



VectorCAST/Manage Automated Regression Test Management

Product Features:

- > Supports logical groups that map to your application architecture
- > Testing can be controlled from the local host, or any other network machine
- > Tests can be run on native workstations, or embedded targets
- > Tests can be scheduled and run automatically and unattended 24x7
- > Built-in SQL database to enable regression trend analysis
- > Built-in graphing to enable data visualization
- > Reporting format enables intuitive navigation from highest to lowest level of project
- > Integrated Python interpreter to extend analysis through complex scripting and result reporting
- > Full Command Line Interface for additional automation
- > Daily summary status of project wide test metrics

< The Regression Test Challenge >

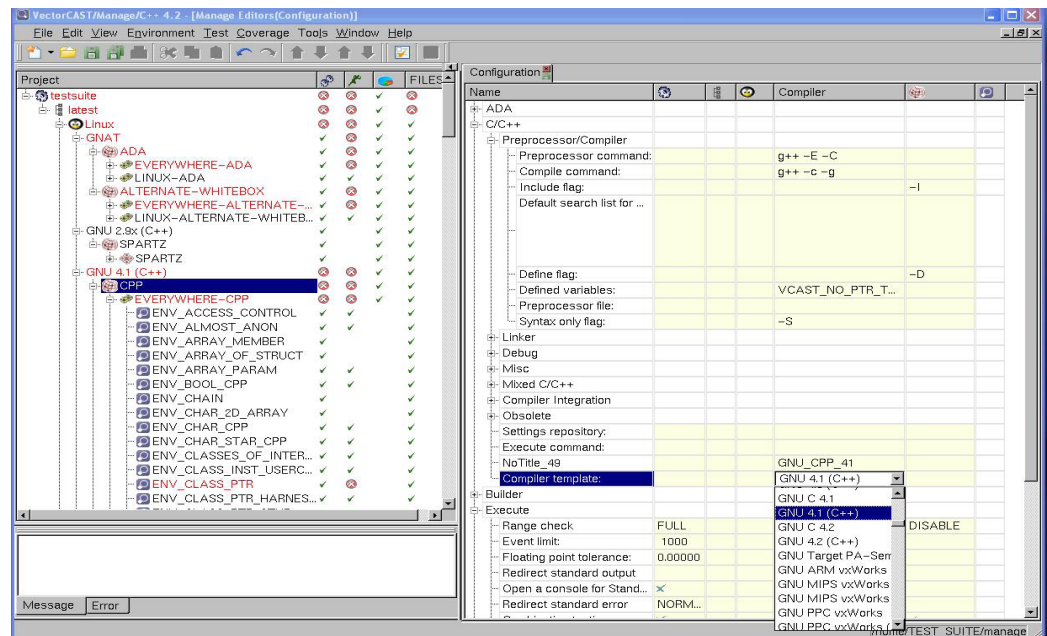
The goal of regression testing is to demonstrate that you have not "broken" something while fixing a bug or adding new functionality to your application. Many projects have no ability to build or modify source code with the confidence it will work because of the lack of regression testing. Proper regression testing of software applications requires the execution of hundreds to thousands of unit and integration test cases and the need to effectively manage the results. Historically, this has been a very difficult, if not impossible, and time consuming process.

< What is 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/ C++ and VectorCAST/Ada 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.

< Product Benefits >

- > Centralized management of all VectorCAST testing activities
- > Automated testing of multiple baselines and releases
- > Supports Extreme Programming, and Agile Development
- > Easy regression testing of an entire application at regular intervals
- > Accurate and immediate reporting of testing status
- > Easy identification of testing trends and regressions
- > Enables management to make confident build/release decisions



The main project tree shows the status for the entire test campaign.

< How it Works >

VectorCAST/Manage takes existing VectorCAST/C/C++/Ada environments and imports them into a VectorCAST/Manage project. These individual "test environments", can then be grouped into larger "Environment Groups", and "Test Suites". Environments can be members of multiple Environment Groups, and Environment Groups can be assigned to multiple Test Suites. (This enables users to structure their VectorCAST/Manage project to match the architecture of their application.

Because Environment Groups and Test Suites can be easily duplicated, the same tests can be run using various source baselines, on different host platforms, or with a different compiler or embedded target.)

< Integrated SQL Database >

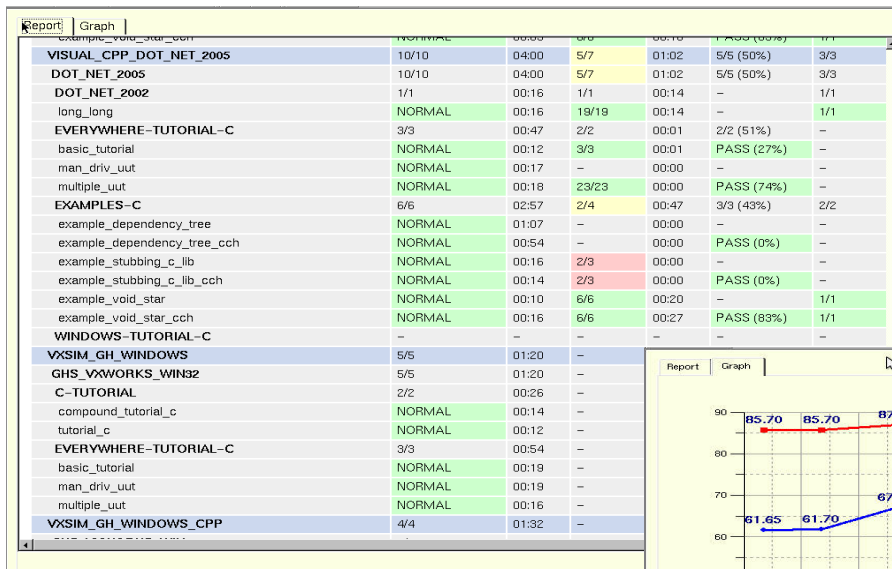
The integrated SQL database and graphing facility in VectorCAST/Manage enables users to view historical data for an individual software component, or any group of software components. This makes it easy to analyze regression trends across the software testing life cycle.

< How it is Used >

VectorCAST/Manage is used by the entire development team. Software managers use the high level reports and graphs to track testing progress and trends. QA Engineers use the tool to easily design test campaigns and monitor release readiness. Developers use the tool to identify and resolve defects.

< Management Summary >

The Management Summary report enables users to view the current status of each test case. Data is automatically recorded for build status and time, test execution status and time, and code coverage achieved. Using the integrated Python interpreter, additional "tests" can be added for each component and a column for the "test" will be added to the report. For example, a user might want to compare the test execution time to some threshold or might want to perform a file "diff" between a test artifact and a known previous result.



Environment	Status	Time	Pass	Fail	Pass (%)	Tests
VISUAL_CPP_DOT_NET_2005	NORMAL	04:00	5/7	01:02	5/5 (50%)	3/3
DOT_NET_2005	NORMAL	04:00	5/7	01:02	5/5 (50%)	3/3
DOT_NET_2002	NORMAL	00:16	1/1	00:14	-	1/1
long_long	NORMAL	00:16	13/19	00:14	-	1/1
EVERYWHERE-TUTORIAL-C	NORMAL	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	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	-	01:20	-	-	-	-
GHS_VXWORKS_WIN32	-	01:20	-	-	-	-
C-TUTORIAL	-	00:26	-	-	-	-
compound_tutorial_c	NORMAL	00:14	-	-	-	-
tutorial_c	NORMAL	00:12	-	-	-	-
EVERYWHERE-TUTORIAL-C	-	00:54	-	-	-	-
basic_tutorial	NORMAL	00:19	-	-	-	-
man_driv_uut	NORMAL	00:19	-	-	-	-
multiple_uut	NORMAL	00:16	-	-	-	-
VXSIM_GH_WINDOWS_CPP	-	01:32	-	-	-	-

HTML Reports allow reporting via email, web server or any HTML publishing application.

Graphs allow the identification of trends and regressions.

