

Product Features:

- > Supports testing of multiple product baselines
- > 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 24/7
- > 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 (CLI) for additional automation
- > Daily summary status of project wide test metrics

VectorCAST/Manage™

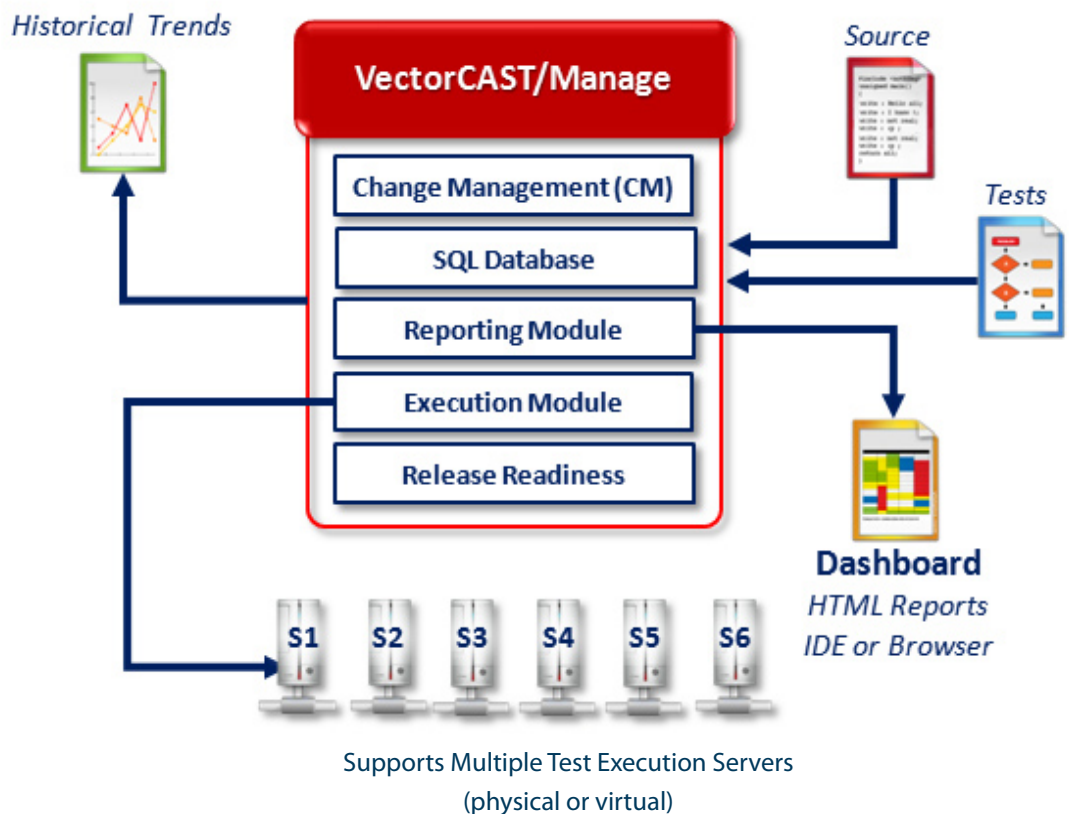
Automated Regression Test Management

<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 and time consuming, if not impossible 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. VectorCAST/Manage provides at-a-glance logs, summary reports, and color-coded pass/fail criteria highlight the status of each test within the regression suite.



VectorCAST/Manage

<How it Works>

VectorCAST/Manage takes existing VectorCAST/C++ and 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.

<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.

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

<Integrated SQL Database>

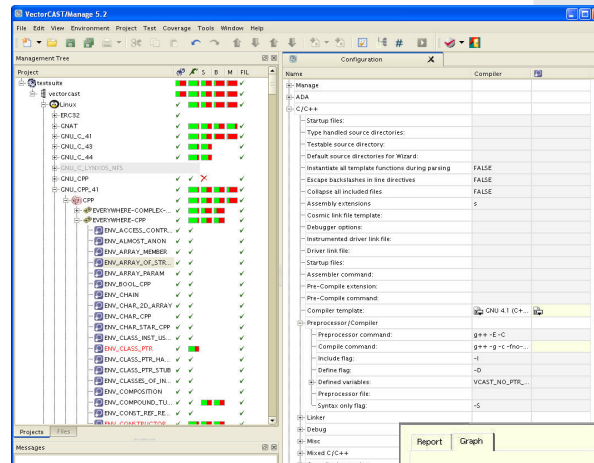
The integrated SQL database and graphing facility in VectorCAST/Manage enables users to view historical data for an individual software component or any group of software components. This makes it easy to analyze regression trends across the software testing life cycle.

<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.

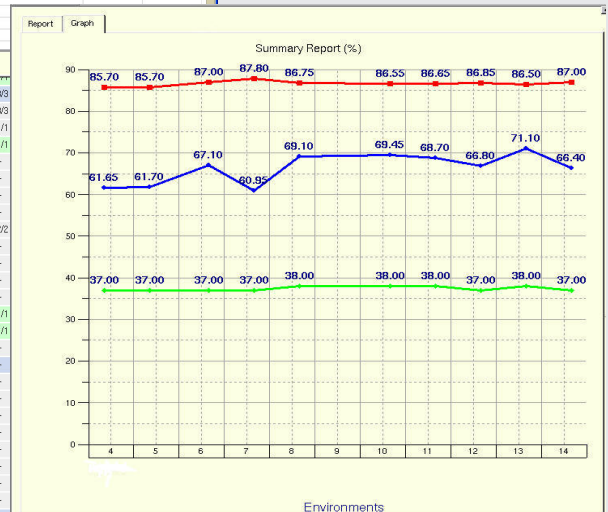
Product Benefits

- > Centralized management of all VectorCAST testing activities
- > Automated testing of multiple baselines and releases
- > Enables the continuous integration and test concepts of 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.

Test Case	Time	Build	Test	Code Coverage	Pass
VISUAL_CPP_DOT_NET_2005	19/10	0400	5/7	0102	5/5 (50%)
DOT_NET_2005	10/10	0400	5/7	0102	5/5 (50%)
DOT_NET_2002	1/1	0016	1/1	0014	-
long_long	NORMAL	0016	19/19	0014	-
EVERYWHERE-TUTORIAL-C	3/3	0047	2/2	0001	2/2 (51%)
basic_tutorial	NORMAL	0012	3/3	0001	PASS (27%)
man_dirv_uut	NORMAL	0017	-	0000	-
multiple_uut	NORMAL	0018	23/23	0000	PASS (74%)
EXAMPLES-C	6/6	0257	2/4	0047	3/3 (43%)
example_dependency_tree	NORMAL	0107	-	0000	-
example_dependency_tree_cch	NORMAL	0054	-	0000	PASS (0%)
example_stubbing_c_lib	NORMAL	0016	2/3	0000	-
example_stubbing_c_lib_cch	NORMAL	0014	2/3	0000	PASS (0%)
example_void_star	NORMAL	0010	6/6	0020	-
example_void_star_cch	NORMAL	0016	6/6	0027	PASS (83%)
WINDOWS-TUTORIAL-C	-	-	-	-	-
VXSIM_GH_WINDOWS	5/5	0120	-	-	-
GHS_VXWORKS_WIN32	5/5	0120	-	-	-
C-TUTORIAL	2/2	0025	-	-	-
compound_tutorial_c	NORMAL	0014	-	-	-
tutorial_c	NORMAL	0012	-	-	-
EVERYWHERE-TUTORIAL-C	3/3	0054	-	-	-
basic_tutorial	NORMAL	0019	-	-	-
man_dirv_uut	NORMAL	0019	-	-	-
multiple_uut	NORMAL	0016	-	-	-
VXSIM_GH_WINDOWS_CPP	4/4	0132	-	-	-



Graphs allow the identification of trends and regressions.