VECTOR

Baseline Testing Services

Whitepaper | Vx.x 2018-04

Table of Contents

1 Introduction

This case study provides details about a recently completed multi-engineer, multi-month Vector Services (VS) engagement that provided testing services to an existing VectorCAST customer. Real-world metrics for customers considering VS Baseline Testing Services are provided here.

2 What is Baseline Testing?

Baseline testing is useful for legacy code bases that have inadequate test cases. Often the lack of sufficient tests means that the application cannot be easily modified since changes often break existing functionality. Having test cases that formalize existing behavior enables developers to refactor and enhance the application with confidence.

3 Customer Challenge

Our customer has a code base of approximately 200,000 lines of C and C++ source code. The software is part of an embedded system that has both mission and human safety considerations. During the development of this code base, the majority of testing was performed at the system level after the various components had been integrated.

During system testing, the customer identified performance shortfalls, which were traced back to software defects. Because the customer was already using the VectorCAST tools on other projects, they contacted our VS group to help them implement unit testing and code coverage analysis for this project.

4 Project Details

VS proposed Baseline testing: development of unit tests to fully document existing behavior and provide 100% source code coverage of the application. These baseline tests would then be used by the developers as a regression test suite for any changes made to the application. Additionally, the developers would use VectorCAST to add tests for the new functionality, which will ensure that the 100% code coverage level is maintained and that new features don't break old functionality.

4.1 First Steps

In consultation with the customer, the seven application subsystems were ranked by criticality:

- 1 High Criticality (15k Lines of Code)
- 3 Medium Criticality (120k Lines of Code)
- 3 Low Criticality (68k Lines of Code)

Based on this ranking, a three-phase test approach was designed. VS has completed Phase I testing for the High Criticality sub-system, and is currently implementing Phase II testing for the Medium Criticality System. The remainder of this data sheet will discuss the findings from the Phase I engagement.

4.2 Project Management

VS assigned a Project Manager to this task, and he built a Project Management Plan (PMP) during the first week of engagement. A PMP establishes the working relationship, objectives, communication protocols, status reporting and detailed completion criteria.

4.3 Software Testing Goals

In consultation with the customer, VS established the following goals for the test activities:

- Correctness Based on Current Behavior: does all of the application have functional unit tests?
- Achieve 100% Source Code Coverage: has all of the source code been tested?
- Prove Robustness: does the application handle out of range values?
- Regression Testing: can the tests be easily re-used?

4.4 Robustness Testing

Not all Baseline Testing Services engagements include Robustness Testing. The value of Robustness Testing is to ensure that the application performs appropriately when unexpected inputs are received. Most software defects found in production software are the result of an unexpected sequence of operation, or out of range input values, as a result, Robustness Testing during development provides HUGE value.

VS created test cases which exercised flows of execution through the system, subsystem or component. The Robustness tests validated buffer and array bounds, found memory errors, as well as validated the robustness of system interfaces such as TCP/IP sockets and serial ports.

4.5 Regression Testing

All test cases were delivered as part of a VectorCAST/QA project that allows push button test execution. The ease-of-use provides tremendous cost savings for software maintenance. As the application changes, VectorCAST detects these changes and prompts the users to create new test cases, remove unnecessary ones or alter existing test cases.

Because all test preconditions, input values, expected results, and post conditions are captured in the test, no special knowledge is required.

4.6 Test Documentation

Three types of documentation were delivered:

- Workstation Setup: including installation and configuration of all tools needed
- Test Cases: including the purpose, test name, and expected execution results
- Defects: including the name of the file, function, line of code, test environment name, subsystem and the reason for the failure. In some cases, a suggestion on how to fix the problem was also provided.

4.7 Mentoring

One of the goals of each VS engagement is to mentor our customers to self-sufficiency. For this project, the VS staff worked closely with the customer's engineers to ensure that they understood the workflow, and could duplicate all testing in their production environment.

4.8 Application Metrics

Files	Functions	SLOC
42	591	14,994

4.9 Test and Coverage Metrics

Code Coverage Achieved	Correctness and Code Coverage Tests Created	Robustness Tests Created
99.5%, the remaining .5% was dead code	3,470	1,042

4.10 Defect Metrics

Vector Services identified over 200 defects which are summarized in the tables below. Notice that the Robustness Testing resulted in nearly twice as many “High” priority bugs as did the baseline and code coverage testing. This is not surprising, in that developers normally test the normal processing quite well and the unexpected processing poorly.

While Robustness Testing is much more complex and time consuming than Baseline Testing and requires a higher level of expertise to determine the areas of risk and create efficient tests, it provides significant benefits in the long term.

Correctness and Code Coverage Testing	High	Medium	Low	Totals
Defects Found	25	15	53	93

Robustness Testing	High	Medium	Low	Totals
Defects Found	48	44	29	121

Defect Type	Totals
Dead Code	43
Logic	112
Vulnerable	26
Suspicious	33

5 Project Results

With the testing goals established and a management plan in place, VS assembled a team, and built a suite of tests which satisfied all of the customer objectives. The tests were delivered with complete automation and are now being used for nightly automated regression testing. The delivered solution is plug and play for the customer; new software files are plugged into VectorCAST for retest, and the developers get a single Pass/Fail indication.

Additionally, the customer can now test new releases in their lab rather than on a fully integrated system, which has resulted in huge cost savings, and faster release cycles.

The project was completed on budget and ahead of the schedule. The project cost and elapsed time were both lower than the projected amount if the customer used their own resources.

The customer received such value from this work that Phase II, which consists of Baseline Testing for three medium criticality subsystems, is now underway.

Get More Information

Visit our website for:
> News
> Products
> Demo software
> Support
> Training classes
> Addresses

www.vector.com