基于持续集成的冒烟测试

姜 文,刘立康
(西安电子科技大学 通信工程学院,陕西 西安 710071)

摘 要:随着软件开发技术的发展,软件持续集成与自动化测试已成为软件开发过程中的一个重要组成部分。集成构建中的自动化测试实质上就是冒烟测试。冒烟测试是对软件版本包的基本功能进行测试验证,同时也是进一步开展全面深入测试的预测试。结合工作实践,介绍了冒烟测试、门槛用例和基于持续集成冒烟测试的特点;叙述了冒烟测试涉及到的角色和软件测试自动化工厂;详细叙述了冒烟测试运行的系统架构和运行流程。最后介绍了一个冒烟测试的工作案例和在该案例测试过程中遇到的一些典型问题。实践表明,基于持续集成的冒烟测试有助于及早发现并解决软件缺陷,提高软件开发效率和软件质量;采用自动化测试脚本进行软件测试,提高了测试效率,减少了测试工程师大量的重复测试验证工作。

关键词:持续集成;ICP-CI;冒烟测试;自动化工厂;版本包
中图分类号:TP311.5 **文献标识码**:A **文章编号**:1673-629X(2018)08-0053-05
doi:10.3969/j.issn.1673-629X.2018.08.011

Smoking Test Based on Continuous Integration

JIANG Wen,LIU Li-kang
(School of Telecommunication Engineering,Xidian University,Xi'an 710071,China)

Abstract:With the development of the software technology,continuous integration and automation test has become an important part in the process of software development. Automation test in integration building is essentially smoking test which tests the basic function of the software version package,also pre-tests in further in-depth test. Combined with practice,we introduce the smoking test,threshold cases and the characteristics of smoking test based on continuous integration,describe the roles involved in smoking test and the software test automation factory,especially the system structure and operation process of smoking test in detail. Finally we give a working case of smoking test and some typical problems in the testing process. Practice shows that smoking test based on the continuous integration helps to detect and solve software defects earlier,improving software development efficiency and quality. Using automated test scripts in software testing can improve testing efficiency and reduce a lot of repetition test validation for testing engineer.

Key words:continuous integration;ICP-CI;smoking test;automated factory;version package

0 引 言

随着软件敏捷开发与测试驱动开发理念的不断普及与发展,持续集成[1-6]在软件产品开发过程中的地位越来越重要。为了保证及时发现每天合入配置库的代码质量,使用持续集成工具开展每日集成构建工作,进行源代码更新、基础静态检查、软件产品模块编译、软件版本包出包以及产品版本包的自动化测试。版本包的自动化测试就是通常所讲的"冒烟测试"。

冒烟测试是一种测试策略,是对集成构建生成的版本包进行测试验证,同时也是进一步开展全面深入测试的预测试。开展冒烟测试工作有助于尽早发现软件代码存在的问题,提高软件代码的质量和开发效率。基于持续集成的冒烟测试采用自动化测试脚本进行测试工作,能够提高测试效率,减少测试人员大量的重复测试验证工作。

文中介绍了冒烟测试、门槛用例;叙述了基于持续集成的冒烟测试的特点、冒烟测试涉及到的角色和软件测试自动化工厂;详细叙述了冒烟测试运行的系统架构和运行流程。最后介绍了一个冒烟测试的工作案例进行说明。

收稿日期:2017-10-23 **修回日期**:2018-02-27 **网络出版时间**:2018-04-28
基金项目:国家部委基础科研计划:国防预研基金项目(A1120110007)
作者简介:姜 文(1986-),女,工程师,硕士,CCF 会员(E200032324M),研究方向为图像处理与分析、数据库应用和软件工程;刘立康,副教授,研究方向为数字通信、图像传输与处理、图像分析与图像识别等。
网络出版地址:http://kns.cnki.net/kcms/detail/61.1450.TP.20180427.1644.078.html

1 冒烟测试的特点

1.1 冒烟测试

冒烟测试[7-13]是在软件开发过程中的一种针对软件版本包的快速基本功能验证策略,是对软件基本功能进行确认验证的手段,并非对软件版本包的深入测试。冒烟测试也是针对软件版本包进行详细测试之前的预测试,执行冒烟测试的主要目的是快速验证软件基本功能是否有缺陷。如果冒烟测试的测试例不能通过,则不必做进一步的测试。进行冒烟测试之前需要确定冒烟测试的用例集,对用例集要求覆盖软件的基本功能。这种版本包出包之后的验证方法通常称为软件版本包的门槛用例验证。

冒烟测试属于 HLT(high level test)测试,HLT 通常指 SDV(系统设计验证)/SIT(系统集成测试)/SVT(系统验证测试)等测试活动。HLT 是站在系统的角度对整个版本进行测试,测试对象是一个完整的产品而不是产品内部的模块,常见的 HLT 测试包括系统测试和验收测试。

冒烟测试可以手动执行,也可以自动化执行。稳定的系统适合自动化冒烟测试,集成过程中的系统适合手工冒烟测试,因为冒烟测试内容在动态变化,变化中的自动化脚本维护工作量比较大。

1.2 门槛用例

门槛用例测试,首先需要确定一个测试用例集作为门槛用例集,对软件版本包进行旧包卸载、新包部署,采用门槛用例集中的测试用例开展基本功能点的测试验证工作。软件版本包只有通过门槛用例测试才可以进行下一步开发或者测试工作。反之如果门槛用例测试有执行失败的用例,需要这个测试用例集的执行责任人第一时间进行用例失败分析,如果不是环境原因与脚本原因导致的用例失败,则需要相关模块的开发人员对问题进行定位,解决之后重新进行测试验证,直到通过为止。门槛用例测试流程如图 1 所示。

1.3 基于持续集成冒烟测试的特点

持续集成工具搭建的构建工程可以每天通过制定定时任务来自动完成从版本库更新代码、静态检查、编译、出包、自动化用例测试等任务。持续集成中的自动化用例测试就是通常所说的冒烟测试。冒烟测试是通过自动化工厂执行自动化脚本来完成测试任务。可以分为两种情形:

(1)每日构建生成增量版本包基线,冒烟测试主要是验证新增的软件代码是否影响前一天软件版本的基本功能,特别是对原有软件代码的修改。

(2)集成构建进行全量编译,生成转测试版本包基线,通常一周进行一次。冒烟测试除了验证以前版本包的基本功能外,也为新增功能测试进行了预测试。通过对新增功能测试完成后,可以向自动化工厂提交新的自动化脚本,从而增加自动化测试用例数量。

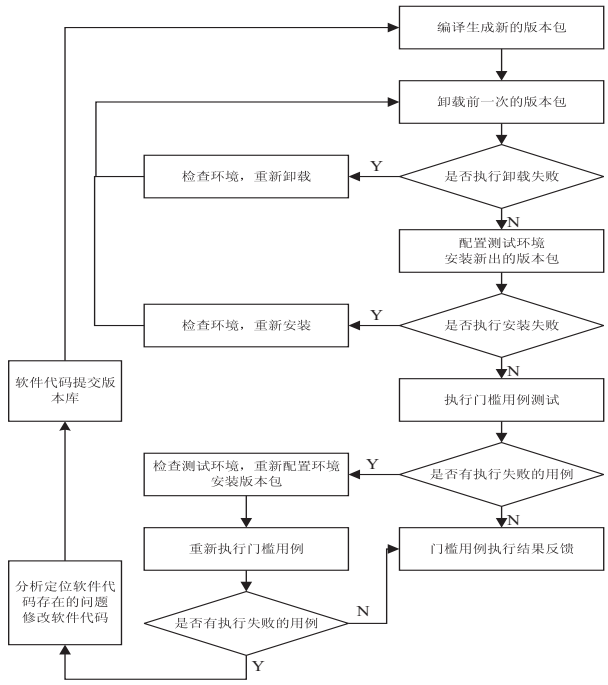

图 1 门槛用例测试运行流程

2 冒烟测试涉及到的角色

冒烟测试在测试环境搭建与执行过程中,涉及到的人员包括:测试架构师、管理自动化工厂的测试工程师、开发工程师、持续集成工程师、质量工程师。在冒烟测试环境搭建与执行过程中,以上各角色各司其职,分工协作共同确保冒烟测试正常执行,保证软件测试版本包能够通过基本功能验证,确保每天都有可用的基线版本提供给开发工程师和测试工程师。

2.1 测试架构师

测试架构师(TSE)根据产品目前所有的特性,经过对各特性测试用例的分析,提供冒烟测试的测试用例集。测试用例集的要求是覆盖目前产品所有特性的基本功能,并随着产品迭代开发进度,不断对测试用例集中所包含的测试用例进行调整,确保冒烟测试用例集不断覆盖新开发的特性。关注软件代码覆盖率质量指标,覆盖率质量指标达不到要求,需要查找原因,适时增加门槛用例数量。冒烟测试失败,参与分析定位处理,关注测试用例设计中存在的各种问题。

2.2 测试工程师

测试工程师根据冒烟测试用例集中的测试用例编写与调试自动化脚本,将脚本合入自动化工厂。冒烟测试失败,参与分析定位处理,重点关注自动化脚本的问题。

2.3 开发工程师

当出现冒烟测试执行失败的情况时,管理自动化

工厂的测试工程师进行脚本失败原因分析之后,排除了测试环境与环境配置的原因,就需要开发工程师根据问题现象和收集到的日志文件对失败的情况进行分析定位。如果定位结论是软件代码缺陷导致的冒烟测试失败,需要管理自动化工厂的测试工程师在缺陷管理系统提交问题单,开发工程师通过修改软件代码并将源代码合入版本库。对于一些复杂的问题需要软件系统架构师参与定位处理。

2.4　持续集成工程师

持续集成工程师从版本库更新源代码,搭建集成构建工程;进行静态检查、软件产品模块编译、软件版本包出包;向自动化工厂提交冒烟测试任务。根据自动化工厂的测试结果,通过邮件将信息反馈给相关人员。冒烟测试失败,从各个环节查找失败的原因,参与问题的分析定位处理。对接公司质量部门度量页面,使软件产品每天都有冒烟测试的相关数据能够呈现到公司质量部门度量页面。

2.5　管理自动化工厂的测试工程师

该测试工程师负责自动化工厂管理、运行和维护。自动化工厂收到冒烟测试任务后,需要完成软件版本包的安装和冒烟测试环境搭建;自动化脚本的部署;调度执行冒烟测试。完成冒烟测试任务后,将测试结果提交给持续集成工程师。当冒烟测试失败时,该测试工程师需要查明哪些测试脚本失败,参与脚本失败原因分析定位,需要特别关注测试环境的搭建和自动化脚本的部署。

2.6　质量工程师

质量工程师(RQA)需要定期根据公司质量部门对软件产品冒烟测试相关度量指标[14]进行审计,关注度量指标主要有测试成功率、执行自动化测试步骤的时间、自动化测试用例数目、成功的用例数、运行阻塞用例数、运行失败用例数;同时关注在开发阶段这些质量指标的变化。通过软件代码覆盖率质量指标监测门槛用例数量变化。

图 2 为应用软件冒烟测试用例图。

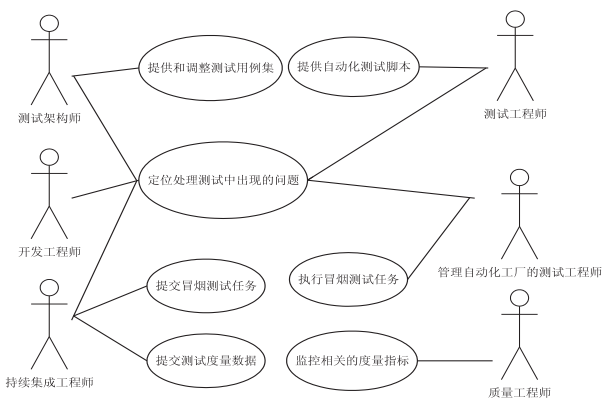

图 2　软件冒烟测试角色用例图

3　软件测试自动化工厂

随着软件版本迭代开发的进度,自动化测试用例与脚本的数量将不断增加,测试环境日趋复杂。自动化测试用例与脚本、测试环境需要由专职的测试工程师对自动化工厂[15-16]进行管理和维护。

以某个大型企业的软件自动化工厂为例叙述自动化工厂的组成。该自动化工厂由四个功能模块组成:测试任务执行控制中心(test integration control center, TICC)、测试资源管理中心(lab configuration manager, LCM)、自动化测试执行平台(test execution platform, TEP)、测试信息管理中心(test management service system, TMSS)。自动化工厂通常管理运行多个软件项目的自动化测试。

3.1　测试任务执行控制中心(TICC)

TICC 负责测试任务执行的调度和申请测试环境资源;相关测试信息的接收和传递。

3.2　测试资源管理中心(LCM)

LCM 是实验环境配置信息的统一管理平台,可实现环境录入、查询、修改、环境信息保存。负责所有测试资源的整合、归并与统一管理,为 TICC 的测试任务分配相关的测试资源。通常创建新版本的门槛用例测试集时,需要在 LCM 进行测试环境配置与统一管理。进行门槛用例测试需要在 LCM 上分配专用测试环境。

3.3　自动化测试执行平台(TEP)

TICC 通过 TiccAgent 工具控制自动化测试任务的下发,TEP 负责执行从 TICC 下发的自动化测试任务,执行任务的 PC 上安装自动化测试工具,执行门槛用例自动化脚本,脚本的语言为 TCL。

3.4　测试信息管理中心(TMSS)

TMSS 负责测试用例、自动化测试脚本以及测试任务相关信息的管理;测试执行结果和测试执行时长等测试执行过程中产生的中间信息都会被该系统自动记录。门槛测试用例集需要在 TMSS 上创建一个以版本号_CI 的测试用例集和测试用例自动化脚本集。测试人员可以在 TMSS 上创建测试任务,下发到 TICC,进行自动化测试脚本的测试工作。

4　基于持续集成的冒烟测试

采用持续集成工具 ICP-CI 搭建集成构建工程,通过该自动化工厂开展软件版本包的冒烟测试。基于持续集成的冒烟测试通常由以下几部分组成:制定门槛测试用例集;门槛用例自动化脚本编写与入厂;门槛用例测试环境搭建;集成构建工程任务管理页面配置冒烟测试任务;冒烟测试任务执行。

冒烟测试运行的系统架构如图 3 所示。

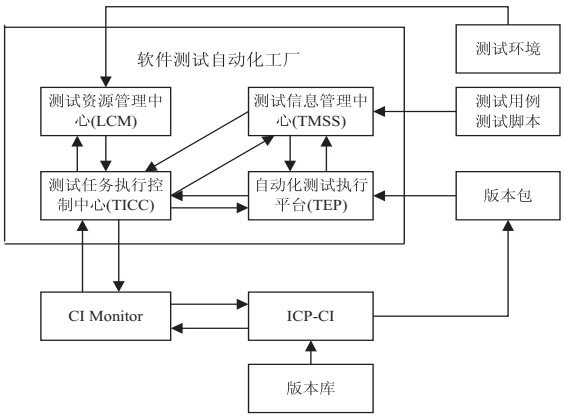

图 3 基于持续集成的冒烟测试系统架构

4.1 制定门槛测试用例集

在软件开发过程中,测试架构师根据目前软件产品的各特性与功能模块确定门槛测试用例集,经过评审之后,这些用例由测试架构师导入 TMSS 下以版本号_CI 命名的测试用例集中,并根据软件开发的情况不断调整该测试用例集中的测试用例。

4.2 门槛用例自动化脚本编写与入厂

门槛用例集确定之后,测试工程师完成门槛用例脚本的编写与调试。测试架构师进行脚本评审,脚本评审通过后,测试工程师通过配置管理工具 Git 将门槛用例脚本合入配置库,管理自动化工厂的测试工程师将脚本从配置库中导入到 TMSS 下以版本号_CI 命名的测试用例脚本集,和测试用例集中的测试用例一一对应。

4.3 门槛用例测试环境搭建

管理自动化工厂的测试工程师编写与调试卸载旧版本包、安装新版本包与环境配置的脚本,搭建与配置测试环境。测试包括执行 PC、产品软件环境。对于云环境,虚拟 PC 机通过 VMware 软件来制作,通常制作虚拟机的镜像文件,操作系统是 Windows 2003 Server。虚拟机的规格是系统盘 20 G,数据盘 40 G。管理自动化工厂的测试工程师需要在 LCM 上已分配给冒烟测试的环境进行搭建与配置。

4.4 集成构建工程任务管理页面配置冒烟测试任务

使用 ICP-CI 工具在集成构建工程的任务管理页面配置冒烟测试任务,需要配置软件产品的自动化测试用例库名称、冒烟测试服务管理系统名称、冒烟测试端口号(通常为 8080),控制冒烟任务下发的配置文件放置在 ICP-CI 主控服务器的构建工程文件夹下和 CIMonitor 工具路径下的 VersionReleaseInfo. ini 文件中。

VersionReleaseInfo. ini 文件的配置内容如下所示:

[Summary_Info]
IssueTime=2017-03-19_03.00.11
cVersion=产品名称 版本号_CI
bVersion=产品名称 版本号_CIB000 //持续集成冒烟测试用例库
IssueStatus=0
VerDescription=版本号
CmsType=DailyTest
Priority=0
[STD_LEM_SRV]
Version=版本号
Directory=版本包路径 //出包任务完成之后,需要将版本包拷贝到该路径下
Filename=版本包名称

4.5 冒烟测试任务执行

ICP-CI 工具通过执行相应的 ANT 脚本,将出包步骤生成的版本包拷贝到 Directory 参数配置的路径下,通过以下步骤来完成冒烟测试任务。

(1)从 ICP-CI 工具下发冒烟测试任务,更新 VersionReleaseInfo. ini 配置文件的 IssueTime,触发 CIMonitor 工具。

(2)CIMonitor 工具将冒烟测试任务提交到 TICC,根据 bVersion 参数(冒烟测试用例库)配置,TICC 从 TMSS 中查找测试该用例集所需要的资源信息,然后 TICC 根据资源分配策略从 LCM 申请测试资源。

(3)TICC 将任务下发至 TEP。

(4)TEP 接受任务后,调用分配到的测试资源,自动执行测试,承担测试任务的执行机卸载试环境中旧的软件版本包,安装本次编译生成的软件版本包。从 TMSS 下载测试用例脚本,开始执行测试工作;测试工作结束后,将测试结果上传给 TICC;所有过程信息和最终测试结果自动保存储到 TMSS 中。

(5)TICC 通过 CIMonitor 工具反馈测试结果到 ICP-CI 的测试结果页面,ICP-CI 工具通过邮件向相关人员发送测试结果信息。

4.6 ICP-CI 工具与自动化工厂的对接与调试

管理自动化工厂的测试工程师检查确认自动化工厂环境状态正常后,持续集成工程师在集成构建工程上勾选冒烟测试任务,启动冒烟测试任务。两人共同观测以下检查点:

(a)测试任务是否能够正常下发到 TICC;

(b)测试任务是否能够正常卸载旧的软件版本包;

(c)测试任务是否能够正常安装新的软件版本包;

(d)测试任务是否能够执行通过全部门槛用例;

(e)测试任务结束后是否能够将测试结果反馈到 ICP-CI 工具的页面上。

以上检查点中,检查点 a 和 e 出现问题由持续集成工程师定位解决;检查点 b、c 和 d 出现问题由管理自动化工厂的测试工程师定位解决。

5　典型案例

5.1　案例介绍

某公司有一个软、硬件结合的中型软件开发项目,总的代码量大约三百万行。采用持续集成工具每天开展集成构建工作。在软件项目开发初期,测试组由专人负责自动化工厂的测试工作。随着软件开发工作的进展,门槛测试用例集中的测试用例数量随着软件特性数量的增加而不断调整,门槛测试用例数增加到 50 多个,软件测试环境也趋于稳定。自动化工厂运行成熟(测试环境与软件版本包稳定)后,项目测试组将自动化工厂的测试工作移交给部门的自动化工厂管理组。该管理组统一开展部门内多个软件项目的自动化工厂的测试工作,进一步优化门槛用例脚本,提升测试效率和质量。

5.2　典型问题的处理

在软件开发过程中,处理了许多冒烟测试的技术问题。下面介绍几个具有代表性的问题。

5.2.1　代码修改导致测试失败

编写脚本的测试工程师定位之后,发现由于软件特性有变更,开发工程师修改合入了代码之后,并没有及时知会测试工程师,导致测试工程师没有及时修改入厂脚本,导致脚本运行失败。测试工程师修改入厂脚本之后,门槛用例执行成功。

5.2.2　测试脚本导致测试失败

某次冒烟测试出现多处失败,编写脚本的测试工程师定位之后,发现由于有一个脚本运行时,需要通过脚本打开一个比较特殊的调试开关,脚本运行完成之后没有关闭调试开关,导致后续相关的脚本运行失败。确认问题之后,测试工程师修改入厂脚本,门槛用例执行成功。

5.2.3　测试环境配置导致测试失败

编写脚本的测试工程师定位之后,发现门槛测试用例环境上有相关的 MML 配置有缺失,导致依赖这个 MML 配置的脚本都执行失败了,确认问题之后,管理自动化工厂的测试工程师重新添加了相关 MML 配置之后,门槛用例执行成功。

软件产品开发过程中引入基于持续集成的冒烟测试之后,先后发现与拦截了多个软件缺陷。工作实践表明,执行冒烟测试有助于及早发现并解决软件缺陷,提高软件迭代开发阶段软件开发与测试的效率,便于产品主管了解工作进度和解决存在的问题。

6　结束语

集成构建工程完成软件版本出包之后,进行冒烟测试可以快速对版本包进行测试验证,及时向开发工程师和测试工程师反馈软件功能特性验证信息。发现软件缺陷,开发工程师能够尽快修复这些缺陷,有效避免大量的缺陷在软件开发的某个阶段集中爆发,同时也为下一步软件代码的开发工作提供了基线版本。长期的工作实践表明,在软件的开发过程中采用基于持续集成的冒烟测试,可以提高软件的质量和开发效率,降低软件的开发成本;采用自动化测试脚本进行测试工作,提高了测试效率,减少了测试人员大量的重复测试验证工作;同时也有助于做好软件项目的管理工作。

参考文献:

[1] DUVALL P M,MATYAS S,GLOVER A. 持续集成软件质量改进和风险降低之道[M]. 北京:电子工业出版社,2012.

[2] 姜　文,刘立康. 基于 SVN 的软件配置管理和持续集成[J]. 电子设计工程,2016,24(2):1-5.

[3] 姜　文,刘立康. 基于 ClearCase 的软件配置管理与持续集成[J]. 计算机技术与发展,2016,26(1):10-17.

[4] 姜　文,刘立康. 基于 SVN 的应用软件持续集成[J]. 计算机测量与控制,2016,24(3):109-113.

[5] 姜　文,刘立康. 软件配置管理中的基线问题研究[J]. 计算机技术与发展,2016,26(6):6-10.

[6] SMART J F. Jenkins:the defintive guide[M]. Sebastopol,California:O'Relly Media. ,Inc. ,2011.

[7] MYERS G J,BADGETT T,SANDLER C. Art of software testing[M]. 3rd ed. Hoboken,New Jersey:John Wiley & Sons,Inc,,2012.

[8] 买志玉,韩玉民. 软件测试实践教程[M]. 北京:清华大学出版社,2015.

[9] 肖利琼. 软件测试之魂核心测试设计精解[M]. 第 2 版. 北京:电子工业出版社,2013.

[10] MCCONNELL S. Daily build and smoke test[J]. IEEE Software,1996,13(4):143-144.

[11] CHAUHAN V K. Smoke testing[J]. International Journal of Scientific and Research Publications,2014,4(2):1-5.

[12] 刘兴宇. 冒烟自动化测试系统的设计与实现[D]. 北京:北京交通大学,2014.

[13] 武孟梦. 网络管理系统自动化测试的应用与实现[D]. 武汉:华中科技大学,2010.

[14] 姜　文,刘立康. 基于持续集成的软件度量[J]. 计算机测量与控制,2017,25(5):136-139.

[15] 张红亮. Web 自动化测试效率的优化方法与实现[D]. 南京:东南大学,2014.

[16] 周景才,张沪寅,查文亮,等. 云计算环境下基于用户行为特征的资源分配策略[J]. 计算机研究与发展,2014,51(5):1108-1119.