

# IPL Testing Tools and MIL-STD-498

## Executive Summary

This paper describes how AdaTEST and Cantata++ can be used within a software development to MIL-STD-498. In particular, it shows how AdaTEST and Cantata++ can be used to meet the verification and testing requirements of the standard.

IPL is an independent software house founded in 1979 and based in Bath. IPL was accredited to ISO9001 in 1988, and gained TickIT accreditation in 1991. Both AdaTEST and Cantata++ have been produced to these standards.

## Copyright

This document is the copyright of IPL Information Processing Ltd. It may not be copied or distributed in any form, in whole or in part, without the prior written consent of IPL.

*IPL  
Eveleigh House  
Grove Street  
Bath  
BA1 5LR  
UK*

*Phone: +44 (0) 1225 475000  
Fax: +44 (0) 1225 444400  
email: [ipl@iplbath.com](mailto:ipl@iplbath.com)*



Certificate Number FM1589

*Last Update: 30/05/2002 14:09  
File: MIL498.DOC*

## 1. Introduction

MIL-STD-498, “Software Development and Documentation” supersedes DOD-STD-2167A and a number of other DOD standards. This has made it one of the most widely used software development standards, being mandated by the US Department of Defense for all software development and procurement. Following the pattern set by DOD-STD-2167, this will lead to its MIL-STD-498’s adoption by many other Defence procurement agencies and may lead to its use outside of the Defence industry in areas such as the aerospace and nuclear industries. An equivalent standard, jointly prepared by the Institute of Electrical and Electronic Engineers and the Electronic Industries Association, J-STD-016-1995, enables such further adoption of the principles of MIL-STD-498.

AdaTEST and Cantata++ are tools which support the dynamic testing, dynamic (coverage) analysis, and static analysis of Ada, C and C++ software. The original requirements for the tools included the suitability for testing safety critical software, although AdaTEST and Cantata++ are sufficiently flexible for the testing of any Ada, C and C++ software. This paper describes how AdaTEST and Cantata++ can be used within any MIL-STD-498 compatible *software development process*. It should be read in conjunction with the standard.

To assist in cross referencing to MIL-STD-498, key items identified by the standard are shown in *italics* in the text of this paper.

The description consists of two parts. Section 2 gives a general description of AdaTEST and Cantata++. Section 3 reviews testing implications of MIL-STD-498 and how AdaTEST and Cantata++ fit into the associated *software development process*.

## 2. A General Description of AdaTEST and Cantata++

AdaTEST and Cantata++ support the analysis and testing of Ada, C, and C++ software at all levels, from testing individual *units* of software through to testing complete *Computer Software Configuration Items* (CSCIs). The facilities provided by AdaTEST and Cantata++ are summarised in Table 1.

Testing with AdaTEST and Cantata++ is centred on a dynamic test harness. The test harness can be used to support testing at all levels, from module test through to full integration testing, in both host and target environments. In the host environment tests can either be run directly or under the control of a debugger. In the target environment, tests can be run directly, under an emulator, or under a debugger. The AdaTEST and Cantata++ simulation facilities enable input/output devices to be simulated in the host environment.

*Test procedures* are implemented as AdaTEST or Cantata++ structured test scripts. Test scripts for both functional testing and structural testing are written in the respective application language, Ada, C or C++, and can be produced independently from the code of the application. Test scripts are fully portable between host and target environments. Test scripts may be written directly by the user, or can be generated automatically by AdaTEST or Cantata++ from a set of test case definitions. Test case

definitions can in turn either be written directly by the user, or can be imported from a CASE tool.

Testing		Analysis	
Test Preparation	Dynamic Testing	Static Analysis	Dynamic Analysis
<ul style="list-style-type: none"> <li>• Scripts in Ada, C or C++</li> <li>• Script generation from test definition</li> <li>• Test definition</li> <li>• Generation from CASE tool</li> </ul>	<ul style="list-style-type: none"> <li>• Test execution</li> <li>• Result collection</li> <li>• Result verification</li> <li>• Timing analysis</li> <li>• Stub simulation</li> <li>• Host testing</li> <li>• Target testing</li> </ul>	<ul style="list-style-type: none"> <li>• Metrics: <ul style="list-style-type: none"> <li>Counts</li> <li>Complexity</li> </ul> </li> <li>• Flowgraphs</li> <li>• Structure</li> <li>• Dataflow<sup>*1</sup></li> <li>• Instrumentation</li> </ul>	<ul style="list-style-type: none"> <li>• Coverage: <ul style="list-style-type: none"> <li>Statements</li> <li>Decisions</li> <li>Conditions</li> <li>MCDC<sup>*2</sup></li> <li>Calls</li> <li>Call-Pairs<sup>*1</sup></li> <li>Exceptions<sup>*2</sup></li> </ul> </li> <li>• Trace</li> <li>• Assertions</li> <li>• Paths</li> </ul>

**Table 1 - AdaTEST and Cantata++ Facilities**

\*1: Cantata++ only \*2: AdaTEST only

AdaTEST and Cantata++ static analysis analyses source code and also instruments the code in preparation for later dynamic analysis. The results of static analysis are reported in a list file and can be made available to the test script through the instrumented code. Static analysis results can also be exported to spreadsheets and databases.

AdaTEST and Cantata++ dynamic analysis provides a number of analysis commands which can be incorporated into test scripts. During test execution these commands are used to gather test coverage data, to trace execution, to verify data flow, and to report on and check the results of static analysis and dynamic analysis.

When a test script is executed, AdaTEST and Cantata++ produce a *test report* giving a detailed commentary on the execution of the tests, followed by an overall pass/fail summary at the end of the report. The anticipated outcomes of static analysis, dynamic analysis and dynamic testing can thus be specified in a single test script and automatically checked against actual outcomes.

To assist configuration management, all AdaTEST and Cantata++ outputs include date and time information. The reports output by AdaTEST and Cantata++ would typically be incorporated into test logs, review logs, safety logs etc.

Instrumented code is not a compulsory part of testing with AdaTEST and Cantata++. The AdaTEST and Cantata++ test harnesses can be used in a “non-analysis” mode with un-instrumented code.

AdaTEST and Cantata++ have been developed according to the IPL Quality Management System (QMS), which has been accredited to ISO 9001 and TickIT. The IPL QMS does not map directly onto MIL-STD-498, having originated from the requirements of the British Ministry of Defence, but it does fulfil the general objectives of MIL-STD-498.

Both AdaTEST and Cantata++ are in widespread use, providing additional confidence in the tool's integrity. Clients and Certification Authorities wishing to audit the development and continuing maintenance of the products may do so by arrangement with IPL.

### 3. Testing Implications of MIL-STD-498

#### 3.1. General Requirements and the Software Development Process

*Ref: 4.1, 4.2, 5.14, 5.15, 5.16, 5.17.*

MIL-STD-498 does not prescribe a *software development process*; it requires that developers have a documented process which is consistent with contract requirements and constraints placed by the standard.

Major activities which the software development process is required to cover and which AdaTEST and Cantata++ can provide assistance with include:

- (a) *Software implementation and unit testing.* Unit and integration testing are the primary uses of AdaTEST and Cantata++;
- (b) *Unit integration and testing.* Unit and integration testing are the primary uses of AdaTEST and Cantata++;
- (c) *CSCI qualification testing.* AdaTEST and Cantata++ can also be used at higher levels of testing, using the tools to simulate other CSCIs and measure test coverage of a CSCI;
- (d) *CSCI/HWCI integration and testing.* When a CSCI is integrated with a HWCI, other systems can be simulated by using AdaTEST or Cantata++ scripts on the associated hardware;
- (e) *System qualification testing.* Use of AdaTEST and Cantata++ scripts to simulate other systems is also applicable to *system qualification testing*.

The standard also identified integral processes which span the software development process:

- (a) *Software configuration management.* AdaTEST and Cantata++ attach date and time information to all test outputs. Test scripts written in AdaTEST or Cantata++ can also confirm that they are being applied to the correct version of the software under test by verifying a checksum attached to each instrumented unit;
- (b) *Software product evaluation.* AdaTEST and Cantata++ test scripts can be used to implement tests for any product supplied with an API. However, coverage analysis will only be available if the associated source code is available;
- (c) *Software quality assurance.* AdaTEST and Cantata++ provide full reports on all static analysis and dynamic test executions, providing an audit trail for quality assurance;
- (d) *Corrective action.* Automated repeatability of tests built using AdaTEST and Cantata++ facilitates regression testing following *corrective action*.

MIL-STD-498 promotes *incorporating reusable software products* and *developing reusable software products*. It is in this area that one of the greatest economic advantages of tools such as AdaTEST and Cantata++ can be achieved. Reusable products can be fully tested with an AdaTEST or Cantata++ script, which can then be supplied with the product and used to validate the product in new environments and verify that any adaptation of the product has not had detrimental side effects.

Where *critical components* are being developed there will be an associated requirement for activities such as *safety assurance* or *security assurance*. Typically this will be expressed in the form of a domain specific standard, such as DO178B for airborne systems. IPL provides a range of papers relating the requirements of such standards to AdaTEST and Cantata++.

### 3.2. Project Planning

Ref: 5.1, 5.2.

The use of AdaTEST or Cantata++ within a project is best addressed as part of the overall planning process. The *Software Development Plan* will include statements on the *software engineering environment*, software quality, safety, security, risk management and testing strategy (which provides an introduction to the later *Software Test Plan*). It is at this point in the *software development process* that a decision to use AdaTEST or Cantata++ should be documented.

The *Software Test Plan* will detail the *software test environment* and elaborate on the use of AdaTEST or Cantata++ for each level of testing.

Where multiple builds are planned, test planning must address each build. During the testing of later builds the *Software Test Plan* has to address regression testing of functionality inherited from preceding builds. Building tests using AdaTEST or Cantata++ provides automated repeatability of all tests, greatly facilitating regression testing.

### 3.3. Requirements Analysis and Design

Ref: 5.3, 5.4, 5.5, 5.6.

The software development process will include a number of levels of *requirements analysis*, *architectural design*, and *detailed design*:

- (a) *System requirements analysis*;
- (b) *System design* incorporating *system architectural design*;
- (c) *Software requirements analysis*;
- (d) *Software design* incorporating *CSCI architectural design* and *CSCI detailed design*.

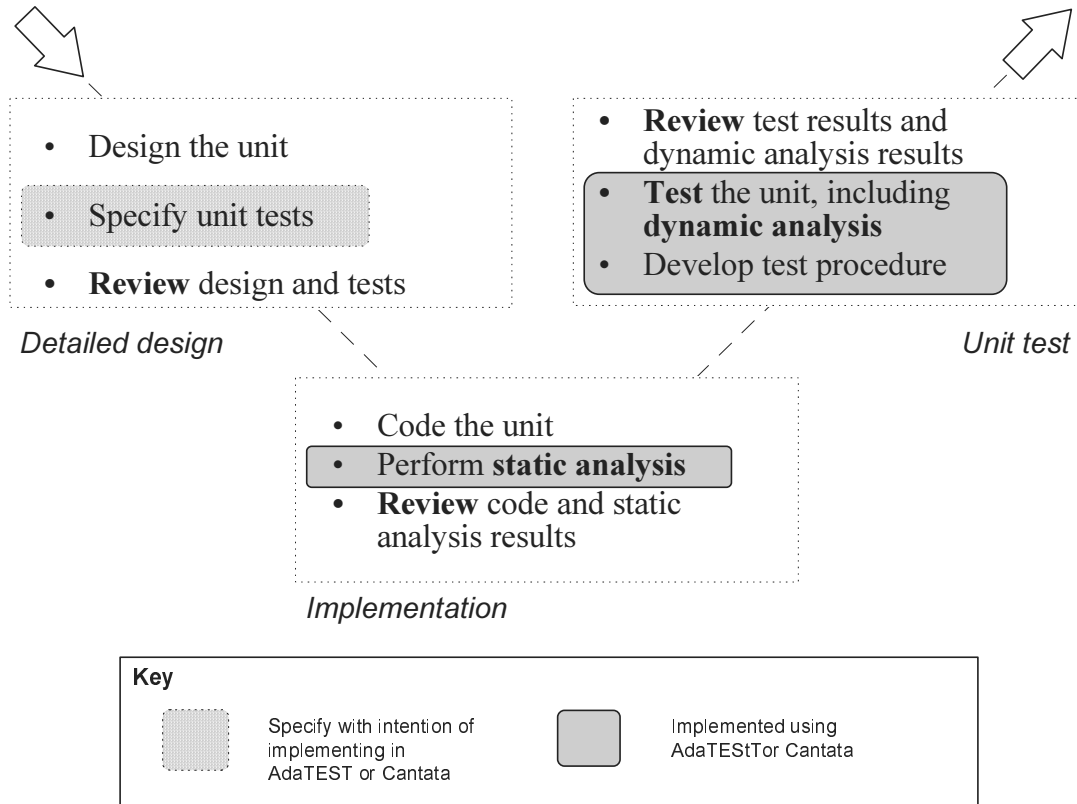
Decisions made during all levels of requirements analysis and design can greatly affect the ease with which a system and its associated software can be tested. It is important that the means by which a system will be tested is fully considered during these activities and the requirements and design structured and phrased to facilitate testing.

### 3.4. Software Implementation and Unit Testing

Ref 5.6, 5.7.

Preparation for unit testing will begin with the specification of *test cases*. These will then be developed into a detailed *test procedure* incorporating all test data. When AdaTEST or Cantata++ are used, the *test procedure* will be in the form of an AdaTEST or Cantata++ script or Test Case Definition (TCD) file.

Figure 1 illustrates the use of AdaTEST and Cantata++ during *detailed design*, *implementation* and *unit test*.



**Figure 1 - Use of AdaTEST and Cantata++ from Detailed Design to Unit Test**

When a unit requires *revision and retesting*, the associated AdaTEST or Cantata++ script can be revised and all necessary retesting conducted by recompiling, building and executing the test script.

*Test results* are output to a test results file, together with associated dynamic *analysis results* detailing automated measurements such as test coverage, traces and execution statistics.

Test scripts and test results files should be held in the appropriate *software development files*.

### 3.5. Unit Integration and Testing

Ref: 5.8.

The use of AdaTEST and Cantata++ during unit integration and testing is very similar to the use of the products during unit testing. The only difference is that test scripts are

written and test executables built to test groups of tested units progressively until an entire CSCI has been integrated.

### 3.6. Higher levels of Testing

*Ref: 5.9, 5.10, 5.11.*

During *CSCI qualification testing* AdaTEST and Cantata++ can be used to simulate other CSCIs and measure test coverage of a CSCI. Upon progressing to *CSCI/HWCI integration and testing*, other systems can be simulated by using AdaTEST or Cantata++ scripts on the associated hardware. The use of AdaTEST and Cantata++ scripts to simulate other systems is also applicable to *system qualification testing*.

The IPL paper “Host-Target Testing” provides guidance on integration of software in both a host development environment and a target execution environment.

Where *independence* is required, AdaTEST and Cantata++ scripts can be developed by separate teams or even organisations against a published interface for the software under test.

## 4. Tool Integrity and Development Standards

The integrity of the toolsets used in the development of any software is a significant consideration, especially where there are safety or security implications. AdaTEST and Cantata++ are required to support the development of all standards of Ada, C, and C++ software, including safety and security critical software. The development of AdaTEST and Cantata++ followed IPL's normal ISO9001/TickIT development standards with some additional high integrity measures for certain parts of the products.

Key points of the development and ongoing maintenance are:

- (a) The IPL quality management system, accredited to ISO 9001 and TickIT;
- (b) The IPL Software Code of Practice, forming part of the Quality Management System;
- (c) Hazard analysis and the maintenance of a hazard log (AdaTEST);
- (d) Independent Audit (AdaTEST);
- (e) The use of a safe language subset for the core functionality. Minimal exceptions to this subset for other functionality only where absolutely necessary and justified in the hazard reporting process (AdaTEST). Tests can be built using just the core functionality;
- (f) Configuration Management;
- (g) Rigorous dynamic testing, from module level upwards.

AdaTEST and Cantata++ are in widespread use, providing additional confidence in the tools integrity through the product service history. Clients and certification authorities wishing to audit the development and continuing maintenance of the products may do so by arrangement with IPL.

## 5. Conclusion

AdaTEST and Cantata++ are well suited to the development of software to MIL-STD-498. This paper has shown that extensive use can be made of AdaTEST and Cantata++ to provide automation during the *unit testing* and *unit integration* activities within the *software development process*. The products can also be used to support higher levels of testing. It is believed that AdaTEST and Cantata++ are the only tools to offer this comprehensive functionality.

With a growing emphasis on *reusable software products*, both across defence programmes and from the commercial sector, the importance of thorough testing as a prerequisite for reuse has emerged. AdaTEST and Cantata++ provide one of the enabling technologies for such reuse.