Sales & Support
gate@slexn.com
Tel. 02-555-4847
hub.slexn.com | www.slexn.com

Verifysoft
TECHNOLOGY

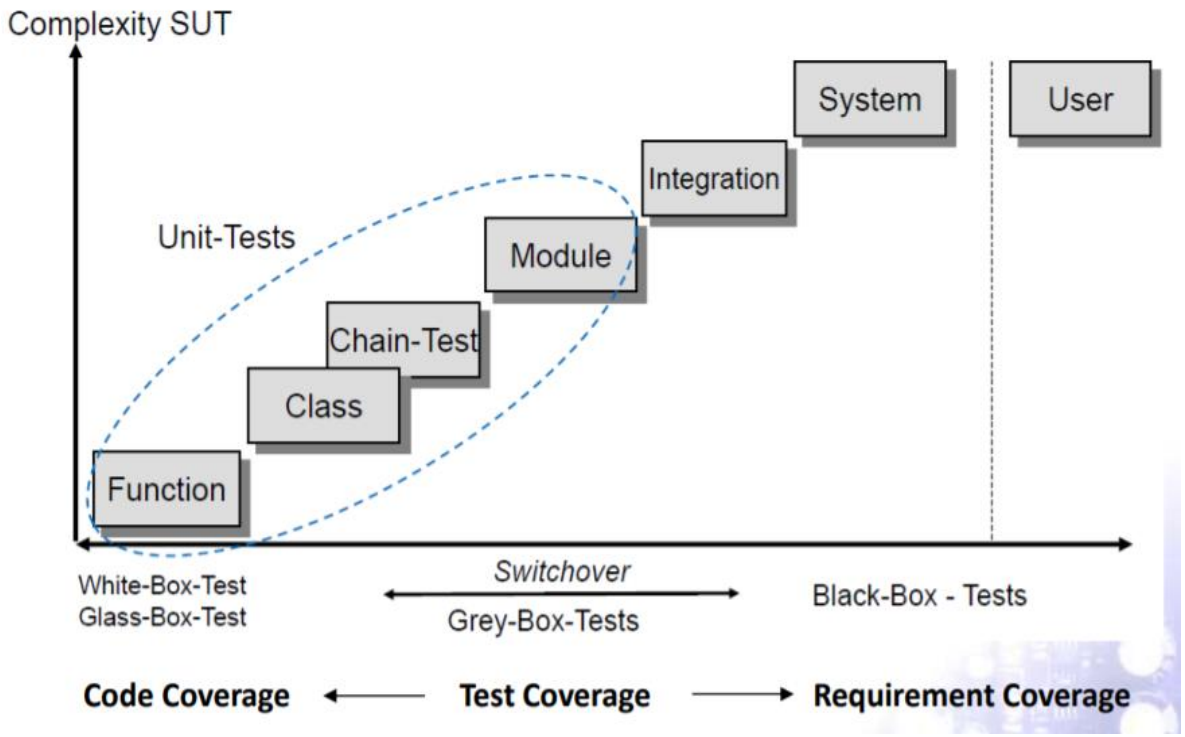
Verifysoft Technology Gmbh는 소프트웨어 테스팅 업계의 투자자 및 전문가 그룹에 의해 2003년에 설립된 회사로, 소프트웨어 테스트와 분석 도구를 공급하고 새로운 가치를 창출해냅니다.
2013년 7월 Verifysoft는 Testwell CTC ++, Testwell CMT ++ 및 CMTJava와 Testwell CTA ++의 지적 재산권을 인수하였으며, 주요 응용 분야는 자동차 및 운송, 항공 우주 및 방위, 철도, 산업 자동화, 의료 및 의료, IT/ICT 서비스 및 컨설팅, 미션 크리티컬 소프트웨어 개발, 연구 및 교육입니다.
또한 Verifysoft Technology는 파트너의 보완 기술을 제공합니다.

- **Imagix, San Luis Obispo(미국)의 코드 검사 및 아키텍처 분석 도구 Imagix 4D**
- **GrammaTech, Ithaca/New York(미국)의 정적 코드 분석 도구 CodeSonar**

HQ : Verifysoft Technology GmbH In der Spoeck 10-12
77656 Offenburg Germany

SLEXN
IT Solution Provider & Services

Why Code Coverage?



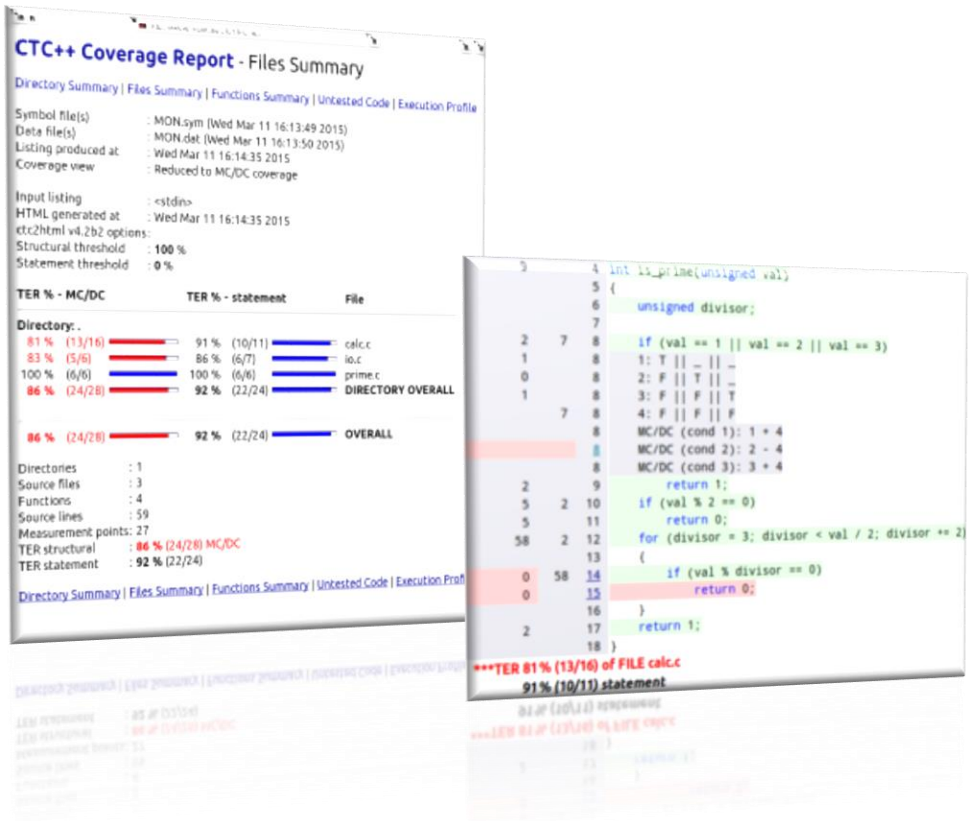
Cause-Reason-Graph	Static Testing	Back-to-Back Testing
Classification Tree Method (CTM)	Equivalent Classes Multidimensional Equivalent Classes Boundary Value Analysis Critical Value Analysis Informal Tests Smoke Tests *Basis*	CRUD
Realtime Testing		Rare Event Testing
Load Tests		Random Testing
Recovery Tests Stress Tests		Monkeytest
		Fuzzing (Fuzz Testing)
Control Flow Oriented Testing	Advanced	Evolutionary Testing
		Pairwise Testing

Established test technique for critical Embedded Systems
Test-End criterion (White-Box-Tests)
Necessary to fulfill requirements of safety standards.

Code Coverage:
shows the parts of the code which have been
executed / not executed
tested / not tested

On-Target Code Coverage for all embedded Targets
Code Coverage for C, C++, Java, and C#

Testwell CTC++는 호스트 및 모든 임베디드 대상(매우 작은 대상 포함)에서 코드 커버리지를 측정하기 위한 최고의 코드 커버리지 도구입니다. 전 세계 기업의 항공우주, 자동차, 운송, 의료 및 기타 산업에서 성공적으로 사용하고 있으며 높은 코드 범위를 달성하고 안전 표준을 준수합니다.



- ✓ 위험을 피하고 복잡한 코드의 모든 부분이 릴리스 전에 테스트되었는지 확인 할 수 있습니다.
- ✓ 자세한 코드 커버리지 정보 덕분에 Testwell CTC++는 테스트의 완전성을 분석하는 편리한 방법을 제공합니다. 이 도구는 테스트 실행 중에 응용 프로그램의 어떤 영역이 실행되었는지 식별하여 테스트 노력의 효율성을 보여줍니다. 코드의 일부 또는 전체 애플리케이션을 한 번에 분석합니다. Testwell CTC++를 사용하면 애플리케이션이 테스트되지 않은 코드로 릴리스되는 것을 방지할 수 있습니다.
- ✓ 전체 명령줄 인터페이스를 통해 기존 빌드 및 테스트 인프라와 쉽게 통합됩니다. 테스트가 진행됨에 따라 Testwell은 단순히 애플리케이션 실행을 모니터링하고 달성한 코드 적용 범위를 기록합니다. 이 정보는 요청 시 또는 테스트 실행이 끝날 때 표시됩니다.

Main Advantages of Testwell CTC++

Advantage 1: All Coverage Levels

Testwell CTC++는 기능 커버리지, 명령문 커버리지, 결정 커버리지, 분기 커버리지, 수정된 조건/결정 커버리지 등 안전 표준에서 요구하는 모든 커버리지 수준, 심지어 다중 조건 커버리지까지 포함합니다. Testwell CTC++를 사용하면 코드 적용 범위의 모든 요구 사항을 충족할 수 있습니다.

Advantage 2: All Embedded Targets

리소스에 민감한 계측은 오버헤드를 매우 낮게 유지합니다. 즉, 모든 임베디드 대상과 모든 마이크로컨트롤러에서 테스트 커버리지를 측정하기 위해 배포될 수 있습니다. 소스 코드 계측을 통해 지원되는 언어의 모든 도구 체인 내에서 Testwell CTC++를 적용할 수 있습니다.

Advantage 3: All Compilers

모든 임베디드 대상을 지원합니다. 신규 또는 추가 컴파일러에도 라이선스가 적용됩니다.

Advantage 4: For C, C++, Java, and C#

Testwell CTC++는 C에서 테스트 커버리지 측정을 위해 1989년부터 개발되었습니다. 나중에 이 도구는 C++로 확장되었습니다. 2007년부터 이 도구는 Java 및 C#에서도 작동합니다.

Advantage 5: Works also with "exotic" Language Constructs

최첨단 프로그래밍 방식을 사용하는 코드 커버리지를 측정할 수 있습니다.
Testwell 고객은 Testwell CTC++를 오프로드 차량과 비교하는 것을 좋아합니다. 이 도구는 모든 곳에서 작동하고 상황에 관계없이 해야 할 일을 정확히 수행합니다. 개발 중인 프로젝트가 창의적이고 실현 가능성이 떨어지더라도 Testwell CTC++는 그대로 따라갑니다.

Advantage 6: Integration in Your IDE

Testwell CTC++는 개발 환경에서 원활하게 실행할 수 있는 유연한 도구입니다. 또한 Testwell CTC++는 IDE에 통합될 수 있습니다.

Advantage 7: Integration in Automated Builds / Continuous Integration

Testwell CTC++는 명령줄에서 실행할 수 있습니다. 명령줄 인터페이스를 사용하면 Testwell CTC++와 자동화된 빌드를 통합할 수 있습니다.

Advantage 8: No Modification of Your Source Code

원본 소스 코드를 수정하지 않고 코드 커버리지를 측정합니다. Testwell CTC++의 계측은 빌드 프로세스 중에 소스 코드의 임시 복사본에 카운터를 자동으로 추가합니다. 원래 코드는 변경되지 않습니다. 그런 다음 테스트 실행은 원래 프로그램과 똑같은 방식으로 계측된 프로그램 버전으로 실행됩니다.

Advantage 9: Understandable Reports

Testwell CTC++는 소프트웨어를 실행할 때마다 자동으로 커버리지 보고서를 생성합니다. 마우스로 몇 번만 클릭하면 세부 정보를 확대하여 테스트 실행 중에 어떤 값이 실행되었는지 정확하게 분석할 수 있습니다. 색상으로 표시하면 전체 코드를 테스트하기 위해 아직 누락된 조건을 확인할 수 있습니다. 테스트 범위는 디렉토리, 파일 및 기능 수준에서 백분율 수치 및 다양한 세부 보기와 함께 개요로 보고서에 포함됩니다. 이러한 보고서는 텍스트, HTML, XML 및 Excel 입력 형식으로 제공됩니다.

Main Advantages of Testwell CTC++

Advantage 10: Compliant to Safety Standards

특히 항공우주, 자동차, 운송, 의료 기술 및 원자력 산업 안전 표준은 소프트웨어에 대한 엄격한 품질을 요구합니다. 위험이 높을수록 필요한 테스트 범위 수준이 높아집니다. Testwell CTC++는 코드 적용 범위와 관련된 안전 표준의 모든 요구 사항을 충족합니다.

Advantage 11: Qualification Kit

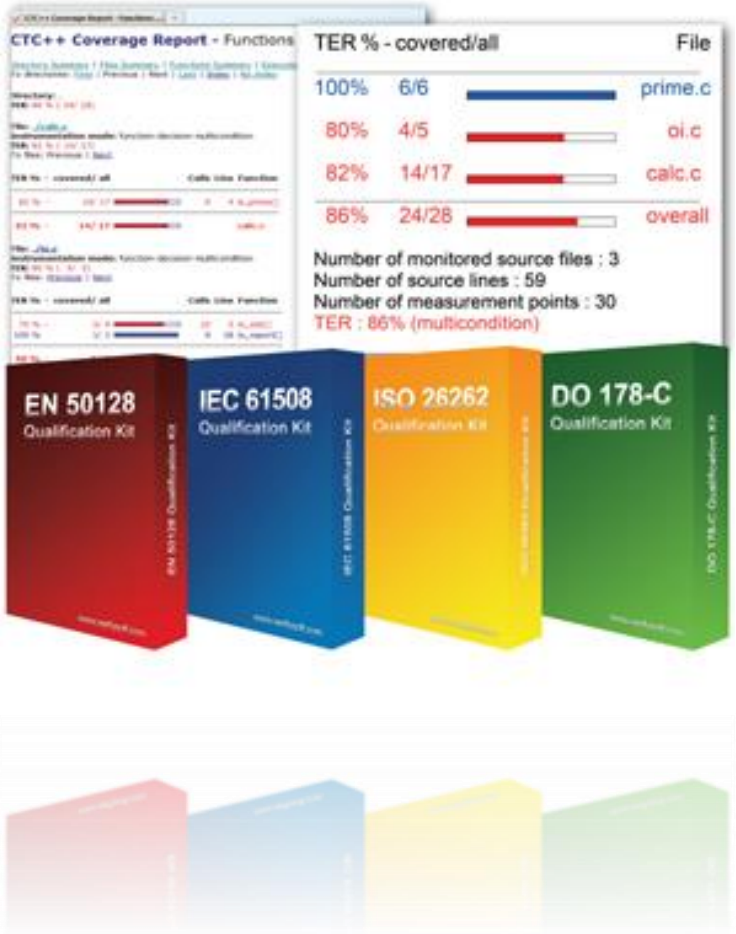
Testwell CTC++가 도구 체인 내에서 제대로 작동한다는 것을 증명하기 위해 모든 중요한 안전 표준에 대해 검증 키트를 사용할 수 있습니다.

Advantage 12: Outstanding Technical Support

우리는 고객에게 높은 생산성과 높은 품질이 가장 중요하다는 것을 이해하기 때문에 신속하고 유능한 지원에 의존할 수 있습니다. 자격을 갖춘 프로그래머 기술 직원들이 특정 질문에 대해 상시 도움을 드릴 수 있습니다.
테스트 활동을 더욱 효율적으로 만들기 위해 교육, 라이브 웨비나 및 비디오를 제공합니다.

Advantage 13: Customer Satisfaction

Testwell CTC++의 첫 번째 버전은 이미 1989년에 핀란드에서 출시되어 그 이후로 지속적으로 개선되었습니다. 전 세계적으로 수천 개의 라이선스가 사용되고 있으며 최고 품질의 소프트웨어 개발을 보장합니다. 많은 국제 그룹 및 대기업과 수많은 중소기업이 소프트웨어 개발의 품질 및 생산성 향상을 위해 Testwell CTC++를 배포합니다.
현재는(2020년 5월 기준) 39개국 600명 이상의 고객이 Testwell CTC++를 사용하고 있습니다.



Benefits

Testwell CTC++: Code Coverage for all Compilers and all Cross-Compilers

Altium Tasking classic toolsets, VX-toolset toolsets, c166, cc166, ccm16c, cc51, ccarm, ccmcs, ccpcp, cctc	Borland/Inprise/Paradigm/Codegear compilers bcc, bcc32, pcc, pcc32 (Paradigm)
Cosmic compilers cx6805, cx6808, cx6812, cxs12x, cxs12z, cxxgate, cx6811, cx6816, cx332, cxst10, cxstm8, cxst7, cxcf, cx56K, cxppc	gcc and all gcc based cross-compilers: 586-mingw32msvc-gcc, x86_64-linux-gnu-gcc, m68k-palmos-coff-gcc, tricore-gcc, arm-linux-gnueabi-gcc, arm-none-eabi-gcc, arm-none-linux-gnueabi-gcc, arm-elf-gcc...
DragonEgg	ARM: DS-5,Keil MDK-ARM
HI-Tech PICC compilers (Windows and Linux)	LLVM : clang, clang++
Microchip MPLAB C Compiler	IAR compilers and toolchains iccm16c, icc430, icc8051, iccarm, iccavr, iccavr32, icccf, icchcs12, iccmaxq, iccdspic, ..
Apple :XCode ⌘	Fujitsu/Softune: fcc896s, fcc907s, fcc911s
GHS/GreenHills/Multi	Hitachi:shc, shcpp, ch38, ccrx
Freescale/Metrowerks:mwccmcf, mwcceppc, mwccmcore, mwcc56800, mwcc56800e,..	Intel compilers (all platforms) :icc, icl, ic86, ic96
HP: HPUX CC, HP C++, aCC	Java compilers: javac, jikes, ecj, gcj, kaffe
Metaware (Both Linux and Windows host): hcac, hcarc, hcarm, mcc, ccac	Microsoft compiler: cl on host, both 32 and 64 bit, cl for Smartphones and PocketPC, csc C# compiler, vjc J# compiler
VisualDSP++: ccblkfn , cc21k, ccts	Windriver
Matlab/Simulink: lcc	Renesas :shc, shcpp, ch38, ccrx, nc, nc308, nc77, nc79A, cc32R, CS+/CubeSuite+ cc78k0, cc78k0r, cx, ca850
Pathscale pathcc/pathCC	Mono comshc, shcpp, ch38, ccrx pilers
Sun compilers: WorkShop compilers, javac	Symbian: mwccsym2, armcc, arm-none-symbianelf-g++ (GCCE)

Benefits

Support of all embedded targets and micro controllers

On-Target Code Coverage for all Embedded Targets and Microcontrollers

Testwell CTC++는 임베디드 소프트웨어의 코드 커버리지를 측정하기 위한 이상적인 도구입니다. Testwell CTC++ 테스트 커버리지 분석기는 호스트뿐만 아니라 모든 임베디드 대상 (매우 작은 대상, 제한된 메모리, 운영 체제 없음 등)에 사용할 수 있습니다. 코드 커버리지를 분석하기 위해 코드 계측을 사용합니다. 실행 카운터가 소스 코드에 추가되어 소스 코드의 일부가 실행된 횟수를 측정하고 다른 보고서에 결과를 표시합니다.

On-Target Code Coverage with Host-Target Add-on

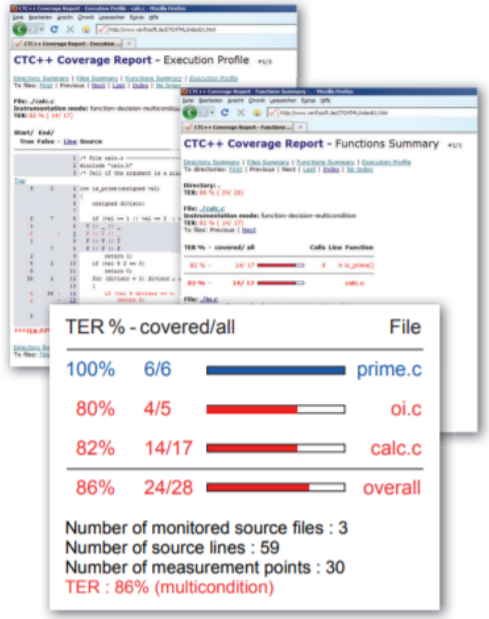
Testwell CTC++의 HOTA(호스트 대상 추가 기능)를 사용하면 대상에 대해 계측된 코드를 교차 컴파일하고 대상에서 테스트를 실행하고 커버리지 데이터를 호스트로 다시 가져오고 호스트에서 커버리지 보고서를 볼 수 있습니다. 기본 Testwell CTC++가 실행되는 호스트에서만 실행하면 효과적으로 작업할 수 있습니다. 대상 시스템 아키텍처와 해당 운영 체제는 무엇이든 될 수 있습니다. Host-Target 추가 기능은 일반적으로 그 아래에 파일 시스템이 있고 사용 가능한 RAM이 몇 킬로그램인 대상에 사용됩니다.

Bitcov Add-on for Targets with very limited Memory

대상에서 사용 가능한 메모리가 제한된 경우 Bitcov 추가 기능은 계측 오버헤드를 크게 줄여 가장 작은 대상에서 코드 적용 범위를 분석할 수 있습니다. Bitcov 추가 기능에서 카운터는 0(실행되지 않음) 및 1(실행됨)로 감소되는 반면 일반 범위 보고서에서 카운터 값은 프로브 위치에서 코드가 실행된 횟수를 확인할 수 있습니다. Testwell CTC++ 라이선스에는 임베디드 소프트웨어의 코드 커버리지 측정에 필요한 모든 추가 기능이 포함되어 있습니다. Non-embedded 분야의 소프트웨어 개발자에 한하여 Testwell CTC++ 호스트만 제공 가능합니다. (Host-Target- 및 Bitcov Add-ons가 없는 패키지)

Benefits

Analyzes for all coverage levels up to MC/DC and MCC Coverage



Call Coverage 란 ?

Function coverage : 테스트 중에 프로그램의 모든 함수가 호출되었는지 여부를 반영하면서 **Call Coverage(call pair coverage)** : 가능한 모든 호출이 실제로 테스트에서 실행 된 경우 전체 **call coverage**가 달성됩니다. 각 개별 기능에 대해 백분율 call coverage은 호출 적용 범위는 실행된 호출과 가능한 호출의 비율로 측정할 수 있습니다. ISO 26262-6 보안 표준은 통합 테스트를 위해 call coverage 또는 function coverage를 측정할 것을 권장합니다.

Testwell CTC++ analyzes for all coverage levels:

- Function Coverage
- Line Coverage
- Statement Coverage
- Decision Coverage / Branch Coverage
- Condition Coverage
- Modified Condition/Decision Coverage (MC/DC) and even for Multi condition Coverage (MCC).

Testwell CTC++ is the ideal tool to analyze the code coverage of your embedded targets and microcontrollers.

It can be used on hosts as well as on targets.

- 매우 작은 계측 오버헤드
- 모든 타겟에 대한 코드 커버리지 분석
- 가장 작은 대상에도 작동
- 모든 컴파일러/크로스 컴파일러와 함께 작동

Compliant for safety critical development

Testing Tools → Compliance with DO178-C, EN61508, EN62304, ISO26262 and other standards

Verifysoft Technology는 개발 팀이 다음과 같은 표준 및 문서의 요구 사항을 충족하도록 지원하기 위해 소프트웨어 테스트 및 분석을 위한 입증된 도구를 제공합니다.
안전에 중요한 소프트웨어의 개발에 사용되는 도구는 도구의 자격이 필요한지 여부를 결정하기 위해 분류(도구 분류)해야 합니다. 이 분류는 프로젝트에 대한 소프트웨어 도구의 영향을 고려합니다.
프로젝트에 영향을 미칠 수 있고 오작동이 즉시 나타나지 않는 도구의 경우 안전 표준은 신뢰성 증명을 요구합니다. 이 증명은 도구의 자격(도구 자격)에 의해 수행되며 도구는 항상 도구 사용자의 구체적인 개발 환경 내에서 검증되어야 합니다.

안전이 중요한 프로젝트에서 Testwell CTC ++의 인증을 단순화하기 위해 Testwell CTC++용 도구 인증 키트를 제공합니다. 인증 키트는 EN 50128, ISO 26262(자동차), DO-178C(항공) 및 IEC 61508(철도) 표준에 적합합니다.

- DO-178C all levels (DAL A, DAL B, DAL C, ...)
- ISO 26262 all levels (ASIL A, ASIL B, ASIL C, ASIL D)
- IEC 61508 all levels (SIL 1, SIL 2, SIL 3, SIL 4)
- EN 50128 all levels (SIL 0, SIL 1, SIL 2, SIL 3, SIL 4)
- IEC 60880 (Nuclear Power)
- IEC/EN 62304 (Medical)

ISO 26262-6

ASIL: Automotive Safety Integrity Level

	Methods	ASIL			
		A	B	C	D
1a	Statement coverage	++	++	+	+
1b	Branch coverage	+	++	++	++
1c	MC/DC (Modified Condition/Decision Coverage)	+	+	+	++

Table 12 (Software Unit Level), ISO 26262-6

	Methods	ASIL			
		A	B	C	D
1a	Function coverage	+	+	++	++
1b	Call coverage	+	+	++	++

Table15 (Software Architectural Level), ISO 26262-6

++ Highly recommended
+ Recommended

DIN EN 61508-3

SIL: Safety Integrity Level

Method		SIL 1	SIL 2	SIL 3	SIL 4
...	...	...	...	...	...
7a	Function Coverage	++	++	++	++
7b	Statement Coverage	+	++	++	++
7c	Branch Coverage	+	+	++	++
7d	MC/DC	+	+	+	++

Table B.2 from DIN EN 61508-3

++ Highly recommended
+ Recommended

Tool Qualification Kit

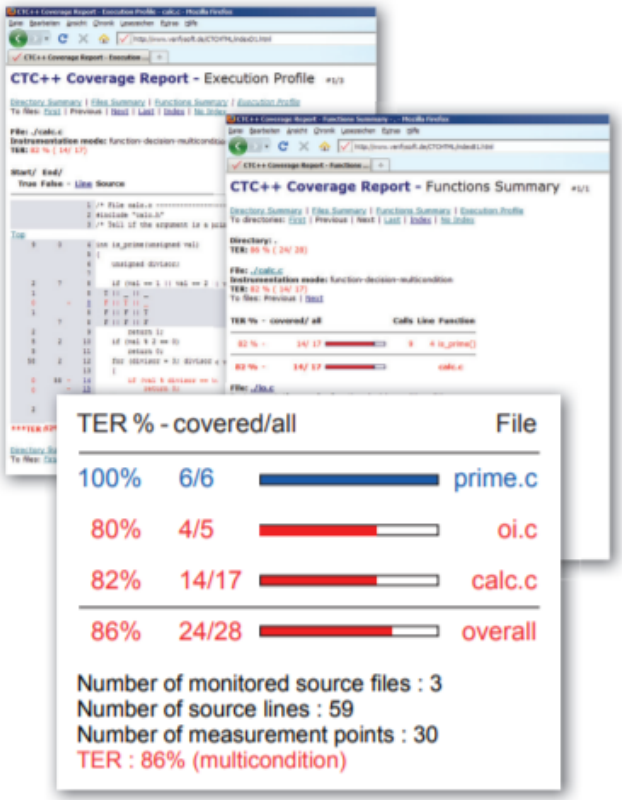
Kit 에는 Testwell CTC++가 필수 보안 표준(ISO 26262, IEC 61508, EN 50128, DO-178 포함)을 준수함을 입증하는 일반 준수 보고서가 포함되어 있습니다.
Testwell CTC++용 Qualification-Kit은 자동차 및 항공우주 분야의 중요한 회사에서 성공적으로 사용하고 대부분은 키트를 "동급 최고"로 평가합니다.
또한 도구 검증 키트는 독일 뮌헨의 Validas AG에서 개발 하였으며 Validas 도구 인증 프로세스는 TÜV에서 성공적으로 인증되었습니다.

The qualification kit for Testwell CTC++ consists of :

- Tool Classification Report
- Tool Qualification Plan/Report
- Tool Safety Manual
- Test Plan
- The test automation unit
- The test suite with test cases.
- The user manual of the qualification kit

Penetration Tests

보안을 평가하기 위해 소프트웨어나 시스템의 취약성을 식별하기 위해 침투 테스트(펜 테스트)가 수행 됩니다.
목표는 권한이 없는 당사자가 기능 및 데이터에 액세스 하는 것을 방지하는 것 입니다.



Analysis of Penetration Tests with Testwell CTC++ Code Coverage Analyzer

Testwell CTC++로 침투 테스트 중에 코드 커버리지를 측정하면 침투 테스트 평가를 크게 단순화할 수 있습니다.

목표는 고전적인 100% 코드 커버리지와 반대입니다. 침투 테스트 동안 가능한 한 적은 수의 소프트웨어 부분을 실행해야 합니다.

Support for C, C++, Java, and C#

Testwell CTC++는 다중 통합 적용 범위(Statement Coverage, Function Coverage, 의사 결정 적용 범위 / Branch 적용 범위, 상황 적용 범위 및 수정된 조건/의사 결정 적용 범위 포함)까지의 모든 적용 수준에 대해 C, C+++, Java 및 C#로 작성된 코드를 분석하는 유일한 코드 적용 도구입니다.

초창기의 Testwell CTC++는 C와 C++를 지원하기 위해 개발되었지만 2007년 이후로 추가적인 프로그래밍 언어로 기능성을 확장하기 위한 추가 기능을 제공합니다.

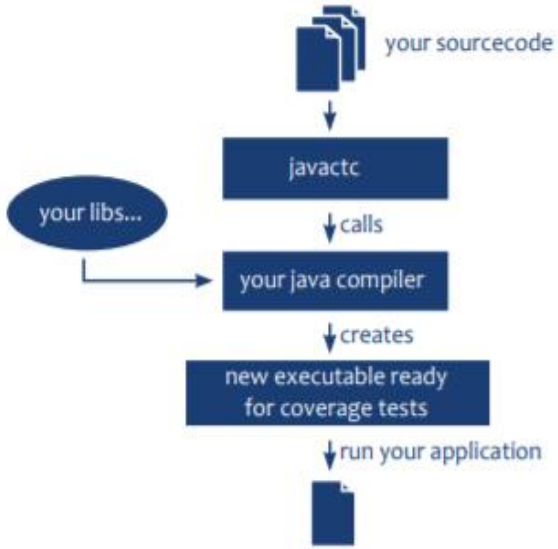
- ✓ **CTC++ for Java and Android add-on**
- ✓ **CTC++ for C# add-on**

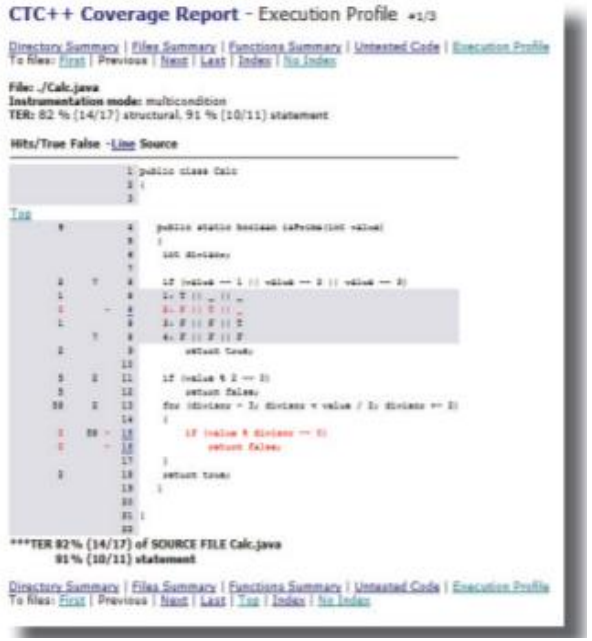
Easy to use

- 디버그 빌드가 필요하지 않음
- 빌드 파일 수정 불필요
- 기존 소스코드 수정 불필요
- 프로젝트의 적용 범위를 테스트하는데 4단계만 필요합니다.

How does it work?

- 소스 코드 작성
- 기본 컴파일러 대신 Java CTC를 호출합니다.
- 애플리케이션을 정상적으로 실행
- ctc2html, ctc2excel과 같은 도구를 사용하여 보고서 작성





Performs Kernel Coverage

Code Coverage of Linux Kernel

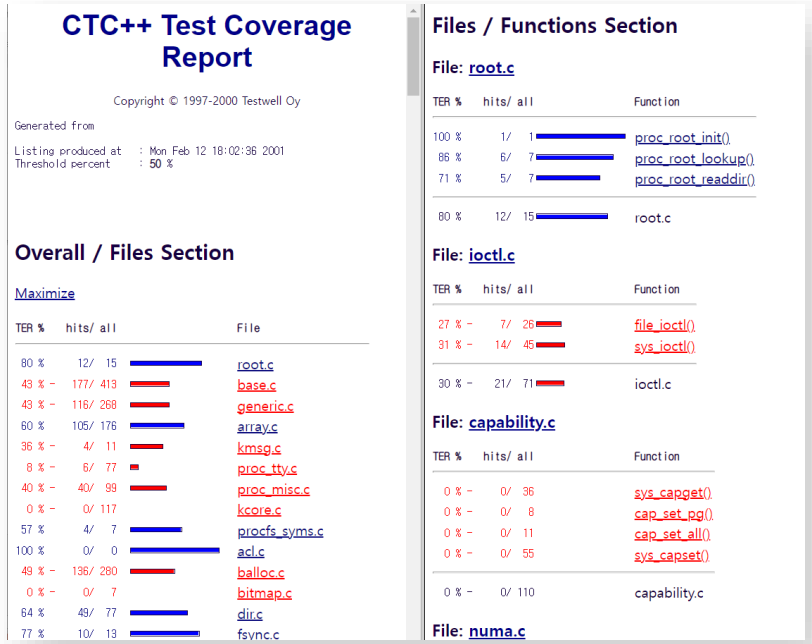
Kernel 은 소프트웨어의 I/O (인풋 아웃풋) 요청을 관리하고 중앙 처리 장치 및 컴퓨터의 기타 전자 부품에 대한 데이터 처리 명령으로 변환합니다.

Proof of Concept

이 기능은 CTC++ Host-Target 추가 기능 패키지의 일부입니다. Kernel coverage는 CTC++ Host-Target 의 전문화 입니다. CTC++ Host-Target과 동일한 구성 요소와 기술이 사용되지만 대상 환경은 호스트 자체의 커널 공간 코드입니다. 또한 Kernelcoverage component 는 수집된 실행 카운터 데이터가 어떻게 커널 공간에서 사용자 공간으로 전송될 수 있는지에 대한 기술을 설명하고 제안합니다.

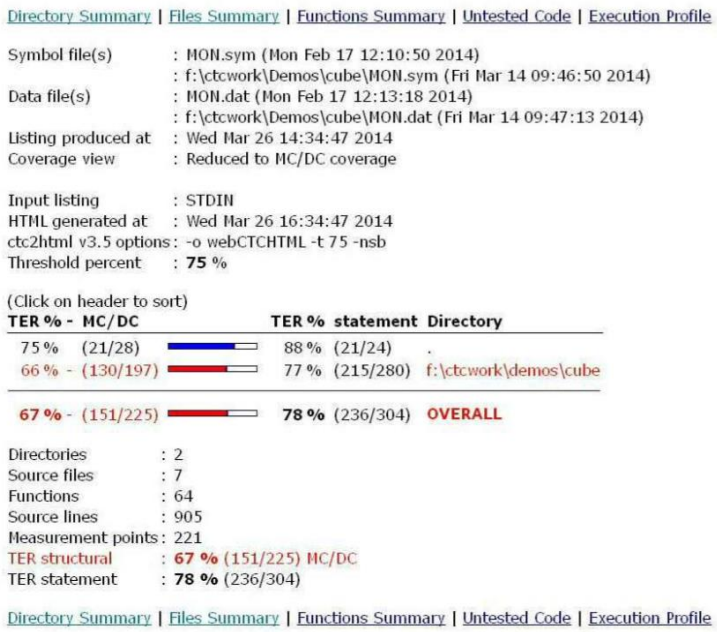
커널 공간 코드에서 계측된 프로브는 사용 공간 코드에서 사용할 수 있는 동안 라이브러리 함수 또는 시스템 호출을 사용할 수 없습니다. 계측을 수행하는 CTC++ 방식과 런타임 지원 계층은 시스템 서비스를 사용하지 않고 기본 C만 가정합니다. 따라서 커널 공간에서 원활하게 실행할 수 있습니다.

Kernel coverage 대한 개념 증명으로서 완전한 Linux 커널을 계측하고 이에 대한 특정 테스트 세트 실행 및 결과 CTC++ 적용 범위 보고서를 볼 수도 있습니다.



Different Reports

CTC++ Coverage Report – Directory Summary



Directory Summary | Files Summary | Functions Summary | Untested Code | Execution Profile

Symbol file(s) : MON.sym (Mon Feb 17 12:10:50 2014)
: f:\ctcwork\Demos\cube\MON.sym (Fri Mar 14 09:46:50 2014)
Data file(s) : MON.dat (Mon Feb 17 12:13:18 2014)
: f:\ctcwork\Demos\cube\MON.dat (Fri Mar 14 09:47:13 2014)
Listing produced at : Wed Mar 26 14:34:47 2014
Coverage view : Reduced to MC/DC coverage

Input listing : STDIN
HTML generated at : Wed Mar 26 16:34:47 2014
ctc2html v3.5 options : -o webCTCHTML -t 75 -nsb
Threshold percent : **75** %

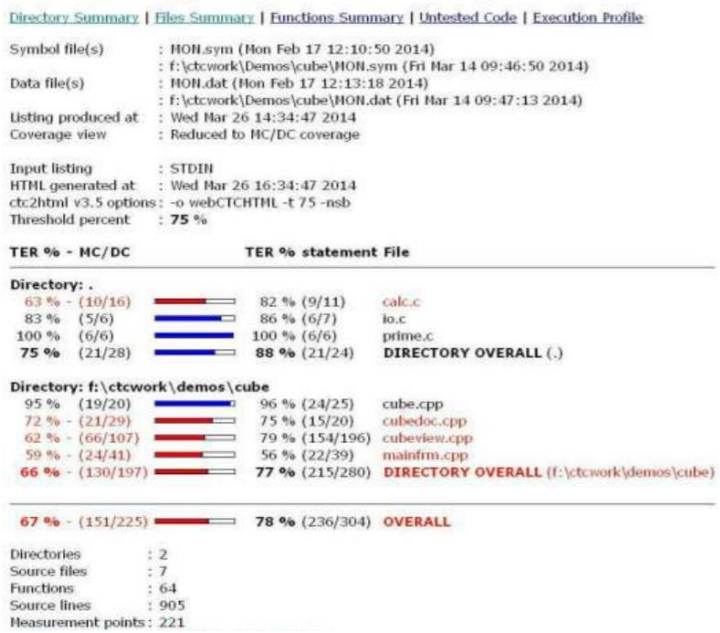
(Click on header to sort)

TER % - MC/DC		TER % statement		Directory
75 %	(21/28)	88 %	(21/24)	.
66 % -	(130/197)	77 %	(215/280)	f:\ctcwork\demos\cube
67 % -	(151/225)	**78 %**	(236/304)	**OVERALL**

Directories : 2
Source files : 7
Functions : 64
Source lines : 905
Measurement points : 221
TER structural : **67 %** (151/225) MC/DC
TER statement : **78 %** (236/304)

Directory Summary | Files Summary | Functions Summary | Untested Code | Execution Profile

CTC++ Coverage Report – Directory Summary



Directory Summary | Files Summary | Functions Summary | Untested Code | Execution Profile

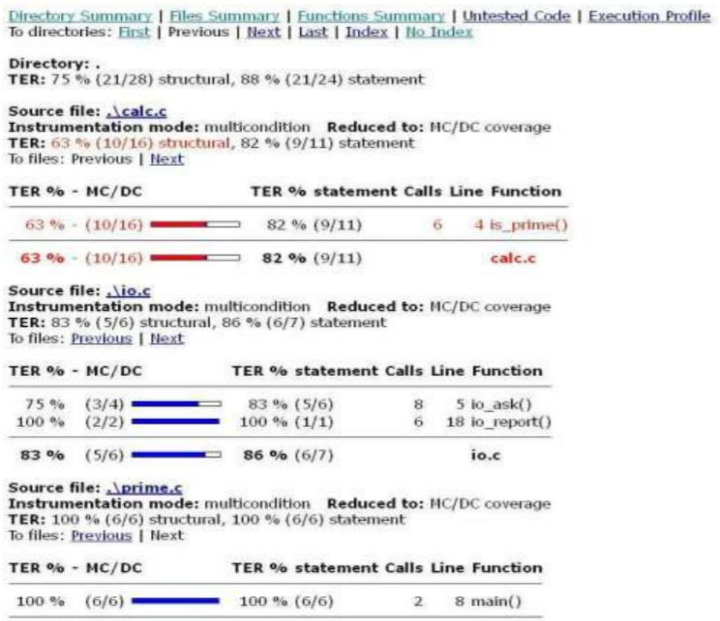
Symbol file(s) : MON.sym (Mon Feb 17 12:10:50 2014)
: f:\ctcwork\Demos\cube\MON.sym (Fri Mar 14 09:46:50 2014)
Data file(s) : MON.dat (Mon Feb 17 12:13:18 2014)
: f:\ctcwork\Demos\cube\MON.dat (Fri Mar 14 09:47:13 2014)
Listing produced at : Wed Mar 26 14:34:47 2014
Coverage view : Reduced to MC/DC coverage

Input listing : STDIN
HTML generated at : Wed Mar 26 16:34:47 2014
ctc2html v3.5 options : -o webCTCHTML -t 75 -nsb
Threshold percent : **75** %

TER % - MC/DC		TER % statement		File
Directory: .				
63 % -	(10/16)	82 %	(9/11)	calc.c
83 %	(5/6)	86 %	(6/7)	io.c
100 %	(6/6)	100 %	(6/6)	prime.c
75 %	(21/28)	**88 %**	(21/24)	**DIRECTORY OVERALL** (.)
Directory: f:\ctcwork\demos\cube				
95 %	(19/20)	96 %	(24/25)	cube.cpp
72 % -	(21/29)	75 %	(15/20)	cubedoc.cpp
62 % -	(66/107)	79 %	(154/196)	cubeview.cpp
59 % -	(24/41)	56 %	(22/39)	mainfrm.cpp
66 % -	(130/197)	**77 %**	(215/280)	**DIRECTORY OVERALL** (f:\ctcwork\demos\cube)
67 % -	(151/225)	**78 %**	(236/304)	**OVERALL**

Directories : 2
Source files : 7
Functions : 64
Source lines : 905
Measurement points : 221

CTC++ Coverage Report – Directory Summary



Directory Summary | Files Summary | Functions Summary | Untested Code | Execution Profile
To directories: First | Previous | Next | Last | Index | No Index

Directory: .
TER: 75 % (21/28) structural, 88 % (21/24) statement

Source file: .\calc.c
Instrumentation mode: multicondition **Reduced to:** MC/DC coverage
TER: 63 % (10/16) structural, 82 % (9/11) statement
To files: Previous | Next

TER % - MC/DC		TER % statement		Calls	Line	Function
63 % -	(10/16)	82 %	(9/11)	6	4	is_prime()
63 % -	(10/16)	**82 %**	(9/11)			**calc.c**

Source file: .\io.c
Instrumentation mode: multicondition **Reduced to:** MC/DC coverage
TER: 83 % (5/6) structural, 86 % (6/7) statement
To files: Previous | Next

TER % - MC/DC		TER % statement		Calls	Line	Function
75 %	(3/4)	83 %	(5/6)	8	5	io_ask()
100 %	(2/2)	100 %	(1/1)	6	18	io_report()
83 %	(5/6)	**86 %**	(6/7)			**io.c**

Source file: .\prime.c
Instrumentation mode: multicondition **Reduced to:** MC/DC coverage
TER: 100 % (6/6) structural, 100 % (6/6) statement
To files: Previous | Next

TER % - MC/DC		TER % statement		Calls	Line	Function
100 %	(6/6)	100 %	(6/6)	2	8	main()

Different Reports

CTC++ Coverage Report – Execution Profile #1/3

Directory Summary | Files Summary | Functions Summary | Execution Profile
To files: First | Previous | Next | Last | Index | No Index

File: ./calc.c
Instrumentation mode: function-decision-multicondition
TER: 82 % (14/ 17)

Start/ True	End/ False	-	Line	Source
			1	/* File calc.c ---------- */
			2	#include "calc.h"
			3	/* Tell if the argument is a prime (ret 1) or not (ret 0) */
				Top
9	0		4	int is_prime(unsigned val)
			5	{
			6	unsigned divisor;
			7	
2	7		8	if (val == 1 || val == 2 || val == 3)
1			8	T || _ || _
0		-	8	F || T || _
1			8	F || F || T
	7		8	F || F || F
2			9	return 1;
5	2		10	if (val % 2 == 0)
5			11	return 0;
58	2		12	for (divisor = 3; divisor < val / 2; divisor += 2)
			13	{
0	58	-	14	if (val % divisor == 0)
0		-	15	return 0;
			16	}
2			17	return 1;
			18	}

***TER 82% (14/17) of SOURCE FILE calc.c

Directory Summary | Files Summary | Functions Summary | Execution Profile
To files: First | Previous | Next | Last | Top | Index | No Index

CTC++ Coverage Report – Execution Profile #1/7

Directory Summary | Files Summary | Functions Summary | Untested Code | Execution Profile
To files: First | Previous | Next | Last | Index | No Index

Source file: .\calc.c
Instrumentation mode: multicondition **Reduced to:** MC/DC coverage
TER: 63 % (10/16) structural, 82 % (9/11) statement

Hits/True	False	-	Line	Source
			1	/* File calc.c ---------- */
			2	#include "calc.h"
			3	/* Tell if the argument is a prime (ret 1) or not (ret 0) */
				Top
6			4	int is_prime(unsigned val)
			5	{
			6	unsigned divisor;
			7	
2	4		8	if (val == 1 || val == 2 || val == 3)
0			8	1: T || _ || _
2			8	2: F || T || _
0			8	3: F || F || T
	4		8	4: F || F || F
		-	8	MC/DC (cond 1): 1 - 4
			8	MC/DC (cond 2): 2 + 4
		-	8	MC/DC (cond 3): 3 - 4
2			9	return 1;
2	2		10	if (val % 2 == 0)
2			11	return 0;
0	2	-	12	for (divisor = 3; divisor < val / 2; divisor += 2)
			13	{
0	0	-	14	if (val % divisor == 0)
0		-	15	return 0;
			16	}
2			17	return 1;
			18	}

***TER 63% (10/16) of FILE calc.c
82% (9/11) statement

Directory Summary | Files Summary | Functions Summary | Untested Code | Execution Profile
To files: First | Previous | Next | Last | Top | Index | No Index

CTC++ Coverage Report – Untested Code

Directory Summary | Files Summary | Functions Summary | Untested Code | Execution Profile
To files: Index | No Index

Source file: .\calc.c
Instrumentation mode: multicondition **Reduced to:** MC/DC coverage
TER: 63 % (10/16) structural, 82 % (9/11) statement

Hits/True	False	-	Line	Source
6			4	FUNCTION is_prime()
		-	8	MC/DC (cond 1): 1 - 4
		-	8	MC/DC (cond 2): 2 + 4
		-	8	MC/DC (cond 3): 3 - 4
0	2	-	12	for (;divisor < val / 2;)
0	0	-	14	if (val % divisor == 0)
0		-	15	return 0

***TER 63% (10/16) of FILE calc.c
82% (9/11) statement

Source file: .\io.c
Instrumentation mode: multicondition **Reduced to:** MC/DC coverage
TER: 83 % (5/6) structural, 86 % (6/7) statement

Hits/True	False	-	Line	Source
8			5	FUNCTION io_ask()
0	8	-	11	if ((amount = scanf ("%u" , & val)) <= 0)

***TER 83% (5/6) of FILE io.c
86% (6/7) statement

Source file: f:\ctcwork\demos\cube\cube.cpp
Instrumentation mode: multicondition **Reduced to:** MC/DC coverage
TER: 95 % (19/20) structural, 96 % (24/25) statement

rue	False	-	Line	Source
1			55	FUNCTION CCubeApp::InitInstance()
0	1	-	70	if (m_lpCmdLine [0] != '\0')

***TER 95% (19/20) of FILE cube.cpp
96% (24/25) statement

Integrations in many tool chains and testing environments

Testwell CTC++ 는 많은 IDE 및 다음 툴 체인, 테스트 환경 및 소프트웨어 품질 툴과 통합됩니다.

- ✓ CATIA Systems - AUTOSAR Builder (DASSAULT SYSTEMES)
- ✓ Cygwin
- ✓ dSPACE SystemDesk®
- ✓ dSPACE TargetLink®
- ✓ Imagix 4D
- ✓ Jenkins
- ✓ Lauterbach
- ✓ MATLAB Simulink
- ✓ PikeTec Time Partition Testing (TPT)
- ✓ QTronic TestWeaver
- ✓ QTronic Silver
- ✓ SonarQube
- ✓ TIOBE Software Quality Framework TICS

Costumers


ABB
AIRBUS DEFENCE & SPACE
AIRBUS HELICOPTERS
AKKA TECHNOLOGIES
ALSTOM
AMG
arianeGROUP
ATTENSITY
Audi
Baxter
BMW
THALES
BOMBARDIER
BorgWarner
BOSCH
BT
Federal Office for Information Security
ca technologies
Canon
CATL
CINTERION WIRELESS MODULES
CISCO
COMARCH INFORMATION TECHNOLOGY
Westinghouse
Continental
DAIMLER
DGA
DLR
DONGFENG
dSPACE
EATON Powering Business Worldwide
Elektrobit
EMBRAER
esa
ESO
Exonetik
YASKAWA
ZEISS
framatome
GETINGE
gemalto security to be free
General Electrics imagination at work
Giesecke & Devrient security at work.
长城汽车 Great Wall Motors
HARMAN
HELLA
HITACHI Inspire the Next
HONDA
Honeywell
VORWERK
HYUNDAI DYMOS
IDEMIA
Infineon
intel inside
jenaoptronik Aerospace and Security
KNORR-BREMSE
KONGSBERG
LIEBHERR
MAQUET GETINGE GROUP
MBtech Mercedes-Benz technology
McLaren AUTOMOTIVE
TOMTOM
medela
Mentor Graphics
Micron
NASA
НПП ИТЭЛМА
NISSAN
NOKIA
NXP founded by Philips
OHB
PHILIPS
VIESMANN
PORSCHE
Raytheon
RENESAS
Roche
SAFRAN Morpho
SCANIA
SCHAEFFLER
Schindler
Schneider Electric
secunet
SICK
WATLOW Powered by Possibility
SIEMENS
SKF
ST Kinetics
Sopra group
SPIE
ST
SAMSUNG
SONY
T-Mobile
T-Systems
TELESPAZIO a LEONARDO and THALES company
thyssenkrupp
VW