ICS
CCS

团　　体　　标　　准

T/CI XXX-2023

无源码的白盒化测试标准

Standard for glass-box testing without source code

（征求意见稿）

2023-X-XX 发布　　2023-X-XX 实施

中国国际科技促进会　发布

目 次

前　言

本文件按照GB/T 1.1—2020《标准化工作导则 第1部分：标准化文件的结构和起草规则》的规定起草。

本文件由中国国际科技促进会标准化工作委员会提出。

本文件由中国国际科技促进会归口。

本文件的发布机构提请注意，声明符合本文件时，可能涉及到ZL201710766422.3、ZL201810501975.0、ZL201910819271.2和ZL202011267313.5相关的专利的使用。

本文件的发布机构对于该专利的真实性、有效性和范围无任何立场。

该专利持有人已向本文件的发布机构保证，他愿意同任何申请人在合理且无歧视的条款和条件下，就专利授权许可进行谈判。该专利持有人的声明已在本文件的发布机构备案。相关信息可以通过以下联系方式获得：

专利持有人姓名：华南理工大学

地址：广东省广州市

请注意除上述专利外，本文件的某些内容仍可能涉及专利。本文件的发布机构不承担识别这些专利的责任。

本文件起草单位：华南理工大学、微科智检(佛山市)科技有限公司、北明软件有限公司。

本文件主要起草人：黄翰、曹捷、刘方青、向毅、徐杨。

本文件是首次发布。

无源码的白盒化测试标准

1. 适用范围

本文件规定了无源码的白盒化测试标准的测试流程、需求模型、测试类型等要求。

本文件适用于软件系统的无源码的白盒化测试以及相应的测试技术人员。

2. 规范性引用文件

下列文件中的内容通过文中的规范性引用而构成本文件必不可少的条款。其中，注日期的引用文件，仅该日期对应的版本适用于本文件。不注日期的引用文件，其最新版本（包括所有的修改单）适用于本文件。

GB/T 25000.23-2019 系统与软件工程 系统与软件质量要求与评价(SQuaRE) 第23部分系统与软件产品质量测量

GB/T 25000.51-2016 系统与软件工程 系统与软件质量要求和评价(SQuaRE) 第51部分就绪可用软件产品(RUSP)的质量要求和测试细则

GB/T 15532-2008 计算机软件测试规范

GB/T 39788-2021 系统与软件工程 性能测试方法

GB/T 30279-2020 信息安全技术 网络安全漏洞分类分级指南

3. 术语和定义

下列术语和定义适用于本文件。

3.1 黑盒程序 black box binary program

可执行的无源代码二进制程序文件。

3.2 黑盒 black box

一个系统或布局，它的输入、输出和通用功能是已知的，但它的内容或实现是未知的或无关的。

3.3 黑盒测试 black-box testing

忽略系统或部件的内部机制只集中于响应所选择输入和执行条件产生的输出的一种测试。

3.4 白盒 glass box

一个系统或布局，它的实现内容是已知的。

3.5 白盒测试 glass-box testing

侧重于系统或部件内部机制的测试。类型包括分支测试、路径测试、语句测试。

3.6 插桩 instrument

在软件或系统测试中，将某些器件或指令安装或插入至硬件或软件中，以监控系统或部件的操作。

3.7 程序插桩 program instrumentation

插入到计算机程序中的探头。如指令或断言，以利于执行监控、正确性证明、资源监控或其他活动。

3.8 仿真 simulation

一个模型，当提供一组控制输入时，他的行为或操作像一给定的系统。

3.9 傅里叶网络 fourier neural network

以一组傅立叶基函数作为三层前向神经网络各隐含层单元的输出特性，再以其加权和作为网络的非线性输出，构成的一种神经网络模型。

3.10 训练集 train set

在机器学习中，一般将样本分成独立的三部分训练集，验证集和测试集。其中，训练集用于建立模型。

3.11 可执行程序 executable program

可在操作系统存储空间中浮动定位的二进制可执行程序。它可以加载到内存中，由操作系统加载并执行。

4. 无源码的白盒化测试概述

无源码的白盒化测试用于软件开发商自行软件质量风险评估及第三方质量风险评估，用于软件正式交付用户使用前，进行软件质量风险分析、判定等，同时，对已使用中的软件可能出现的风险进行评估，以及针对已发生问题的软件进行责任判定。

无源码的白盒化测试规定软件质量风险快速评估的操作规范，具备技术通用性及一致性，可以更好的进行软件发布前使用中的质量风险评估，及出现事故后

的软件质量风险的问题鉴定，能给软件开发方及第三方提供科学有效的操作规范和指南。

5. 无源码的白盒化测试过程

5.1 概述

无源码的白盒化测试过程包括黑盒程序内部机制还原、黑盒程序插桩、黑盒程序白盒化测试执行和白盒化测试总结四个过程。图 1 给出了无源码的白盒化测试过程。

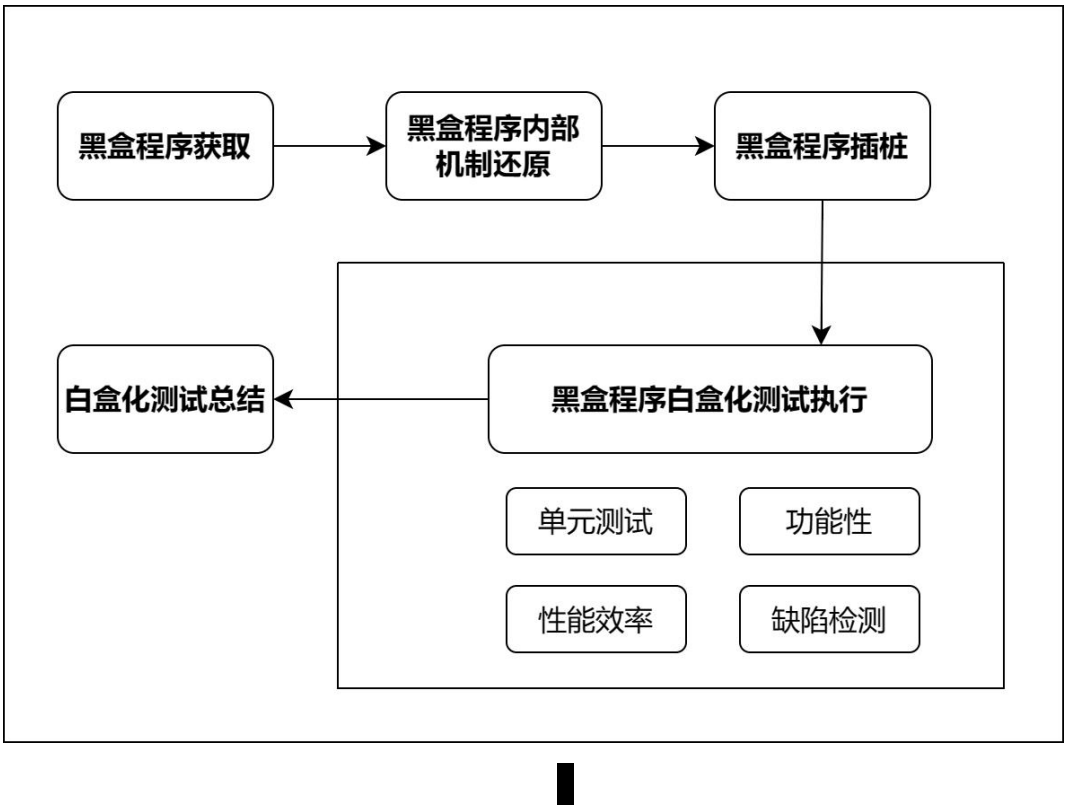

图 1 无源码的白盒化测试过程

5.2 黑盒程序内部机制还原

黑盒程序内部机制还原包括下列活动：

a) 获取黑盒程序对应的输入和输出形成训练集。

b) 根据获取的黑盒程序对应的训练集训练傅里叶网络，得到黑盒程序内部机制还原的仿真模型。

c) 在训练的初始阶段需要对训练集的输入和输出做二进制处理，得到符合傅里叶网络输入的数据。

根据黑盒程序的输出需要对仿真模型的二进制输出做还原处理，得到符合黑

盒程序的输出。

5.3 黑盒程序插桩

黑盒程序插桩通过插入各种类型的桩节点，动态地监测程序执行的整个过程。插桩的过程如图 2 所示，相关的活动包括：

a) 读取黑盒程序内部机制还原后的中间程序信息。

b) 解析黑盒程序的控制流程图。

c) 在程序中的分支前后插入桩节点。桩节点不改变程序原有功能，只用于记录程序运行时的执行情况。

d) 输出插桩后的可执行文件。执行该插桩后的程序，得到响应的桩节点序列即为程序执行过程的路径信息。

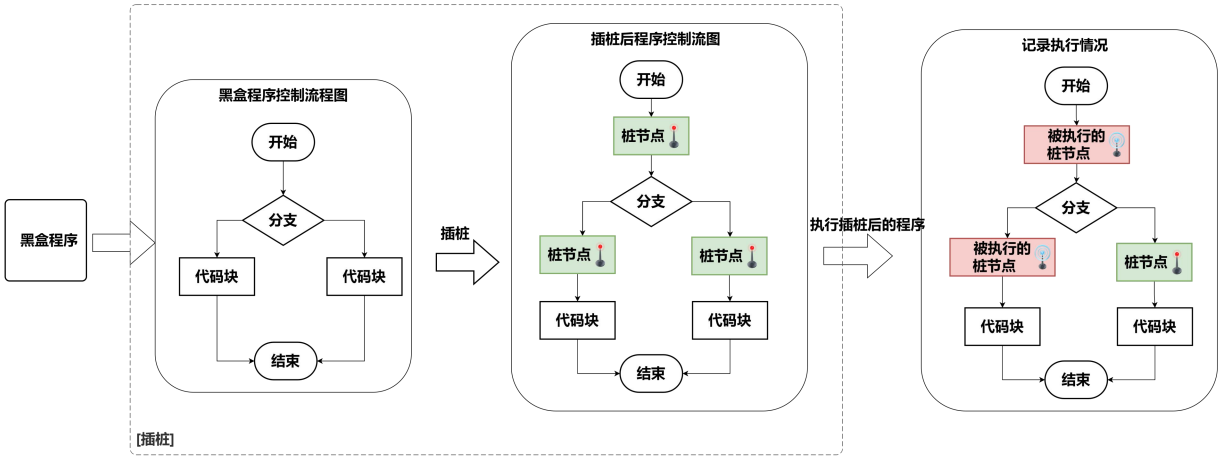

图 2 黑盒程序插桩过程

5.4 黑盒程序白盒化测试执行

黑盒程序白盒化测试执行包括下列活动：

a) 自动部署测试环境。

b) 根据被测软件的需求设定测试目标，如可要求测试用例满足语句覆盖、分支覆盖或路径覆盖等要求。

c) 设定算法并自动生成测试用例。

d) 根据测试约束，在测试用例用中自动进行断言检测。

e) 输出符合测试目标的测试用例集。

f) 继续执行缺陷检测，功能性检测及性能效率检测。

5.5 白盒化测试总结

白盒化测试总结包括下列活动：

a）整理白盒化测试结果。白盒化测试结果应根据不同的测试类型进行综合分析，测试类型包括：单元测试、缺陷检测、功能性检测、性能效率检测。采用数学和统计方法进行数据综合分析考虑。

b）编写黑盒程序白盒化测试报告，内容应包括：测试结果分析、测试用例统计、测试覆盖率统计、对软件的评价和建议。

c）根据测试结果和测试报告内容，输出风险分析报告单，内容应包括风险等级、风险概率、风险修复建议。

6. 无源码白盒化测试的技术要求

6.1 概述

无源码白盒化测试技术要求应考虑环境、数据、输入输出参数等因素。

6.2 环境需求

针对不同类型的黑盒软件，应考虑测试环境对白盒化测试的影响，推荐使用系统或软件的实际生产环境作为白盒化测试环境。在进行白盒化测试环境规划和设计时，应考虑以下因素：

a）独立性：测试环境应使用全新纯净的虚拟化容器以防止与其他系统或软件产生冲突造成测试结果失真。

b）可控制性：测试环境中的所有设备和资源应可被监控或控制。

c）性能：测试环境应支持黑盒仿真算法以及用例自动生成算法的性能需求。

6.3 数据需求

无源码的白盒化测试所需数据包含一下需求：

a）测试数据应根据实际需求变更，与软件的业务需求相匹配。

b）无源码被测试程序的接口已知、可调用。

6.4 输入输出参数需求

a）无源码被测试程序的输入输出参数清晰、无隐藏参数；输出参数可获取。

b）无源码被测试程序的输入参数维度可控（10 维以内）。

7. 无源码的白盒化测试类型

7.1 应用场景

白盒化技术主要能够应用于以下五类场景：

a) 我方软件缺陷检测。白盒化技术是基于生成大量复杂黑盒系统的输入-输出数据进行仿真的，在此过程中会生成大量的测试用例，相关技术会记录用例的运行状态与日志，若输入测试用例触发了我方软件存在的缺陷漏洞，本技术能够将触发用例与对应软件的代码位置返回，方便我方软件开发人员对软件潜在缺陷进行修复。

b) 敌方软件缺陷检测。与我方软件缺陷检测技术类似，在获取了触发敌方软件缺陷的用例与信息后，可根据实际需要对敌方软件进行攻击，瘫痪敌方软件系统。

c) （离线的）口令生成器仿真。在军事应用场景中存在口令生成器来进行敌我识别辨认的场景。而白盒化技术能够针对该场景仿真出实际情况下对敌方口令生成器的对应口令，从而使我军战士潜入需要敌方口令才能进入的区域。

d) 注入木马/病毒前后程序版本对比。存在木马或病毒注入的程序可能对我方软件系统造成巨大的破坏，白盒化系统具备对程序的版本对比功能，对于注入木马/病毒的程序，系统能够检测出异常程序的存在，并提供相应修改代码段信息。

e) 卡脖子算法白盒化与仿制。本技术能够对外国软件产品的核心算法进行仿制，为提升国产算法的设计水平提供思路。

7.2 基本要求

7.2.1 应能够对程序的输入输出进行抓取。

7.2.2 软件在使用过程中存在的任何问题都叫软件缺陷。

7.2.3 应在最低的功能/参数上进行缺陷检测。

7.2.4 缺陷检测应能检测出程序的内存泄漏、功能错误、程序闪退等问题。

7.2.5 缺陷检测过后，机器状态保持不变。

7.2.6 应保证在超时限制内完成缺陷检测，超时则自动终止。

7.2.7 缺陷检测应产生可重复的、一致的结果。

7.2.8 应形成可阅读的缺陷检测报告。

7.2.9 应可仿真口令生成器。

7.2.10 应提供二进制软件对比报告。

7.2.11 二进制软件对比应明确指出不同点。

7.2.12 应可进行算法的白盒化和仿制。

7.3 功能性

7.3.1 应验证是否有不正确或遗漏的功能，功能实现是否满足用户需求和系统设计的隐藏需求，并验证业务流程是否正确且合理。

7.3.2 应测试软件在极限值情况下整体产品表现，具体包括以下项目：

a) 软件本身支持的边界值（最大值和最小值）的测试，如字符串的最大/小长度、存储空间或条数的大小等；

b) 整体产品的极限值测试，比如，在宿主设备内存、CPU 等达到极限值时软件各功能的表现。

7.3.3 应对软件进行缺陷检测，包括单元测试和整个软件的缺陷检测。具体包括以下项目：

a) 模块功能的缺陷，包括内存泄露，输出错误等问题。

b) 整个程序的缺陷，包括内存泄露，输出错误等问题。

c) 软件未实现需求说明书中明确要求的功能。

d) 软件出现了需求说明书中指明不应该出现的错误。

e) 软件实现的功能超出需求说明书指明的范围。

f) 软件未实现需求说明书中虽未明确指明但应该实现的要求。

g) 软件难以理解，不易使用，运行缓慢，用户体验不好。

7.3.4 应能够对口令生成器进行仿真，包括模块仿真和程序仿真。具体包括以下项目：

a) 在机器上正确运行。

b) 获取模块或者程序的输入数据。

c) 获取模块或者程序的输出数据。

d) 使用输入输出数据对程序进行仿真。

7.3.5 应可对注入木马/病毒前后程序版本进行对比。具体包括以下项目：

a) 检测出异常程序的存在。

b) 识别出对比程序的不同之处。

c) 提供程序对比报告。

7.3.6 应能够对算法进行仿真。具体包括以下项目：

a) 在机器上正确运行。

b） 获取算法的输入数据。

c） 获取算法的输出数据。

d） 使用输入输出数据对程序进行仿真。

7.4 性能效率

7.4.1 根据程序特性，应能在有限合理的时间范围内生成能触发我方或敌方程序缺陷的测试用例，相关技术利用测试用例应在有限时间内返回程序缺陷相关信息。

7.4.2 应能在生成触发程序缺陷的测试用例的同时，不影响程序的后续使用。

7.4.3 根据程序特性，应能在有限合理的时间范围内生成（离线的）口令生成器的大量输入-输出数据，进行仿真，返回仿真结果。

7.4.4 根据程序特性，应能在有限合理的时间范围内对注入木马/病毒前后程序版本进行对比，提供程序对比报告。

7.4.5 根据程序特性，应能在有限合理的时间范围内生成算法程序的大量输入-输出数据，进行仿真，返回仿真结果。

7.4.6 应测试是否存在测试用例可能使得程序陷入无限循环（或运行时间不合理），并在检测出异常后及时退出测试进程，避免资源的浪费。

8. 无源码的白盒化测试报告规范

完成无源码的白盒化测试后应出具相应的白盒化测试报告，白盒化测试报告应包括下列内容：

a） 白盒化测试报告表；

b） 总检汇总分析；

c） 被测软件信息；

d） 测试环境信息；

e） 测试用例执行情况统计表；

f） 测试附件信息。

其中白盒化测试报告表为整份测试报告的汇总信息表。总检汇总分析为所有无源码的白盒化测试类型的测试结果汇总表，包括每个白盒化测试类型的评价等级、关键信息、错误描述和整体测试的综述小结。被测软件信息主要描述被测软件的各项关键信息数据。测试环境信息主要描述本次白盒化测试所使用的测试环

境的各项关键信息，方便后续对测试结果进行复现与确定，内容应区分软件环境与硬件环境。测试用例执行情况统计表主要描述不同白盒化测试类型的用例执行情况。

另外软件附件信息主要描述不同白盒化测试类型的具体测试过程信息和结果信息，应包含的内容有：单元测试结果附件、缺陷检测结果附件、功能性结果附件、性能效率结果附件。