

Structural Testing of Safety Critical Software

Ada
Aerospace Testing Expo 2005+

Embedded

Vector Software, Inc.

www.vectorcast.com

www.do178b.com

Structural Testing Introduction

- The “Structural testing” terminology is associated with FAA/Do-178B related programs
- The structural testing methodology revolves around testing code structures and showing the associated code coverage
- Structural testing can also be referred to as unit or module testing
- Involves testing of low level requirements
- Is usually performed after functional test
- Objective is to identify (with coverage) and test the code structures not previously exercised
- This presentation revolves around the mechanics of doing structural testing and achieving the desired code coverage

What is required?

Phases of Structural Test:

- **Developing Test Data and Test Cases**
- **Building the Test Simulation Environment (Commonly Stub and Driver Programs linked with Unit Under Test)**
- **Executing Test Cases and Determining Code Coverage**
- **Verify Desired Level of Code Coverage: Level A, B, or C**
- **Evaluation of Test Results and Report Generation**

The Structural Test Process

- Performed by Developers
- Is Extremely Labor Intensive
- Structural Test can Consume 50% of a Development Budget
- Requires Generation of at least 1 Test SLOC per Deliverable SLOC
- For DO-178B Applications this can be as high as 10 Test SLOC per Deliverable
- Getting “Ready to Test” can be 75% of the Effort

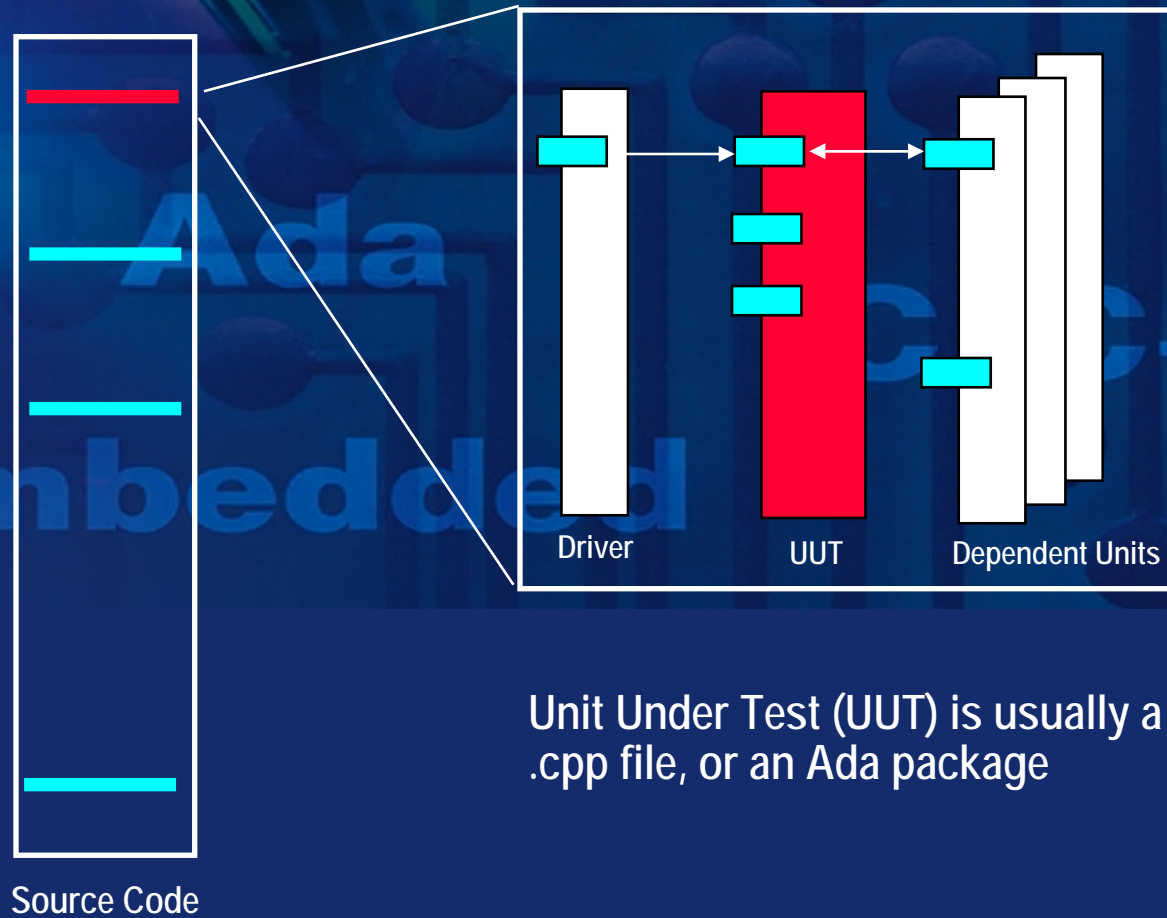
Your Test Process ?

- Left up to Discretion of Each Individual Developer
- Done Manually or with Home-grown Tools
- Looked Upon as an Art-Form vs. Engineering
- Difficult to QA Review and Peer Review
- Very Expensive to Administer
- Difficult to Attach a Price to the Test Effort
- Abandoned Once Integration Test is Reached
- No easy way to perform regression testing

Manual Testing

- 1) Determine the Closure of the Unit Under Test
- 2) Determine Which Dependent Units to Stub
- 3) Build Stubs
- 4) Build Test Driver to Exercise Unit Under Test
- 5) Compile/Link Test Harness - Iterative Process to Correct Compile Errors
- 6) Run Test Driver Main Program
- 7) Gather Test Data Results and Compile into Readable Format
- 8) Build Test Reports for Individual Test Case and Include in SDF
- 9) Determine Level of Source Code Coverage
- 10) Modify Harness Components for Next Test Case

Building a Test Environment



Unit Under Test (UUT) is usually a .c or .cpp file, or an Ada package

Using an Automated Tool (VectorCAST)

- Parses all project code files to determine data and call dependencies between units
- Allows real code or stubs to be used for dependent units
- Builds the Test Driver Necessary to Invoke the Subprograms within the Unit Under Test
- Compiles and Links the Test Harness into an Executable
- Provides a Test Case Construction Utility to Assist in Building Test Cases
- Provides a Test Execution Manager to Run the Test Cases Generated
- Automatically Generates a Detailed Test Report Associated With Each Test Case
- Provides Comprehensive Coverage Analysis

Features of an Automated Tool

- Utilizes Code Generators to Automatically Build all Source Code Required for Test Simulation Environment
- Interactive Point and Click Test Case Construction
- Allows for Black Box and White Box Testing
- Automates the Regression Testing Process with the Re-Execution of Tests and the Comparison of Expected and Actual Results
- Provides Detailed Standards-Compliant Test Reports
- Allows Test Case Creation and Execution Without Compilation
- Provides Graphical and Command Line Interfaces
- Fully Integrated with Leading Compilation Systems
- Allows Harness and Test Case Building from Scripts

Benefits of an Automated Tool

- Consistent Test Methodology / Documentation
- Quick Pay Back (Labor Saved and Quality Improved)
- Precludes the Writing of Disposable Test Software
- Satisfies Mil-Std, FAA, FDA and Automotive Test Reporting Requirements
- Allows Testing to Begin Sooner (stubbing)
- Supports Target Machine Testing (VectorCAST/RSP)
- Removes the Mundane Tasks Associated with Test
- Automates What Developers do Manually with Minimal Training
- Helps to Improve SEI Maturity Rating with Automated Test Process
- Makes Regression Testing Possible at the Unit Level

What is VectorCAST?

- **Source Code Based Test System**
- **Language, Compiler and Platform Specific**
- **Allows the quick and automatic construction of complete test simulation environments necessary for component level test**
- **Provides additional utilities for the following:**
 - **Building test cases/scripts**
 - **Executing tests**
 - **Generating execution reports**
 - **Dynamic analysis (code coverage)**
 - **Static analysis (code complexity and basis path)**
 - **Automatic test case building from basis paths**
 - **MC/DC support available for DO-178B Level A**