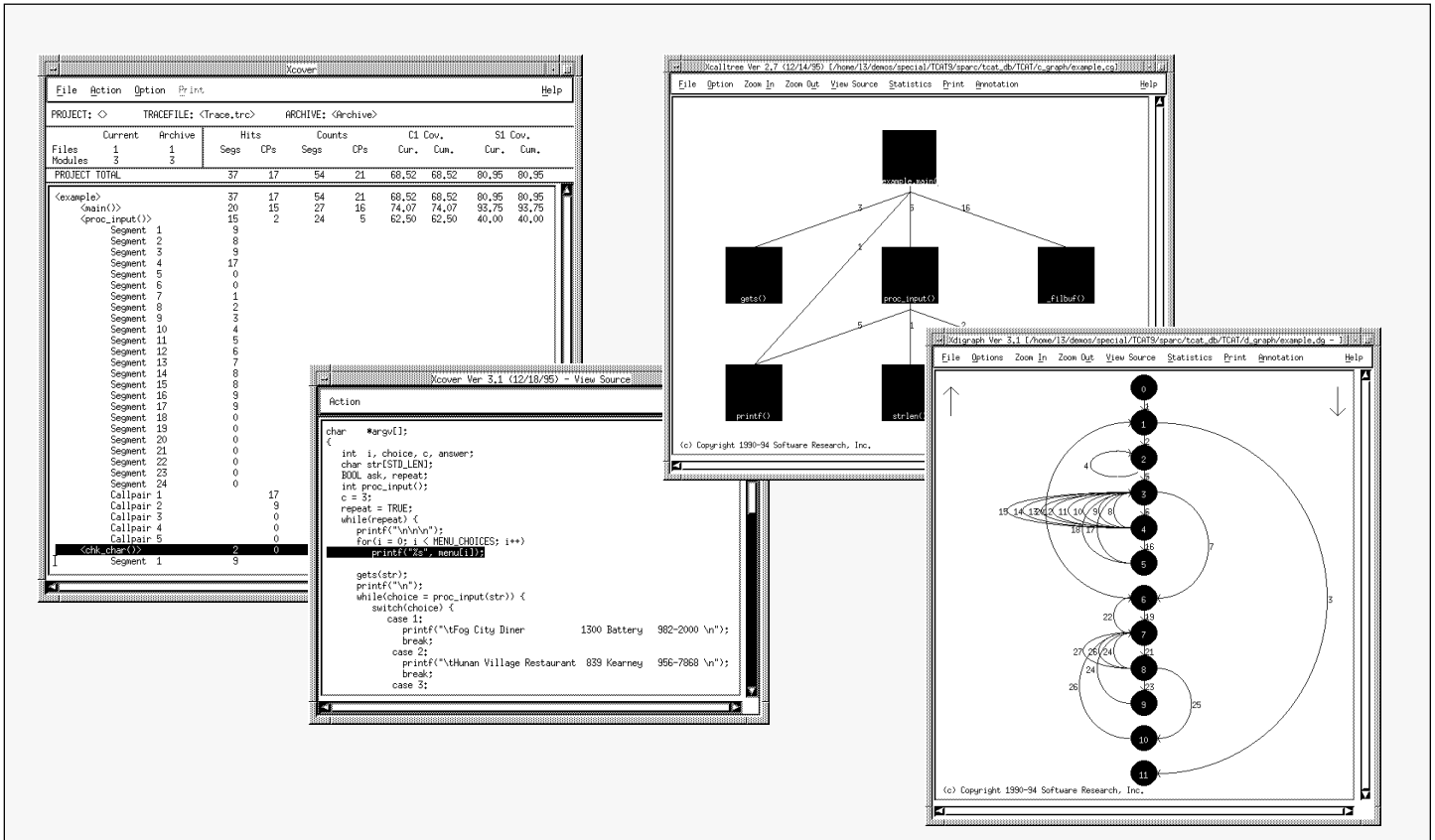# TCAT C/C++

TEST COVERAGE ANALYSIS TOOL

TCAT generates coverage reports (upper left), directed graphs (upper right), and calltrees (lower right). From the calltree (upper right) or directed graph (lower right), source code associated with the node or call pair can be displayed (lower left).

PRODUCT DESCRIPTION

STW/Coverage™'s **TCAT**™ **Version 9** coverage analysis tool uses a new consolidated C and C++ instrumentor to measure logical path and branch completeness at the individual function level (*C1*) and to determine at the system level if all of the interfaces have been exercised (*S1*). This new compiler-based technology combines the functionality of previous releases of **TCAT** and **S-TCAT**.

TCAT measures the number of times logical branches have been exercised for both True and False conditions using its *C1* metric. The *S1* measurement expresses test effectiveness as the percentage of function-calls (every function, not just one per function) exercised in a program by a set of tests, relative to the number of such function-calls in the system. *S1* identifies system interface errors by tracing exercises of the system's caller-callee relationships. **TCAT** instruments the program, placing markers at each logical branch and function-call. When test cases have been run against the instrumented code, the *C1* and *S1* metrics collect data on the markers and store it in a trace file, from which **TCAT** extracts information to create tabular coverage reports that show which code remains untested, or frequently tested, and which test cases duplicate coverage. **TCAT** also creates an archive file that stores all cumulative test information.

The instrumentation process creates directed graphs and call trees that depict a module's control-flow structure and the caller-callee structure of a program, respectively. These graphical representations are one of **TCAT**'s fundamental advantages: they are easy to interpret and give the user a quick overview of the results prior to looking at detailed reports.

The directed graphs allow the user to view a logical branch's level of coverage with **TCAT**'s unique annotation feature, and then to display and highlight the source code for that logical branch. The directed graphs can be used to identify unexercised code and to study the architecture of a module to aid in re-designing new test cases for unexercised branches. Call-trees aid the user in understanding code because they organize and identify a program's modules and connecting function-calls in a hierarchy.

INSTRUMENTATION

- *Selective Instrumentation* — Allows for the instrumentation of isolated modules or exclusion of modules that have already reached a certain level of coverage.

- *Make File Support* — Easily integrates instrumentation steps into user's existing make file.

- *Control Structures* — Recognizes and processes all control structures.

RUNTIME SUPPORT

- *Cross-Development Environments* — Performs coverage in cross, remote and embedded development environments.

- *Multiple Concurrent Processes* — Supports multi-process programs.

- *In-Memory Reduction* — Optionally accumulates trace records in memory instead of writing them to a disk.

COVERAGE REPORTS

- *Test Results* — Reports past, current and cumulative test results.

- *Report Types* — Provides hit, not hit, newly hit, newly missed, linear and logarithmic histogram reports.

- *Report Flexibility* — Provides tabular and graphic reports.

ANALYSIS THROUGH DISPLAYS

- *Directed Graphs* — Uses graphic and color overlays to diagram a module's logical branch relationship between decision points (nodes).

- *Call Trees* — Uses graphic and color overlays to diagram a program's module dependencies.

- *Annotation* — Uses different color overlays based on lower and upper threshold values that indicate if a logical branch or call pair has been unexercised or has been heavily executed.

- *Source Viewing* — Allows navigation from a directed graph or call tree to the associated source code.

- *Statistics* — Provides call-tree data which iclude the number of modules, recursive modules, function-calls and tree depth

DISPLAY OPTIONS

- *Directed Graph Options* — Sets node shape, size and length-to-width ratio, as well as logical branch ellipse eccentricity.

- *Calltree Options* — Establishes a calltree's module size and length-to-width ratio.

- *Zoom* — Sets the zoom scale of a directed graph or calltree , so the user can focus in on areas of interest.

- *Print Option* — Prints calltrees and directed graphs to Postscript output.

ANNOTATION OPTIONS

- *Low-level Color* — Sets the color of the logical branches and call paris that are exercised less than a lower threshold value.

- *Mid-level Color* — Sets the color of the logical branches and call pairs that are exercised between upper and lower threshold values.

- *High-level Color* — Sets the color of the logical branches and call pairs that are exercised more than a upper threshold value.

- User Defined Annotation — Along with the annotation options provided with TCAT, users may define their own annotation for calltrees and directed graphs.

MULTIPLE INTERFACES

- Graphical User Interface (GUI)
- Command Lines
- Menu

SUPPORTED PLATFORMS

- Sun SPARC
- x86 Solaris
- SCO
- SGI
- IBM RS/6000
- HP 9000 - 700/800
- DEC Alpha

TECHNICAL SUPPORT

- Telephone hot-line assistance for installation and technical questions is available.

- Maintenance contracts provide continuing product support and upgrades.

PLATFORM AVAILABILITY

All product specifications and capabilities are subject to change without notice. Some features of this product may not be available on some platforms.

For more information about **TCAT** or any Software Research products, call or write:

SOFTWARE RESEARCH, INC.

1663 MISSION STREET, SUITE 400
SAN FRANCISCO, CA 94103 USA
PHONE: (415) 861-2800
TOLL FREE: (800) 942-SOFT
FAX: (415) 861-9801
E-MAIL: info@soft.com
Web Site: http://www.soft.com