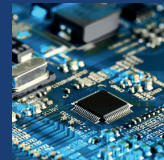


Software Testing Solutions for your Productivity and Quality



- ✓ Code Coverage
- ✓ Software Complexity Measurement
- ✓ Static Code Analysis
- ✓ Dynamic Code Analysis
- ✓ Safety-critical Embedded



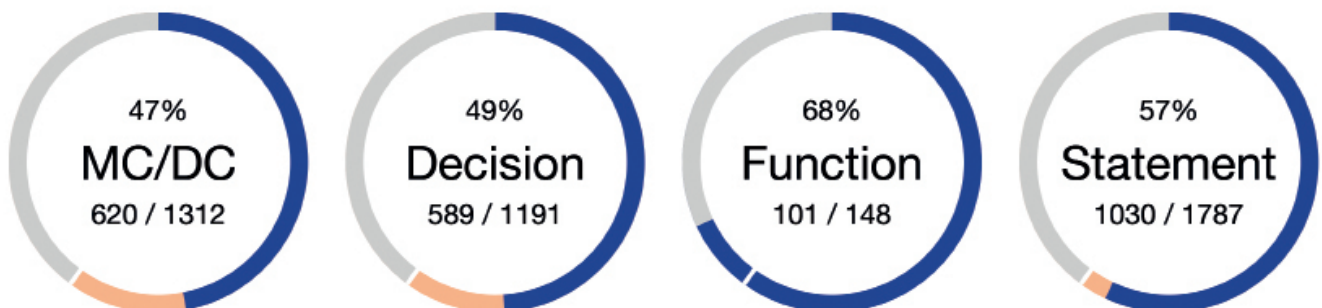
Testwell CTC++ Code Coverage Analyser

Code coverage for the highest requirements of safety standards

Testwell CTC++ analyzes which parts of your source code have been tested. Testwell CTC++ supports all coverage levels and is used by leading companies for safety-critical projects.

Coverage levels

Testwell CTC++ provides all coverage levels required by standards for safety-critical software development: function coverage, statement coverage, decision or equivalently branch coverage and modified condition/decision coverage (MC/DC). Condition and multicondition coverage can also be determined.



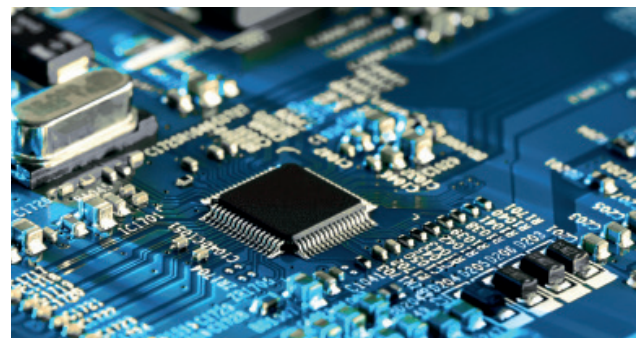
Desktop applications

- ✓ Low impact on the build process
- ✓ Scalable for large projects
- ✓ Compiler-independent
- ✓ For Windows, Linux, macOS



Embedded software

- ✓ Low memory requirements
- ✓ Testing on any target
- ✓ With any cross compiler
- ✓ Customizable runtime layer



Easy to use

- ✓ Generic build integration
- ✓ Very fast execution
- ✓ Seamless integration into many IDEs
- ✓ Ease of integration by modular architecture

Programming languages

- ✓ C, C++
- ✓ Add-On for Java

Working method

Coverage measurement is performed with Testwell CTC++ in three independent phases:



During compilation, Testwell CTC++ automatically instruments a copy of the source code by injecting measurement code. This creates an instrumented version of the program or test executable – fully automatically during the build process or on the basis of a simple, one-time build configuration.

Any type of tests can be executed as usual: Unit tests, integration tests or complete system tests. The coverage measurement data are written to a file. When performing tests on a target, this write-out is fully adjustable, e.g. the data can be transferred directly to the host computer.

In the third phase, Testwell CTC++ generates coverage reports based on the raw data. Data from different builds and different tests can be combined. A structured HTML report and any text-based exchange formats are available as output formats.

Functional safety

- ✓ Suitable for safety-critical development according to:
 - ✓ ISO 26262
 - ✓ DO 178-C
 - ✓ EN 50657 / EN 50128
 - ✓ IEC 61508
 - ✓ IEC 62304
 - ✓ IEC 62443
 - ✓ IEC 60880
 - ✓ ISO SAE 21434
 - ✓ ISO 25119 / DIN EN 16590
- ✓ Qualification support
- ✓ TÜV-certified
 - ✓ ISO 26262
 - ✓ IEC 61508
 - ✓ EN 50128
 - ✓ IEC 62304



Coverage reports

Testwell CTC++ provides a comprehensive HTML report that is adaptable to the user's needs and to the type and size of the project.

Directories and Files	MC/DC	Decision	Function	Statement
#\fluid\	2029 / 8828	1885 / 7613	327 / 723	2358 / 9682
#\src\	7602 / 24912	7208 / 21936	1254 / 2628	11951 / 28184
#\src\drivers\GDI\	620 / 1312	589 / 1191	101 / 148	1030 / 1787
#\src\drivers\WinAPI\	359 / 1079	350 / 915	70 / 137	524 / 1301
fl_WinAPI_gL_platform_init.cxx	2 / 2	2 / 2	1 / 1	1 / 1
FL_WinAPI_GL_Window_Driver.cxx	76 / 137	71 / 115	11 / 17	78 / 110
fl_WinAPI_platform_init.cxx	12 / 14	12 / 14	5 / 6	9 / 10
FL_WinAPI_Screen_Driver.cxx	84 / 173	83 / 146	20 / 23	147 / 214
FL_WinAPI_System_Driver.cxx	101 / 465	100 / 390	17 / 58	159 / 552
FL_WinAPI_Window_Driver.cxx	84 / 288	82 / 248	16 / 32	130 / 414

```

63 // turn a stored font name into a pretty name:
64 const char* FL_GDI_Graphics_Driver::get_font_name(FL_Font fnum, int* ap) {
65     FL_Fontdesc *f = fl_fonts + fnum;
66     if (!f->fontname[0]) {
67         const char* p = f->name;
68         if (!p || !*p) {if (ap) *ap = 0; return "";}
69         if (! p || ! * p)
70             1 T || _
71             2 F || T
72             3 F || F
73             MC/DC ! p: {1, 3}
74             MC/DC ! * p: {2, 3}
75         if (ap)
76             return ""
77         int type;
78         switch (*p) {
79             case 'B': type = FL_BOLD; break;
80             case 'B':
81             break
    
```

Configurable report layout

- ✓ Desired coverage levels in any combination
- ✓ Selectable report levels with drill-down: directories, source files, functions

Optional source code view

- ✓ Highlighting of executed and not executed lines
- ✓ Display of all coverage counters
- ✓ Compact visualization of complex coverage measures like MC/DC
- ✓ Visibility of missing test cases

Justification of missing coverage

Justifications can be used to record the reasons when full coverage cannot be achieved.

Testwell CTC++ derives which code parts are covered by a justification.

- ✓ Own categorization of justifications
- ✓ Recording in comments or in companion files
- ✓ Clear distinction between tested and justified code
- ✓ Recognition of over-justification



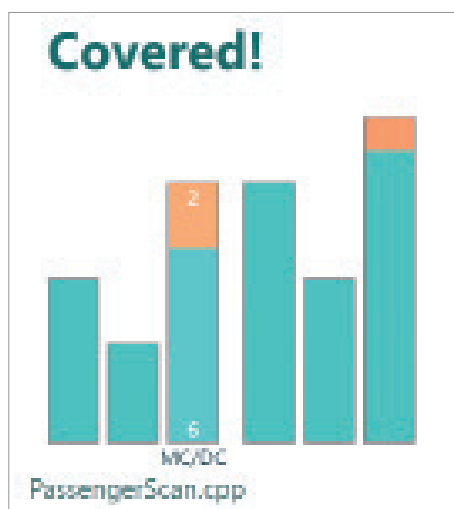
True	False	Source & Details	factory.cxx
69		469	<code>int FL_Input_Type::textstuff(int w, FL_Font& f, int& s, FL_Color& c) {</code>
31	38	470	<code>FL_Input_*myo = (FL_Input_*)(w==4 ? ((FL_Widget_Type*)factory)->o : o);</code>
31		471	<code>switch (w) {</code>
31		472	<code>case 4: {</code>
62		473	<code>case 0: f = myo->textfont(); s = myo->textsize(); c = myo->textcolor(); break;</code>
		474	<code>break</code>
3		475	<code>case 1: myo->textfont(f); break;</code>
3		476	<code>break</code>
4		477	<code>case 2: myo->textsize(s); break;</code>
4		478	<code>break</code>
0		479	<code>case 3: myo->textcolor(c); break;</code>
0		480	<code>break</code>
0		481	<code>} //CTC++ Justify DEFENSIVE: Missing Default is impossible to test here.</code>
69		482	<code>return 1;</code>
		483	<code>}</code>
		484	<code>FL_Menu_Item *subtypes() {return 0;}</code>

Coverage data in any form

Create template-based reports in any form. With a simple template language for data export, Testwell CTC++ supports structured reports such as HTML reports as well as the export of single text files.

	A	B	C	D
1			MC/DC	
2	Function	Hits	Total	Ratio
3	hasAdmission	5	5	75%
4	calcPrice	9	10	90%
5	main	5	5	100%
6	TicketApp::showInstruction	3	6	50%
7	TicketApp::switch2BatchMode	2	2	100%
8	TicketApp::askInput	3	4	75%
9	TicketApp::reportPrice	3	4	75%

Classic exchange formats like CSV, XML, JSON



Overall result as a badge or on dashboards

```

1  # Coverage Report: Coaster as Rankdown
2
3  This report was generated at 2024-03-04 09:44:08 using
4  | 'ctcreport -Template example_rankdown -- coverage.ed'
5
6  ## Coverage in Total
7  - 54% MC/DC (48 / 88)
8  - 85% Statement (69 / 79)
9
10 ## Coverage per Source File:
11 ### Directory: C:\Coaster-Code\
12 ### PassengerScan.cpp
13 - 75% MC/DC (6 / 8)
14 - 100% Statement (3 / 3)
15
16 ### PriceCalculation.cpp
17 - 90% MC/DC (9 / 10)
18 - 100% Statement (5 / 5)
    
```

Text reports, e.g. in Markdown, for easy archiving and management in a repository

Imagix 4D

For visualization and verification of programs
Easily understand and evaluate unknown source code

Imagix 4D is a tool to understand, document and improve complex, third party or legacy source code written in **C, C++ and Java**. Imagix 4D automates the analysis of control flow and dependencies. It detects problems in data usage and task interactions. With Imagix 4D you increase productivity, improve quality, and reduce risk.

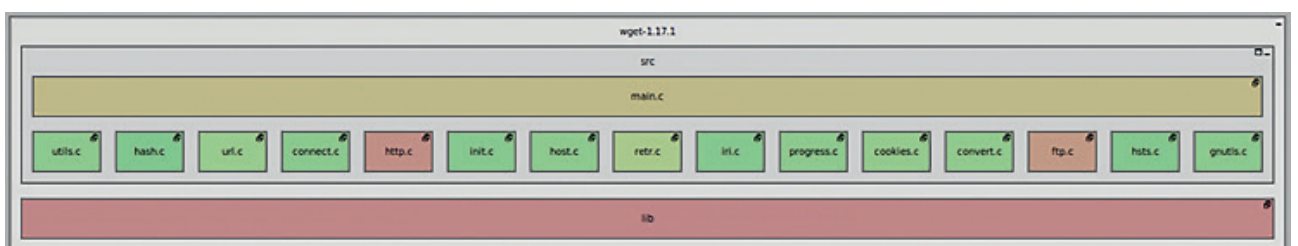
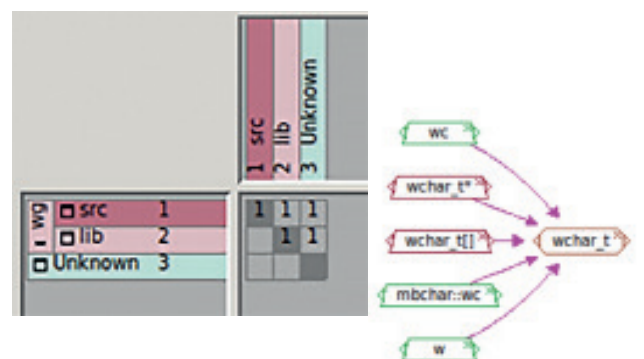
Improve quality by uncovering bugs and vulnerabilities ✓



- ✓ Automated checkers find anomalies in the sourcecode
- ✓ Over 100 metrics (including McCabe Cyclomatic and Essential Complexity, Halstead Complexity, Maintainability Index, HIS Metrics etc.) to identify critical modules in a matter of seconds
- ✓ Semi-automated reviews assist in performing efficient qualitative analyses compliant to CWE or to your own requirements

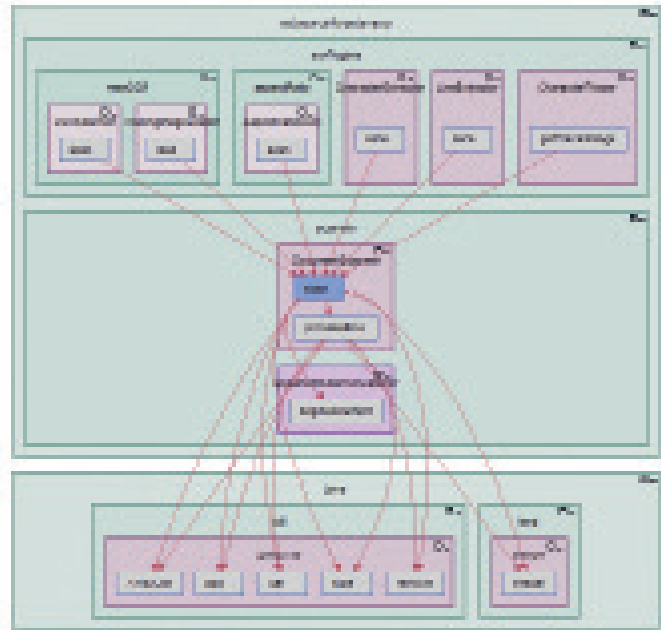
Keep control even for large projects ✓

- ✓ Meaningful diagrams provide views from a global perspective up to granular characteristics of a single data type
- ✓ Δ-Analyses enable a detailed change tracking of revisions
- ✓ Checks of the existing architecture are compliant to structural requirements based on comprehensive architecture diagrams



Many analyses and capabilities

Imagix 4D combines a variety of useful tools and capabilities in order to assess source code: Architecture diagrams, reports, delta analysis, profiler integration, visualization of Code Coverage measured by Testwell CTC++, function call diagrams, include hierarchy diagrams, bug finding, inheritance diagrams, control flow graphs, file call graphs, UML class diagrams, CWE compliance checks, metrics (variable, function, class, file, directory, architecture), design- and structure matrices, file editor, dependencies calculation tree, assignment calculation tree, diagram export, static source code analysis, symbol lists, grep based file search, concurrency analysis, include analysis.



Improve your software development cycle

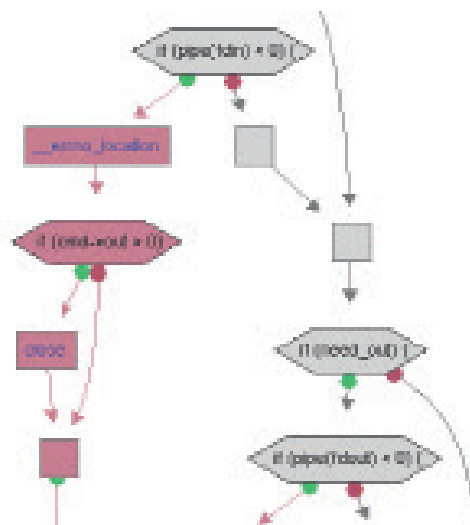
Speed up looking up information for specific symbols by efficient database queries

- ✓ Easily understand and evaluate unknown source code with Imagix 4D
- ✓ Automatically generated documents based on the present source code representing the recent state of the project



Benefit from the integration of Testwell CTC++

- ✓ Visualization of Code Coverage in control flow diagrams
- ✓ Understand the correlations between Tests and Test Coverage for a faster development of suitable test cases
- ✓ Function- and call-coverage reports complete the portfolio of Testwell CTC++



Improve your productivity and evaluate Imagix 4D now!

CodeSecure CodeSonar

Static analysis of source and binary code

Static Code Analysis can be used early in the development process and uncovers critical software defects. Risks, such as dangerous security vulnerabilities, non-deterministic concurrency errors and memory leaks, can thus be minimized. Maintenance costs are reduced thanks to more readable code.

Static source code analysis

Static binary analysis

High number of tests

Floating Point warning classes

Check for compliance with coding standards

- ✓ MISRA C
- ✓ MISRA C++
- ✓ AUTOSAR C++
- ✓ And many more

Taint Data Tracking

Metrics

- ✓ Cyclomatic complexity
- ✓ Halstead metrics
- ✓ Watson and McCabe
- ✓ HIS
- ✓ And many more

Concurrency analysis

Well documented results

High Performance

Incremental analysis

GitLab, GitHub and Docker

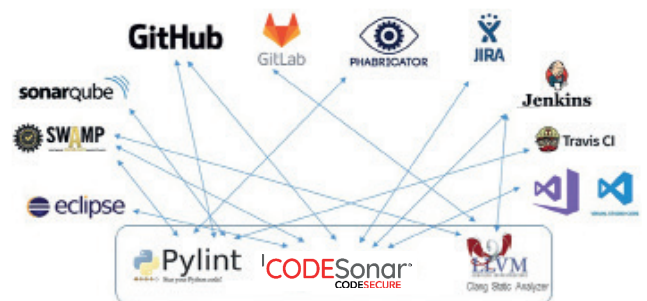
Plug-in API

Integration

- ✓ Eclipse Integration
- ✓ Microsoft Visual Studio/
Visual Studio Code Integration
- ✓ Continuous Integration
- ✓ Connection to Bug-Tracking tools

SARIF

SARIF (Static Analysis Results Interchange Format) is an open standard from OASIS (Organization for the Advancement of Structured Information Standards). CodeSonar can read SARIF data and exports analysis results in SARIF format.



BACKGROUND KNOWLEDGE

Supported programming languages

- ✓ C/C++
- ✓ Java
- ✓ C#
- ✓ Python
- ✓ Kotlin
- ✓ Go
- ✓ Rust
- ✓ .NET
- ✓ Others in preparation

Supported platforms

- ✓ Windows
- ✓ Linux
- ✓ NetBSD
- ✓ FreeBSD

Certification

CodeSonar has been certified by **exida** as a suitable tool for obtaining certifications according to:

- ✓ ISO 26262 to ASIL D, TCL3
- ✓ IEC 61508 to SIL4
- ✓ EN 50128 to SW-SIL 4

Prequalification documents help you to minimize time and costs.

Qualification

Regarding the certification of your applications, depending on the result of the classification, in some cases (e.g. according to DO 178-B/C) a qualification of CodeSonar as part of the toolchain used is necessary. The test cases required for this can be provided.

CodeSecure CodeSentry

Composition analysis of binary code

Third-party software is used extensively in software projects. The underlying components and the associated vulnerabilities are often unknown to the organization that uses them. This poses a security risk.

CodeSentry carries out an analysis of the software composition and inventories open source and third-party code in a „Software Bill of Materials“ (SBOM) component list in order to identify the vulnerabilities it contains.

Already known security vulnerabilities (N-Day Vulnerabilities) are identified by referencing CVEs (Common Vulnerability Enumeration) from the National Vulnerability Database and the Risk Based Security Database.

CodeSentry also offers „0-Day Analysis“ as a further form of analysis. This uncovers problems that have not yet been identified or published elsewhere.

CodeSentry is an API-first platform whose full functionality can be accessed by clients via a web browser. The user interface is intuitive to use and clearly laid out.

CodeSentry enables quality managers to quickly and easily assess vulnerabilities.



Testwell CMT++ CMTJava

*Testwell CMT++
Testwell CMTJava*

Software complexity analysis for C, C++, C# and Java

Testwell CMT++ and Testwell CMTJava are tools for analysing complexity of C, C++, C# and Java source code. Both tools analyse source code and inform you immediately about the current internal quality of your software product, even those with large project sizes. Avoid software erosion by achieving a good internal code quality and see how maintainability and testability will be significantly improved.

Seminars

Software and its quality determines the economic success of a company. Companies with well-established processes and structures to ensure software quality will be able to maintain and successfully expand their market position.

Do you want to keep up to date with the latest quality measures and software testing activities?

Is your company planning to set up or expand activities in software quality?

Do you want to train your employees to be able to use our software testing and analysis tools productively within the shortest time possible?

We can support you in achieving these goals.

Take advantage of our seminars and gain the knowledge you need to develop and test software efficiently and reliably.



Further information at www.verifysoft.com/en_events



Verifysoft

First-class tools worldwide since 2003!

As an independent and solidly operating company, Verifysoft supports customers worldwide with highly specialized software for the quality assurance of their software products. We develop our core products under the TESTWELL brand with a highly skilled and effective team and also offer first-class complementary tools, seminars and services.

We understand our customers and are a reliable partner for them – also in the long term. At Verifysoft,

the human being is our priority. We enjoy working in a good and fair working atmosphere for the benefit of our customers, colleagues and our social environment.

We have a long-term strategy. Customer satisfaction is more important to us than the „quick money“. For us, success is when customers and employees are satisfied.

We look forward to working with you.



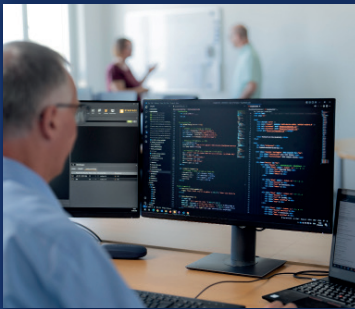
More than 750 customers all over the world



Verifysoft TECHNOLOGY

Verifysoft Technology GmbH is specialized in the development, distribution and support of software testing and analysis tools. In addition to our own Testwell tools, we also distribute complementary tools from our partners. Verifysoft was founded in 2003 in Offenburg, Germany and has since been operating successfully as an independent company in the field of software quality.

With an international team, we support more than 750 customers in over 45 countries. Our development and support staff have many years of experience in the test tool sector. Find software defects and problems as early as possible before release and guarantee the highest software quality with tools from Verifysoft Technology.



Verifysoft Technology offers seminars and trainings around the topics „Software development and test“

Our seminars: www.verifysoft.com/en_events

Further Information and Tools at: www.verifysoft.com

Get your free evaluation – Now!

© 2024 Verifysoft Technology GmbH
Testwell CTC+, Testwell CMT+, and Testwell CMTJava
are products and trademarks of Verifysoft Technology GmbH, Offenburg (Germany).

CodeSonar and SodeSentry are products and trademarks of CodeSecure, (USA)
Imagix 4D is a product and a trademark of Imagix Corp., San Luis Obispo CA (USA)

Verifysoft Technology GmbH, In der Spoeck 10 – 12, 77656 Offenburg (Germany)
Phone: +49 781 127 8118 - 0, info@verifysoft.com



Follow us

