

C++test 用作 DO-178B 认证的资质审核套件

C++test用作DO-178B认证的资质审核套件	1
简介.....	2
概述.....	2
工具的功能与技术特性	3
编码标准	3
先进的数据流执行路径测试.....	4
可控的代码走查流程	4
软件测试	5
安装指南与用户手册等用户信息.....	6
工具的操作环境.....	6
写在最后	7

简介

C++test 可以用作 DO-178B 标准的“验证工具 (Verification Tool)”，本文及其附件一起为认证该标准的“资质审核套件 (Qualification Kit)”。其目的是帮助 Parasoft 用户将 C++test 用作 DO-178B 标准认证的软件验证工具。本文件是源于我们对软件的最佳知识以及“机载系统以及设备认证的软件编码考虑” (SOFTWARE CONSIDERATIONS IN AIRBORNE SYSTEM AND EQUIPMENT CERTIFICATION) DO-178B/ED12B 的要求编写的。该标准由 RTCA SC-167/EUROCAE WG-12 制定，并于 1992 年 12 月 16 日由航空无线电委员会 (RTCA, Inc.) 颁布。

本文件创建于 2008 年 12 月 12 日，在本文档中提到的 C++test 是指的 C++test 7.2 这个版本。相关信息请参阅最新的 C++test 文档。

概述

按照 DO-178B 标准的划分，工具可以划分为两种(请参阅该标准第 12 章“工具的界定”)。以下文字是直接引用的该标准中的描述：

“软件工具可以分为以下两类：

- o 软件开发工具：这种工具的输出是机载软件 (airborne software) 的一部分，所以能够引入错误。例如根据低级需求直接生成的源代码如果未按照第 6 章中的方法进行验证，则必须通过资质审核。

- o 软件验证工具：这种工具不会引入错误，但是有可能会检测不到错误。例如，自动进行软件验证的静态代码分析器，如果其功能未被验证则应该进行审核。类型检测器、分析工具以及测试工具也如此”

按照该标准的划分，C++test 应该属于软件验证工具这一类，因为其输出不会对待认证软件引入错误。对于软件验证工具而言，该标准提供了如下一些对工具进行审核的准则，以下文字仍然是对该标准的直接引用：

“12.2.2 软件验证工具的审核准则

软件验证工具的审核准则应能够证实该工具能够在正常工作条件下满足其相应的工具操作需求 (Tool Operational Requirements)”

以及

“12.2.3.2 工具操作需求 (Tool Operational Requirements)

工具操作需求用以描述该工具的功能。这些数据应该包括：

- a. 对工具的功能和技术特性的描述。对于软件开发工具而言，应包括软件开发流程中的工具完成的任务。
- b. 诸如安装指南以及用户手册等的用户信息。
- c. 关于工具运行环境的描述。
- d. 对于软件开发工具而言，应提供工具在非正常操作条件下的预期响应。”

12.2.3.2 章中的 d 是用来约束软件开发工具的，因此与 C++test 无关。所以我们在文章的后续章节中将着重讨论 a、b 以及 c。

工具的功能与技术特性

Parasoft 的 C++test 提供了一种自动实施经广泛证实的最佳实践以提升软件开发团队的开发效率以及软件质量的集成解决方案。Parasoft C++test 帮助开发团队提升代码质量、测试效率并在开发过程中随时监控软件开发的质量趋势。

C++test 中经广泛证实的最佳实践（如静态代码分析、综合代码走查以及集成代码覆盖率分析的单元和组件测试等）可以在软件开发早期在开发者的桌面环境下进行。

C++test 还提供对命令行的支持，从而使回归测试以及持续性集成环境中的测试能够自动进行，同时还提供用以监控质量趋势的数据供开发者分析。

根据 DO-178B/ED-12B 标准的要求，工具的以下一些功能是非常重要的：

编码标准

关于软件验证的必要性/可行性在 DO-178B/ED-12B 标准的第 6.3.4d 章（源码走查以及分析 — 与标准的一致性 “Reviews and Analyses of the Source Code - Conformance to standards”）中有详细的论述。

C++test 的编码标准分析能够监视用户代码是否满足那些围绕着安全性、可靠性、性能、可维护性以及其它指标指定的业界标准或用户自定义规则。C++test 编码标准分析使用了 Parasoft 专利的基于模式（pattern-based）的分析引擎。

C++test 的编码标准分析有如下功能：

- 支持业界的核心标准（如 JSF、MISRA、FDA、PCI、Ellemtel、Security、Section 508 等）以及项目特定的策略。
- 支持编码标准规则的参数化。
- 支持定义以及检查自定义规则，以防止一些工程相关的错误的重复发生并监控团队定义的策略的一致性。
- 支持将一组规则定制为验证规则集，并根据工程的特殊需要重新映射相应规则的严重级。
- 丰富的报告功能。

- 对每条规则都给出了其相应的正确与错误的示例及其详尽的阐述。

关于编码标准配置以及执行的详细阐述可以在 **Parasoft C++test** 的用户指南中的第 227 页至 246 页“编码标准静态分析”这一章节中找到。

“**C++test** 系统化规则描述”这个文档提供了对各种规则的详尽讲解。

上述两个文档同样是本资质审核套件的一部分。

先进的数据流执行路径测试

关于软件验证的必要性/可行性在 DO-178B/ED-12B 标准的第 6.3.4f 章（源码走查以及分析 — 准确性与一致性 “**Reviews and Analyses of the Source Code - Accuracy and consistency**”）中有详细的论述。

BugDetective 是 **C++test** 的进程间静态代码分析模块，它能模拟应用程序执行时可能跨越多个函数和文件的潜在路径，并确定这些路径在实际执行时是否会造成运行时缺陷。这些缺陷包括使用未初始化或无效内存、空指针引用（**dereferencing**）、数组及缓存溢出、被零除、内存及资源泄漏以及多种无效代码（**dead code**）。这种无需运行代码来查找代码缺陷的方法对于嵌入式系统的代码分析而言是非常有用的，因为，对嵌入式系统代码的运行时分析往往是很费时甚至是不可行的。**C++test** 能在开发者的 IDE 中提供对所有的潜在缺陷的路径最完整的追踪。

关于先进数据流执行路径测试（**BugDetective**）的配置以及执行的详细阐述可以在 **Parasoft C++test** 用户指南中的第 247-257 页“**BugDetective（数据流）静态分析**”这一章节中找到。

“**C++test** 系统化规则描述”这个文档提供了对 **BugDetective** 使用的规则的详尽讲解。

可控的代码走查流程

关于软件验证的必要性/可行性在 DO-178B/ED-12B 标准的第 6.3.4 章（源码走查以及分析 “**Reviews and Analyses of the Source Code**”）中有详细的论述。在 6.3.4 中的从 a 到 f 所指都是以人工代码走查为前提的。不过 a)、b)、c) 以及 e) 中的要求特别适用于人工操作。d)和 f)仍然适合于上述的更加自动化的验证方式。

C++test 的代码走查模块将同行代码走查的准备工作、提示以及跟踪都自动进行，从而营造了一个非常高效的面向团队（**team-oriented**）的流程。包括代码走查器（**reviewer**）的注释在内的所有代码走查状态都在 **C++test** 框架

(**infrastructure**) 中得到维护并自动进行分发。以下是两种 **C++test** 的典型代码走查流程：

- 提交前 (**pre-commit**) 代码走查。这种模式是基于对代码库中的代码更改的自动识别进行的。通过自定义的源码控制接口并根据代码走查器中预设的代码更改映射关系来创建代码走查任务。
- 提交后 (**post-commit**) 代码走查。在这种模式下，用户可以通过两种方法来启动代码走查：在桌面环境中为代码走查选择一组更改过的文件或自动查找本地修改过的代码。

这两种模式都可以混合使用。这样能确保所有的代码都得到验证。

C++test 的静态代码分析功能进一步提升了团队的代码走查效率。由于团队编码策略得到了自动的监控，所以逐行代码审查的必要性已经被排除了。这样就保证了当提交代码时，所有的冲突都已经被查找到并已经得到了清除。从而使得代码走查能更着重于检查算法、设计以及自动化工具无法检测到的一些隐蔽的缺陷。

关于 **C++test** 代码走查模块的详细阐述可以在 **Parasoft C++test** 用户指南中的第 257-287 页“代码走查”这一章节中找到。

软件测试

关于软件测试流程，读者可以在 **DO-178B/ED-12B** 标准中的第 6.4 章（“软件测试流程”）找到详尽的阐述。该标准中给出了两个相辅相成的目标。一个目标是证明软件能满足相应的需求。另一个目标是需要确认系统安全评估进程（**system safety assessment process**）所确定的那些不可接受的错误条件已经被清除掉。

C++test 能根据需求生成类似于 **CppUnit** 格式的纯 **C** 或 **C++** 代码的测试用例。通过一组特定的 **GUI**，测试用例的创建以及管理变得相当简便。使用图形化的测试用例向导（**Test Case Wizard**），开发者能迅捷地为指定功能创建黑盒功能测试套件，而不必担心模块的内部工作情况或数据的相关性等问题。数据源向导（**Data Source Wizard**）能帮助开发者对测试用例以及桩函数进行参数化的设置，只需进行少量的设置，测试范围以及代码覆盖率将得到极大地提升。在桩函数面板中还能对桩函数的分析以及生成进行控制，在该面板中，代码中的所有功能都将被列举出来，同时还允许开发者为当前测试范围内不可用的功能创建桩函数，或者根据特定的测试需求改变既有功能。测试用例浏览器提供对测试执行以及分析的集中控制，统一管理当前既有工程的测试并提供所有的测试通过/失败的状态显示。如需更详细的信息，请参阅 **DO-178B/ED-12B** 标准中的第 6.4.2.1 这一章节（“常规测试用例（**Normal Range Test Cases**）”）。

关于使用 **C++test** 自动生成测试用例的详细阐述可以在 **Parasoft C++test** 用户指南中的第 289-307 页“生成并执行测试”这一章节中找到。

一个包括对语句、分支、路径以及 **MC/DC** 覆盖率的进行多重计量的测试覆

覆盖率分析工具能够帮助用户精确地评估测试的效率以及完成度。测试覆盖率是通过在 GUI 中高亮显示覆盖率计量中所覆盖到的代码或在报告中的通过不同颜色的代码来表示的。覆盖率总结报告包括文件、类以及按照多种格式构造的函数数据。如需更详细的信息, 请参阅 DO-178B/ED-12B 标准中的第 6.4.4.2 和 6.4.4.3 这两部分 (“结构化覆盖率分析 (Structural Coverage Analysis)” 以及 “结构化覆盖率分析的分辨率 (Structural Coverage Analysis Resolution)”)。

关于 C++test 中的多重计量的测试覆盖率分析工具的详细阐述可以在 Parasoft C++test 用户指南中的第 320-332 页 “浏览覆盖率信息” 这一章节中找到。

安装指南与用户手册等用户信息

文档题目	文件名	说明
C++test 7.2 Getting Started Guide	c++test_72_gettingstarted.pdf	C++test 的安装及配置指南
Parasoft C++test User' s Guide	c++test_72_userguide.pdf	C++test 用户手册
C++test Systematic Description Rules	c++test_72_rules.pdf	C++test 提供的规则和编码标准测试的详细描述

工具的操作环境

支持平台

- Windows NT/2000/XP/2003/Vista
- Linux kernel 2.4 或 2.6 或更高版本, 提供 2.2 或更高版本的 glibc, 兼容 x86 处理器
- Linux kernel 2.6 或更高版本, 提供 2.3 或更高版本的 glibc, 兼容 64 位 x86 处理器 (提供 32 位处理器兼容支持包)
- 运行在 UltraSPARC 处理器上的 Solaris 7, 8, 9, 10

编译器

- Windows:
 - Microsoft Visual C++6.0, .NET (7.0), .NET 2003(7.1), 2005 (8.0), 2008 (9.0)
 - GNU 以及 MingW gcc/g++ 2.95.x, 3.2.x, 3.3.x, 3.4.x

- GNU gcc/g++ 4.0.x, 4.1.x, 4.2.x, 4.3
- Linux (x86 处理器):
 - GCC 2.95.x, 3.2.x, 3.3.x, 3.4.x, 4.0.x, 4.1.x, 4.2.x, 4.3
- Linux(64 位 x86 处理器):
 - GCC 3.4.x, 4.0.x, 4.1.x, 4.2.x, 4.3
- Solaris:
 - Sun C++ 5.3 (Sun Forte C++ 6 Update 2), Sun C++ 5.5 (Sun ONE Studio 8), Sun C++ 5.6 (Sun ONE Studio 9), Sun C++ 5.7 (Sun ONE Studio 10), Sun C++ 5.8 (Sun ONE Studio 11)
 - GCC 2.95.x, 3.2.x, 3.3.x, 3.4.x, 4.0.x, 4.1.x, 4.2.x, 4.3

IDE

- Eclipse 3.1, 3.2 (32-bit), 3.3 (32-bit), 3.4 (32-bit)
- Visual Studio .NET 2003, 2005, 2008
- Wind River Workbench 2.6 以及 3.0
- ARM Real View Development Studio (RVDS) 3.0, 3.1 以及 4.0
- NetBurner

支持导入的 IDE

- Microsoft Visual Studio 6, eMbedded Visual C++ 4.0
- Wind River Tornado 2.0, 2.2
- Green Hills MULTI 4.0.x

支持目标编译器

- Wind River GCC 4.1.x, 3.4.x, 2.96, DIAB 5.4-5.6x, EGCS 2.90
- GNU GCC Cross Compilers 2.95-4.3
- ARM ADS 1.2, RVCT 3.0, RVCT 3.1
- Microsoft Visual C++ 8.0 以及 9.0 for Windows Mobile, Embedded Visual C++ 4.0
- QNX QCC 2.95 以及 3.3
- Green Hills 4.0.x
- STMicroelectronics ST20, ST40 (仅支持静态代码分析)

免责声明

关于最新的支持环境的信息，请用户始终以最新版本的 Parasoft C++test 用户手册第 9 章支持环境中的描述为准。

写在最后

如果用户觉得在某些特定 DO-178B/ED-12B 工程认证中，本资质审核套件提供的关于将 C++test 作为验证工具的信息不够详细，Parasoft 的支持团队十分乐于为用户提供支持以及认证所需要的资料以及信息。如有疑问，请联系

info@parasoft-embedded.com