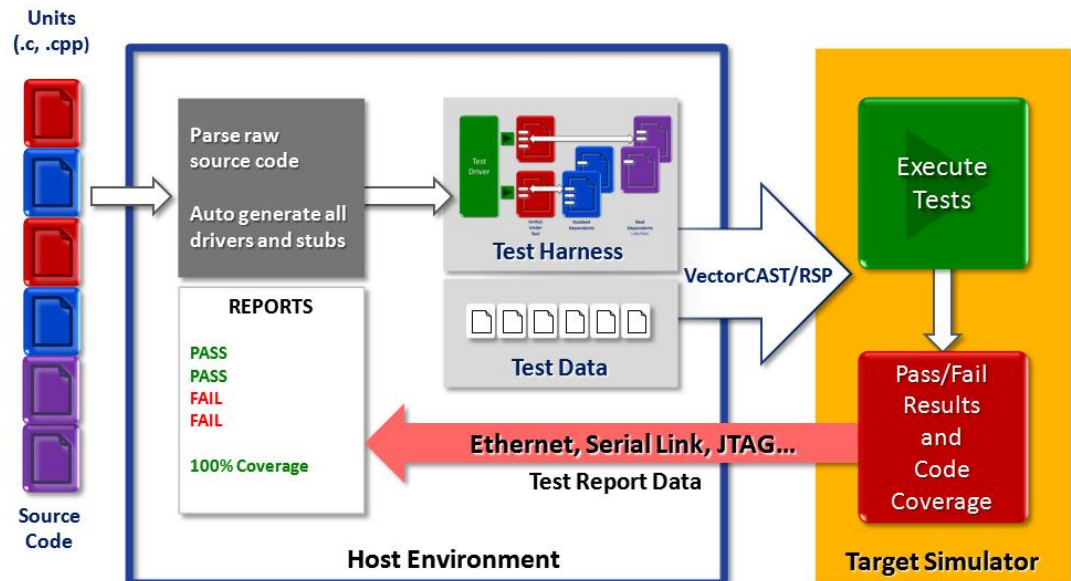


Figure 4 – VectorCAST Embedded Testing Architecture

Commercial Test Tool Validation

“General Principles of Software Validation; Final Guidance for Industry and FDA Staff”, Section 5.2.5, “Testing by the Software Developer”, states the following:

“Software testing tools are frequently used to ensure consistency, thoroughness, and efficiency in the testing of such software products and to fulfill the requirements of the planned testing activities. These tools may include supporting software built in-house to facilitate unit (module) testing and subsequent integration testing (e.g., drivers and stubs) as well as commercial software testing tools. Such tools should have a degree of quality no less than the software product they are used to develop. Appropriate documentation providing evidence of the validation of these software tools for their intended use should be maintained.”

VectorCAST provides tool validation in the form of qualification documents. The qualification documents consist of *Tool Operational Requirements (TOR)* and *Tool Qualification Data (TQD)*.

Tool Operational Requirements document includes:

- > *VectorCAST functionality in verifiable requirements*
- > *Project operational environment*
- > *Configuration management process*
- > *Method of attaining verification that VectorCAST has been satisfactorily tested against specified requirement*

Tool Qualification document includes:

- > *Tool qualification test data and results for each requirement specified*

The tool qualification process normally includes interaction with the qualifying user as well as the associated certification authority if applicable. Samples of these documents are available upon request.

Conclusion

As defined in “*General Principles of Software Validation; Final Guidance for Industry and FDA Staff*”, to provide thorough and rigorous examination of a software product, the development testing is typically organized into the levels outlined in Figure 5. These levels of testing and the associate test artifacts have been satisfied by many medical device manufactures using the VectorCAST product family.

FDA Developer Testing Levels	Satisfied with VectorCAST
<p>Unit (module or component) Level testing focuses on the early examination of sub-program functionality and ensures that functionality not visible at the system level is examined by testing. Unit testing ensures that quality software units are furnished for integration into the finished software product.</p>	<ul style="list-style-type: none"> •VectorCAST/C++ (unit test) •VectorCAST/RSP (target testing) •VectorCAST/Manage (regression test) •VectorCAST/Requirements Gateway (requirements tracing)
<p>Integration Level testing focuses on the transfer of data and control across a program's internal and external interfaces. External interfaces are those with other software (including operating system software), system hardware, and the users and can be described as communications links.</p>	<ul style="list-style-type: none"> •VectorCAST/C++ (unit test) •VectorCAST/RSP (target testing) •VectorCAST/Manage (regression test) •VectorCAST/ Requirements Gateway (requirements tracing)
<p>System Level testing demonstrates that all specified functionality exists and that the software product is trustworthy. This testing verifies the as-built program's functionality and performance with respect to the requirements for the software product as exhibited on the specified operating platform(s).</p>	<ul style="list-style-type: none"> •VectorCAST/Cover •VectorCAST/Requirements Gateway

Figure 5 – FDA Developer Testing Levels and VectorCAST

Additional Resources

This white paper is intended to serve as an introduction to the product literature available under **Resources** on www.vectorcast.com

VectorCAST/C++	<i>Unit and Integration Testing for C/C++</i>
-----------------------	---

VectorCAST/Cover	<i>The Multi-Language Code Coverage Tool</i>
-------------------------	--

VectorCAST/Manage	<i>Regression Test Manager</i>
--------------------------	--------------------------------

VectorCAST/RSP	<i>Real-Time Embedded Testing</i>
-----------------------	-----------------------------------

About Vector Software

Vector Software, Inc., is the leading independent provider of automated software testing tools for developers of safety critical embedded applications. Vector Software's VectorCAST line of products, automate and manage the complex tasks associated with unit, integration, and system level testing. VectorCAST products support the C, C++, and Ada programming languages.

Vector Software, Inc.

1351 South County Trail, Suite 310
East Greenwich, RI 02818
USA
P: 401.398.7185
F: 401.398.7186
E: info@vectorcast.com
W: vectorcast.com