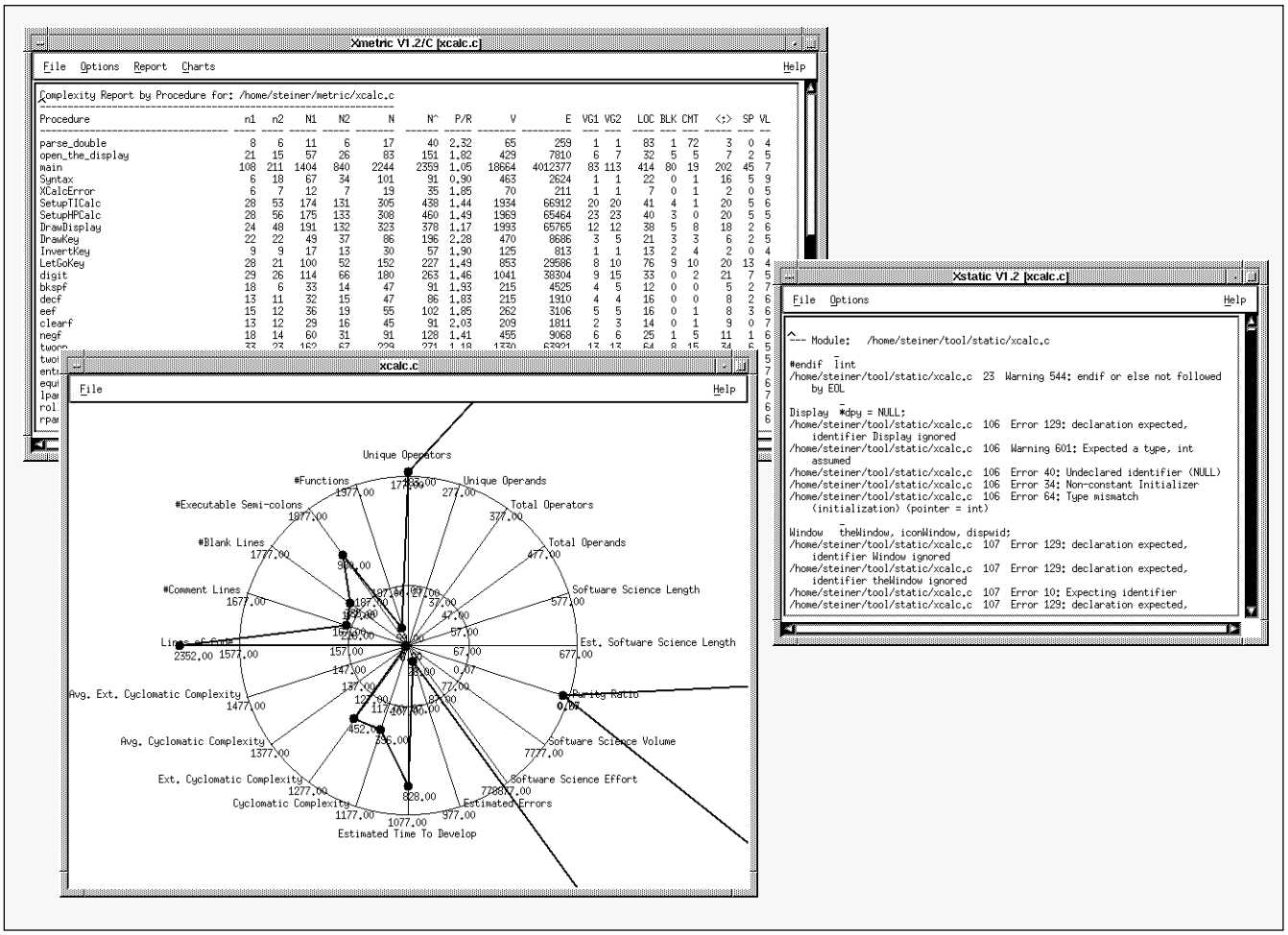
SOFTWARE RESEARCH, INC.

STW/Advisor

ADVISOR TOOL SUITE



STW/Advisor *generates software metrics for each program module, as well as an entire program, into an easy-to-understand tabular report (top left) and graphical display (bottom left). Its source code analysis capability provides a semantic and syntactic error report.*

PRODUCT DESCRIPTION

As software complexity grows, developers, testers and managers have to manage the development process and allocate limited resources. **STW/Advisor**™ analyzes source code to provide measurements and reports that enable these key decisions to be made. Three components are included in **STW/Advisor**: **METRIC**™ for quantitative analysis, **STATIC**™ for semantic and syntax analysis, and **TDGEN**™ for test data / file generation.

METRIC analyzes C, C++, Ada or FORTRAN source and calculates the Halstead Software Science Metrics to measure data complexity, the Cyclomatic Complexity Metrics to assess logic complexity, and basic size metrics, such as number of lines, comments and executable statements. User-definable thresholds can be used to establish code acceptance standards, locate error-prone functions, and help better schedule and control projects.

C programs are often the source of obscure bugs; many compilers pass bugs as legal C statements. **STATIC** handles C's unique problems by providing detailed syntax and semantic error / inconsistency reports for C programs. **STATIC** performs more detailed analyses than compilers, including locating non-portable constructs. Analysis results are presented in an easy-to-read report.

In order to make up for limited resources and more fully test applications, **TDGEN** creates additional tests from existing tests. **TDGEN** accomplishes this by mapping a template file and an input test values file into a test case, creating additional tests by substituting either random or sequential selections of values.

METRIC'S OUTPUT

- *Full Report* — Provides a set of metrics for each of the modules in a given source file.
- *Summary Report* — Provides metrics for the program as a whole.
- *Exception Report* — Lists where the code exceeds user-defined metric thresholds. It advises whether each procedure and function falls within organizational acceptable levels.
- *Kiviat Charts* — Displays the Summary report in diagram form.

SOFTWARE SCIENCE METRICS

- n_1 — Determines the number of unique operators in a program (e.g. keywords).
- n_2 — Determines the number of unique operands in a program (e.g. data objects).
- N_1 — Determines the total number of operators.
- N_2 — Determines the total number of operands.
- *Hybrid Metrics* — The above metrics derive hybrid metrics that include program length, predicted length, purity ratio, volume, and effort.

CYCLOMATIC COMPLEXITY METRICS

- *Cyclomatic Complexity* — Determines the control-flow complexity of a program, based on the number and arrangement of decision points within the code.
- *Extended Cyclomatic Complexity* — Determines the control-flow complexity of a program, based on the decision-making predicates.

SIZE METRICS

- Number of Lines of Code
- Number of Blank Lines
- Number of Comment Lines
- Number of Executable Statements

STATIC AS AN OMBUDSMAN

- *Easy Location of Problem Areas* — Message report displays the source-code line, the file name, the line number, and a brief message of problem.
- *Types of Messages* — Includes syntax and semantic messages, warning messages, information messages, and elective notes.
- *Report Customizing* — Configures reports to only significant types of messages.

MORE DETAILED THAN A COMPILER

- *Legal C Code* — Points out code that is legal C, but is probably not what was intended, such as assignment (`=`) versus test (`==`), use of bitwise operators (`&` or `|` versus `&&` or `||`), never executed code or an empty `for` loop.
- *Variable References and Function-Call Parameters* — Looks across all source files to see if variable references and function-call parameters match.
- *Non-portable Constructs* — Identifies pointer / pointer mixing, different sizes for `short`, `int` and `long`, problems when `char` is used to hold a character, signed / unsigned quantities, character set differences, and identifier length.

TEST DATA/FILE GENERATION

- *Template File* — Describes how selected test data values are to be placed in a typical test file.
- *Values File* — Indicates the actual input test values, test value ranges or test value generation rules for data descriptors that appear in the template file.
- *Generation* — **TDGEN** processes the values file by constructing a data table with field names as keys, then scans the template file for special syntax with identifier field names, and finally substitutes values from the data table associated with corresponding field names.
- *Selection of Values* — Selects test values from the data table sequentially, randomly, specifically, or calculates the total number of combinations possible.

SUPPORTED PLATFORMS

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

TECHNICAL SUPPORT

- Telephone hot-line assistance for installation and technical questions is available.
- Maintenance contracts provide continuing product support and upgrades.

For more details on **STW/Advisor**, contact:

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
http://www.soft.com