Introduction

As software systems grow more complex, the need to test software thoroughly at every stage of the development lifecycle, is becoming ever more apparent. Software project failures continue to show the consequences of mediocre or non-existent testing.

Most software developers now appreciate the need to test software thoroughly and systematically, but there is still a problem; testing is expensive. So despite a virtuous desire to improve quality, developers are still faced with the dilemma of choosing between quality and cost.

AdaTEST 95 now brings professional testing within the reach of all Ada 95 software developers by greatly increasing the productivity of the testing process. AdaTEST 95 has been specifically designed to offer developers a high-productivity solution to the verification of Ada 95 software in a unique, integrated toolset.

AdaTEST 95 provides facilities for:

- **Dynamic Testing**: executing the software under test in order to verify its compliance with requirements -- *does it do what it should?*
- **Coverage Analysis**: measuring the proportion of software exercised by dynamic testing -- *are the tests thorough?*
- **Static Analysis**: examining source code to assess its complexity and use of language constructs -- *is the software maintainable? Does it meet coding standards?*

The AdaTEST 95 toolset provides a wealth of facilities in these three areas.
Dynamic Testing

Dynamic testing of software during its development (unit and integration testing) has repeatedly been shown to be highly effective in finding defects. More importantly, this type of testing finds defects early in the lifecycle, minimizing the cost implications of faults. Until now, testing at these levels has been ignored by all but the most rigorous of software developers because of its cost; unit and integration testing could account for a very considerable proportion of the effort expended in a software project.

AdaTEST 95’s dynamic testing facilities offer both productivity and ease of use, allowing Ada 95 developers the best of both worlds: high quality software at a reasonable cost. AdaTEST 95 allows testing to be performed in an intuitive way making the tool exceptionally easy to use.

AdaTEST 95 provides comprehensive facilities for all forms of unit and integration testing. The tool has been specifically designed to operate in both host and target systems and so allow full portability of tests between these environments.

Coverage Analysis

As all programmers know, it is very easy to develop a software test suite that works. A more important question is “Has the software been fully exercised?”. Coverage analysis answers this question by measuring the proportion of the code that has been executed.

AdaTEST 95 coverage analysis facilities allow the user to assess the coverage of individual software units (procedures, functions, tasks and protected entries), packages, and even entire Ada systems. AdaTEST 95 provides all of the commonly quoted coverage metrics. Coverage information is reported in a form that is easy to understand, allowing the user to identify unexecuted code.

In addition to checking the usage of the program code, AdaTEST 95 allows the user to check the coverage of important data items. Using this facility it is possible for the user to automatically verify that a particular variable has held a series of required values.

Coverage results can be "built-in" to the overall test pass/fail criteria, ensuring that code is fully exercised as part of the dynamic testing process.
Static Analysis

Even software that has been thoroughly dynamically tested can have its problems. Static analysis gives the developer useful information on non-functional qualities of the source code, such as its maintainability and compliance with coding standards.

AdaTEST 95's static analysis facilities allow the user to check that software complies with coding standards and is within acceptable limits of size and complexity.

AdaTEST 95 provides a host of static analysis measures against which the user can evaluate code. These include a large number of 'common sense' metrics, such as the number of code statements, as well as measures of complexity, such as the McCabe's cyclomatic complexity metric.

Analysis data can be reported to the user in a variety of formats, but more importantly metrics can be verified as part of the overall testing process. Users can specify their own language standards and automatically ensure that all units tested are fully compliant.

Thus, users of AdaTEST 95 can define their own quality model based upon dynamic testing, coverage analysis and static analysis criteria. This model can be implemented in a single 'test script' that verifies all facets of the software unit under test. The following sections describe AdaTEST 95's dynamic testing and analysis facilities in more detail.
AdaTEST 95 - Dynamic Testing

The Testing Problem
Dynamic testing, at the conceptual level, is a generic problem. This is illustrated in the diagram below.

The first requirement of all software testing is that there must be some form of specification for the software; it is not possible to test without a model of the software's required behavior. The developer must not only implement the specification to produce executable software, but also conceive a set of test cases to properly test the software.

Given the prerequisites of the software to be tested and test cases with which to exercise it, the next problem to address is implementing the tests. What is required is a tool which will take the software to be tested and test case data, and produce test results.

Any such tool must be able to handle test cases generated using any of the popular functional (black-box) or structural (white-box) test case selection techniques. It must also be a practical tool; able to test a software unit in isolation, before the other units which it references are available to be tested. Integration testing is another important requirement; testing that any integrated set of software units performs as expected. Finally, the tool must apply best practice in testing; checking not only for expected behavior (positive testing) but also that the software does not do anything unexpected, such as corrupting memory (negative testing).

The AdaTEST 95 Solution
AdaTEST 95 solves the generic testing problem for unit and integration testing of Ada 95 software.

AdaTEST 95 tests are controlled by a 'test script' which can be either coded by the user or, in the case of Ada 83 source code, generated by the AdaTEST Test Script generator (ATS). The test script is compiled and linked with AdaTEST Harness (ATH) and the software under test. This will produce an executable program which, when run, produces a test results file.

The benefits of this approach are:
- **Productivity**: Developers can quickly produce working test software, resulting in significant productivity increases.
- **Documentation**: The tool provides documented proof of testing.
- **Repeatability**: AdaTEST 95 tests can be easily repeated by QA or other staff.
- **Maintainability**: Test Scripts are easy to understand and can be updated as the software is enhanced.
- **Quality**: Systematic testing leads to a higher quality product.
AdaTEST 95 - Dynamic Testing

The Test Harness

AdaTEST Test Harness (ATH) provides facilities to run, verify the results of, document, and repeat dynamic tests. ATH consists of a set of library directives, which are accessed from the test script. The software under test is called from the test script and the effects of the call on the software environment are checked by the ATH directives embedded in the script.

An AdaTEST 95 test script is thus simply an Ada procedure which exercises the software under test. The use of Ada for test scripts, together with a simple script syntax means that AdaTEST 95 test scripts are very easy to understand and modify.

AdaTEST Harness can test all ISO/IEC-8652:1995 Ada language constructs, including tasking and exceptions.

When an AdaTEST 95 test is run, a results file is produced detailing every step of the test and highlighting any failures. A results table is also displayed summarizing the results for the test as a whole. Most important of all, an overall statement of **test pass or fail** is provided and may also be returned to the command shell.

Test scripts are normally coded manually. However, in the case where the user is using an Ada 83 subset of the Ada language, the AdaTEST Test Script generator (ATS) can be used to assist in the generation of scripts.

Checking Values

Probably the single most important aspect of dynamic testing is checking that the outputs from the software under test are as expected. ATH verifies data using **CHECK** directives. These cause the comparison of a data item with its expected value.

Check directives are provided within ATH for all Ada 95 defined types. AdaTEST 95 has other checking facilities too: For example, for comparing blocks of memory, and for checking an 'observation', i.e. the occurrence of a physical event.

Simulating External Software

Developers often need to test software units in isolation from one another. This means that calls to external units and external data must be simulated.

AdaTEST 95 provides a sophisticated yet easy-to-use set of simulation facilities for external code and data. Calls to external subprograms, tasks and protected objects can be simulated, ensuring that they are made in the expected order and that input parameters have the correct values at each call. Output parameters and return values can be individually set. Similarly, external data areas may be simulated and checked.

Ada 95 Language Features

AdaTEST 95 provides explicit support for many of the new Ada 95 language features. These include the new categories of types: modular and decimal types, tagged types, type hierarchies, limited types and controlled types. AdaTEST 95 also supports the new tasking and synchronization constructs, protected objects, and the extensive predefined language environment including its string handling package.

Exceptions

A useful but difficult to test aspect of the Ada language is its exception facilities. AdaTEST 95 allows the user to verify that exceptions are both correctly raised and handled. AdaTEST 95 can check that expected exceptions are correctly raised and that no unexpected exceptions are propagated from the software under test. Exceptions can also be raised from within the script to be handled by the software
AdaTEST 95 - Dynamic Testing
under test.
AdaTEST 95 - Dynamic Testing

Timing Analysis

In many systems it is not only the correct functioning of software that determines its acceptability; real-time applications need to perform certain activities both correctly and within defined time constraints. AdaTEST 95 permits execution times to be recorded, using the Ada 95 real-time timing capabilities, and tests passed or failed depending on the real-time performance of the software under test.

AdaTEST 95 Results

At execution time, the progress of the test run is marked by output from the ATH directives. Any event which can be classed as 'unexpected' is highlighted with >>, and an appropriate diagnostic message. For example, a check which fails will be marked with >>>FAILED, and a diagnostic will give both the actual and expected values of the item being checked.

The box below contains a typical section of a ATH output file:

At the end, a table of results gives the complete statistics for the overall test.

This table of results shows that the test has failed due to a failed data check.

The box below contains a typical section of a ATH output file:

At the end, a table of results gives the complete statistics for the overall test.
AdaTEST 95 - Dynamic Testing

Test Script Generation

The AdaTEST Test Script generator (ATS) prepares dynamic test scripts for execution with AdaTEST Test Harness (ATH). **ATS can only be used if the user enforces an Ada 83 subset of the Ada language.**

ATS takes information from a **test case definition file** (which is produced by the user or a third party tool) and generates a **test script**. The test case definition file specifies test cases for which initial conditions are established and expected results defined.

ATS simplifies and expedites the task of developing a dynamic test suite. ATS works by scanning the test case definition file and the software under test to produce the test script.

The format of the test case definition file is one of several standard formats generated by third party test case generation tools, or an easy to comprehend format which can be coded manually.

All test scripts produced by ATS feature full positive and negative data checking. ATS automatically codes check routines for user defined types and stubs for external subprograms. Test scripts are easy to understand and can be manually modified, if required.

```
   Software
     
   AdaTEST ST
     
   Test Cases

   Test Script
```
AdaTEST 95 - Analysis

AdaTEST Analysis
AdaTEST Analysis (ATA) provides the user with Coverage Analysis and Static Analysis features. The following sections describe these two techniques and how they are addressed by AdaTEST 95.

Coverage Analysis
Coverage analysis measures the proportion of software executed during dynamic testing, giving developers and users alike confidence that tests are comprehensive.

There are many types of coverage metric. The ones supported by AdaTEST 95 have been shown to be those that contribute most effectively to error detection. They are:

- **statement coverage**, which is a measure of the proportion of code statements executed during a test run.
- **decision** (or **branch**) coverage, which is a measure of the proportion of decision outcomes executed.
- **boolean operator coverage**, which checks that boolean operators (in both decision and assignment statements) are properly tested. For example, in the expression
  \[ \text{if } A > 10 \text{ and } B < 20 \ldots \]
  it is possible to test that the Boolean operator **and** is fully exercised.
- **boolean operand coverage**, which checks that boolean operands in boolean expressions have held both **TRUE** and **FALSE** values.
- **MCDC (Modified Condition Decision Coverage)**, which checks that each boolean operand can independently influence the outcome of a boolean expression. Metrics covering both the ‘unique-cause’ and ‘masking’ forms of MCDC coverage are provided.
- **exception coverage**, which measures the proportion of exception handlers within the software which have been exercised.
- **exception statement coverage**, which measures the proportion of code statements in exception handlers which have been executed.
- **call coverage**, which is a check that all expected subprograms have been called. This is especially relevant for package testing.
- **data assertion coverage**, which checks that program variables have held a series of (user defined) values during the testing process. This facility is useful in checking that data boundary values have been exercised and that loops have been adequately tested.

Static Analysis
Static Analysis, as the name suggests, involves the static inspection of the source code in order to provide an assessment of various non-functional features relating to the software. AdaTEST 95 can assist developers in two important areas:

- enforcement of **coding standards**
- objective measurement of **code complexity** and **structure**.

Coding Standards
Organizations are increasingly adopting coding standards as a means of improving software quality and maintainability. However, unless these standards can be verified in an automated way it is difficult to enforce them effectively.

AdaTEST 95 provides the developer with access to many static analysis metrics on the use of coding constructs. These metrics can be examined as part of a report, or checks on their value can be incorporated into overall test pass/fail criteria, as part of a test script. For example, users may check that no **goto** statements (or labels) are used, or that no **use** clauses are present. Specific Ada 95 metrics include the number of tagged types, type extensions and unchecked accesses.

Users can define their own static analysis metrics to check on the use of code constructs which are not otherwise measured by AdaTEST 95. For instance, these facilities can be used to restrict access to certain library routine calls.
AdaTEST Analysis

Code Complexity
As a means of increasing the maintainability of software, organizations are taking an interest in objective measurement of code complexity. Many of the recognized ‘academic’ metrics are supported by AdaTEST 95. These include:
- McCabe’s measure and Myers’ extension
- Essential McCabe’s
- Hansen’s measure of software complexity by the pair
- Halstead’s software science metrics

AdaTEST 95 also supports a number of common sense complexity metrics, such as average and maximum statement nesting level.

Using AdaTEST Analysis
AdaTEST Analysis comprises a series of components which have been designed to offer, above all, flexibility to the user. They may be used in many different ways and, if required, be extended by the user to meet client specific requirements.

AdaTEST Analysis comprises:
- an Instrumenter program, to analyze source code files and insert coverage ‘probes’ (ATI);
- an additional library of test directives, which may be incorporated into a test script (the ATA library).

AdaTEST Analysis can be used in two main ways. The first of these is as an extension to ATH, allowing the user to fully integrate dynamic testing with coverage analysis and static verification. This diagram illustrates the use of AdaTEST Analysis in this way:

The AdaTEST instrumenter is used to produce instrumented source code: that is source code containing ‘probes’ to facilitate the collection of coverage data. The AdaTEST instrumenter also produces a ‘list’ file (an annotated code listing containing the source code and a static analysis report).

The instrumented source code is compiled and linked with a test script (based upon ATH) and the ATA library. When run, the resultant executable produces test results which may include both static analysis and coverage analysis information as well as the normal dynamic test results.

The AdaTEST 95 approach is unique in that test coverage measurement can be completely integrated into the dynamic testing process and that it is possible to enforce a required level of test coverage. (No post-processing of trace files is necessary).
AdaTEST 95 - Analysis

Stand-Alone AdaTEST Analysis

AdaTEST Analysis can also be used stand-alone. In this mode, developers can use their own test software (or a third party tool) to exercise the application under test, while generating analysis reports using AdaTEST Analysis:

In this case complete coverage reports are produced directly by the application’s executable.
AdaTEST 95 - Analysis

Coverage Analysis Results
AdaTEST Analysis can produce coverage (and static) analysis results in a variety of forms. The following sections give some idea of the flexibility of the product in this respect.

Coverage Analysis Within a Test Script
When used with ATH, AdaTEST Analysis is at its most powerful. Users have complete access to all coverage analysis and static analysis results from within the test script. Checks on metrics can be made and complete analysis reports generated as part of the dynamic test results. (Static analysis results can be similarly reported and checked.)

For instance, it is possible to verify that decision or statement coverage of any preset level has been attained, or to give a complete report on any of the coverage metrics. All reports are cross-referenced to the list file generated by the instrumenter.

This example shows a simple decision coverage report, indicating execution of each decision:

```
++ Analysis Report on DISPLAY ++
Filename           display.ada
Instrumented on    Thu Mar 17 11:20:08 1994
++ DECISION STATISTICS ++
Line Number     Decision     Statement                  Executions
               Type            Outcome                   
          8         if              TRUE                            2
                       FALSE                           1
          12        case-when                      1
          15        case-when                      1
          18        case-when                      0 <=
          21        case-when                      0 <=
          26      for-loop        Loop executions                 2
                      Complete iterations             8
                      Incomplete iterations           0
                      Null loops                      0
                      Normal completions            2
          34    while-loop       Loop executions                 1
                      Complete iterations             0
                      Incomplete iterations           0
                      Null loops                      1
                      Normal completions            0 <=
          40          loop        Loop executions                 1
                      Complete iterations             0
                      Incomplete iterations           0
                      Null loops                      0
                      Normal completions            0
          44      exit-when       TRUE                            1
                       FALSE                           0 <=
++ End of Report ++

CHECK_ANALYSIS ( DISPLAY,
                   DECISION_COVERAGE,
                   Lower Limit   100.00,
                   Upper Limit   100.01 );  >>FAILED
                  VALUE   53.85 %
```

Stand-Alone Coverage Analysis
When used in stand-alone mode, AdaTEST coverage analysis can be used to provide coverage reporting from units, modules or complete application programs which are independently executed, i.e. not under the control of ATH. The full range of coverage reports are still available in this mode of operation.
Ada TEST 95 - Analysis

Static Analysis Results

Static analysis results can be accessed in a number of ways:

- as part of the instrumenter list file
- as a comma separated value (.CSV) file, which can be exported to spreadsheet or database packages
- from within the ATH test script, where they may be checked

An example of each of these formats is shown on the following pages. This report is a simple static analysis instrumenter listing:

Path Verification and Trace

Another feature of AdaTEST Analysis which is of interest to high-integrity software developers is path verification. This facility allows the user to verify that a particular path through the code has been followed during dynamic testing.

AdaTEST 95's path verification feature indicates the point at which the software first deviates from its expected behavior and, therefore, provides a powerful debugging aid. It can also be used to perform 'feasible path' coverage analysis for the highest levels of quality assurance, such as safety critical software development.

In addition, AdaTEST Analysis can also produce a complete trace listing showing executed statements, decisions (and decision outcome) and entry points.

Summary

AdaTEST Analysis can provide:

- a static analysis report on source code.
- a static analysis report as a .CSV file for import into any spreadsheet package (e.g.: Microsoft Excel®).
- coverage and static analysis reports as part of the dynamic testing process.
- verification of coverage and static analysis metrics as part of the dynamic testing process.
- stand alone coverage analysis reports.
- a trace of the execution of the software under test.
- verification that the expected path through the software has been
followed
Developed as an extension to AdaTEST 95, VisualMetrics is an easy-to-use tool that provides state-of-the-art graphical presentations of the metrics data gathered on your software. VisualMetrics transforms AdaTEST 95 .CSV files into charts so that you can quickly document or present your findings. VisualMetrics is a powerful tool that provides sorting, filtering, user definable reports, over 20 different graphing options, in color or black and white.

**Bar Graphs**

- **Source/Comment/Blank Lines**

**Scatter Diagrams**

- **Lines of Code**

**Customize!**

Use VisualMetrics to choose just the metrics you want to see. Filter out values within those metrics that you do not want to see so you can zoom in on problem areas.

**3D Graphs**

- **Cyclomatic Complexity > 3**

**Easy-to-use**

VisualMetrics supports all the metrics produced in AdaTEST 95 so there are over 100 metrics available to build custom reports from. VisualMetrics is an easy-to-use tool that provides sorting, filtering, user definable reports, over 20 different graphing options, in color or black and white.
AdaTEST 95 - Summary

**AdaTEST 95 - Benefits**

- High productivity in testing
- Simpler and more reliable testing
- Full visibility of testing activities
- Repeatability of tests
- Maintainability of tests
- Documented proof of testing
- One tool to support all testing activities
- Increased confidence in predicted timescales and budget
- Improved product quality

**AdaTEST 95 - Quality**

Quality is of particular importance in software verification tools. There is little point in testing with a tool that has been produced to standards that are lower than the software being tested. For this reason AdaTEST 95 has been produced to the highest practical standards under the control of IPL's rigorous ISO9001/TickIT Quality Management System. The product is itself fully tested at the unit, task and system level, and is used by IPL for all in-house Ada software development.

**AdaTEST 95 - Support**

IPL and QCS provide full warranty/maintenance service for AdaTEST 95. This comprises telephone support as well as automatic product update service. We can also provide full training, and on-site assistance during the early stages of using the product.

If required, we can provide consultancy on the use of the product and its adoption within the framework of a Quality Management System. IPL is a software house with many consultants who specialize in software quality issues. The company has experience of developing software to many diverse standards, including high integrity standards.

**AdaTEST 95 - Features Summary**

- Dynamic Testing
- Coverage Analysis
- Static Analysis
- Host and Target Testing
- High Productivity
AdaTEST 95 - Your Questions Answered

Q: On what platforms and in what environments can AdaTEST 95 be used?
A: AdaTEST 95 is available for any Ada 95 environment.

Q: Is it possible to take an AdaTEST 95 test script which was originally run on a host machine, and rebuild it to run on a target or under simulation?
A: AdaTEST 95 can be ported to any environment with an Ada 95 compiler. In a cross-development environment this means that test scripts can be recompiled and tests rerun on the target without change.

Q: What if AdaTEST 95 does not provide the functionality I require?
A: AdaTEST 95 is an open and very flexible product. This means that if a facility you require is not supported by the tool, you can almost certainly build it for yourself. Examples of this are: special check procedures, hardware or operating system interface routines, output of results to database packages etc.

Q: We use a specific development environment / CASE design tool. Can AdaTEST 95 be integrated with the other tools that we use?
A: AdaTEST 95 can be integrated with CASE tools and development environments.

Q: I am developing my application to a recognized standard. How can AdaTEST 95 help me?
A: AdaTEST 95 can be used to fulfil the testing requirements of most software development standards including: ISO9001, the SEI Capability Maturity Model, IEC 1508, Mil Std 498/DOD 2167A, DO-178B, EN50128, ESA PSS-05, IEC 880, MISRA and Def Stan 00-55. See our papers on this subject. We would be happy to review your development standards and advise on how AdaTEST 95 may be able to help you.

Q: I am using Ada 83, can AdaTEST 95 help me?
A: Yes it can. Both AdaTEST 95 and its sister product AdaTEST can be used to test Ada 83 code. AdaTEST was IPL's original Ada testing tool and has gained wide acceptance amongst Ada developers. Contact us for a list of users of IPL's software testing products.
AdaTEST 95 - Want To Know More?

For further information please contact:

US Distributor: Quality Checked Software, Ltd.
PO Box 6656
Beaverton, Oregon 97007

Phone: 503-645-5610
Email: info@qcsltd.com
Web: www.qcsltd.com

Manufacturer: IPL, Software Products Group
Eveleigh House
Grove Street
Bath BA1 5LR UK

Phone: +44-1225-475000
Internet: tools@iplbath.com
Web: www.iplbath.com

IPL

Certificate Number FM1589

Updated 12/16/98
All trademarks acknowledged.
AdaTEST® is the registered trademark of IPL Information Processing Limited.
This document © 1996-97 IPL Information Processing Limited. Unauthorized copying and distribution prohibited.