基于STAF框架下的自动化测试

张 磊,王晓军
(南京邮电大学 信息网络研究所,江苏 南京 210003)

摘 要:当前很多大型软件都不是采用单一的实现技术,面对于不同的实现技术,要求测试人员在不同的平台下分别对所测试的部分进行测试。为了提高自动化测试的效率,减少测试开销,提出了一种基于STAF的自动化测试平台。首先对目前现有的自动化测试框架归纳分析;然后提出基于STAF的自动化测试平台的系统架构;最后给出平台的关键部分:测试引擎的实现方式。结果表明该平台具有提高自动化测试效率,减少测试开销,支持分布式测试方法等优越性。

关键词:自动化测试平台;软件测试自动化框架;测试引擎
中图分类号:TP311.56 **文献标识码**:A **文章编号**:1673-629X(2010)03-0116-05

Automated Testing Based on STAF Framework

ZHANG Lei, WANG Xiao-jun
(Inst. of Info. Network, Nanjing Univ. of Posts and Telecommunications, Nanjing 210003, China)

Abstracts: Currently, many large softwares are not using a single implementation technology. When face on different implementation techniques, required to testers would tested were part of the test on the different platform. This article is to improve the efficiency of automated testing, reduce testing overhead, is proposed based on STAF (Software Test Automation Framework) automated test platform. First of all, analyses current framework for automated testing. Then puts forward an automated test platform system architecture based on the STAF. Finally, presents a key part of the platform: testing engine. The results show that the platform is to improve the efficiency of automated testing, reduce testing overhead, to support distributed testing methods.

Key words: automated test platform; STAF(software test automation framework); test engine

0 引 言

当前软件测试在软件开发生命周期中发挥着越来越重要的作用。伴随着软件规模的不断增大,软件对于运行时环境的要求也日益复杂,这意味着测试人员为了运行测试用例必须花费更多的时间用于准备测试环境。自动化测试通过减少在测试过程中需要人工操作的步骤,使测试人员得以分配更多的时间用于产生测试工作核心价值的行为,如寻找软件错误和缺陷、验证业务逻辑等。提高软件测试环节中的自动化程度,对提高整个软件测试环节的生产率具有重要的意义。

1 自动化测试框架概述

1.1 自动化测试框架的定义

一个自动化测试框架就是一个由假设、概念以及为自动化测试提供支持的实践的集合。自动化测试框架可以减少测试脚本实现和维护的成本,使测试人员把精力集中在测试用例的设计上。自动化测试框架的好坏直接影响到自动化测试的成功与否。

1.2 自动化测试框架的优点

为了提高测试效率,越来越多的测试工作引入了自动化测试的思想和方法。实践证明,软件自动化测试技术提高了软件测试的速度和效率,节省了软件测试成本,缩短了产品发布周期。同时,自动化测试技术也完成了许多手工测试无法实现的工作。所以,采用自动化测试方法和相应的测试框架成为了软件开发组织测试工作的重要支撑手段。例如,采用自动化测试工具能在测试活动中减少一部分开销,同时,有些测试活动是靠手工方式难以实现和度量的;自动化测试框架能够提高测试效率,快速定位测试软件各版本中的功能、性能缺陷。

1.3 自动化测试框架的开发原则

(1)测试框架与被测应用程序独立。虽然测试的应用程序不一样,但被测应用程序之间却会有相同的地方,测试框架应聚焦在不同测试应用程序中共同的

收稿日期:2009-07-05;**修回日期**:2009-10-30
作者简介:张 磊(1983-),男,新疆乌鲁木齐人,硕士研究生,研究方向为计算机网络与分布式计算;王晓军,硕士研究生导师,研究方向为面向服务的软件体系结构、业务流程管理和协同,特别是对Web服务以及基于Web服务的应用支撑环境的研究等。

部分,把与具体应用程序有关的部分从框架中移除。

(2)测试框架应易于扩展、维护。测试框架应被高度模块化,这样可以提高框架的维护性。各个模块之间相互独立,对模块内部的修改不应该影响其他模块。除此之外,系统应该有充足、详细的文档,与软件开发一样,这也是必不可少的。设计文档可以帮助开发人员扩展、维护测试框架,而使用文档则可以告诉用户要怎么使用该框架。

(3)测试脚本所使用的测试语言应该是与框架独立的。不同的测试框架可能在不同的应用领域有不同的表现,有些适用于Java应用程序的测试,有些可能适用于Web应用程序的测试,如果测试脚本所采用的语言是私有的、与测试框架绑定的,那么当需要从一个测试框架迁移到另外一个测试框架时,所有的测试脚本都需要重写。

(4)测试框架不应该让框架的复杂性影响到测试人员。在大多数情况下,测试人员就是测试人员而不是开发人员,甚至有的时候,他们不是专业的测试人员,可能只是具有很少软件开发经验的某个应用领域的专家。对于这些使用者来说,测试框架的使用要简单,测试语言要易于理解,这样可以使他们专注于业务相关内容的编写。

1.4　自动化测试框架的发展以及一些典型对比

文献[1]实现了一种测试数据和测试驱动分离的自动化测试框架,文献[2]则提出了由测试用例生成自动化测试脚本的方案,它们均可以提高测试用例的复用率,从而减少开发、维护测试用例的成本,但由于它们的底层测试驱动模块都使用脚本语言Perl来开发,所以只适用于单机测试环境。文献[3]设计了一种C/S模式的自动化测试网络模型,实现了对自动化测试任务的远程执行和监控,但在此系统执行的自动化测试任务必须使用CORBA实现通信功能,这无疑增加了测试工作的复杂度,而且测试任务也很难被复用。文献[4]提出了ESTP,一种通过表格规范而开发的自动化产生测试数据的软件测试平台,通过基于表格的分离策略,决策表测试,有意义影响策略,以及变异产生和报告分析来产生测试用例。文献[5]提出了ATS,这是一种利用J2EE技术,构建了一个通用的、与业务平台无关的软件自动化测试与开发的综合平台。还有一些常用的自动化测试框架,比如:

(1)测试脚本模块化框架(The Test Script Modularity Framework):通过创建小的独立的脚本来代表被测试应用程序的模块和函数,然后用一种分层的方式将这些小脚本组成更大的测试,从而实现一个特定的测试用例。

(2)测试库构架框架(The Test Library Architecture Framework):测试库构架框架和测试脚本模块化框架非常相似,有着同样的优势,但是它把要测试的应用程序分成过程和函数,而不是脚本。这种框架要求创建代表测试下应用程序模块、零件和函数的库文件(SQABasic libraries, APIs, DLLs等等)。然后这些库文件被测试用例脚本直接调用。

(3)数据驱动测试框架(The Data-Driven Testing Framework):将数据驱动脚本技术运用到自动化测试框架中就形成了数据驱动测试框架。这种框架从数据文件(数据池,ODBC源,CVS文件,Excel文件,DAO对象等)中读取输入和输出的测试数据,然后通过变量传入事先录制好的或手工编写的测试脚本中。

(4)关键字驱动或表驱动测试框架[6](The Keyword-Driven or Table-Driven Testing Framework):关键字驱动和表格驱动测试是一种独立于应用程序的自动化框架,它们是可以互相替换的术语。它是对数据驱动自动化测试的有效改进和补充。这个框架需要开发出数据表和关键字,这些数据表和关键字独立于执行它们的测试自动化工具,并可以用来"驱动"待测应用程序和数据的测试脚本代码,使用自动化测试框架独立于应用程序。

2　基于STAF框架下的自动化测试

2.1　STAF的简要介绍

Software Test Automation Framework(STAF)是由IBM开发的开源、跨平台、支持多语言并且基于可重用的组件来构建的自动化测试框架[7]。它封装了不同平台和不同语言间通信的复杂性,提供了消息、互斥、同步、日志等可复用的服务,使用户可以在此基础上方便快速地构建自动化测试解决方案。STAF在功能级别实施服务调用,各个服务端点(称作STAF客户端)是对等的,从一个端点可直接调用另一端点(在另一台机器运行的程序)提供的服务。

STAF的特点:

1)对环境需求最小化(包括硬件和软件)。

2)在各种语言中都很容易使用,包括Java,C/C++,Perl,TCL,及命令行shell环境。

3)易于扩展,让用户能方便地创建一个服务插入到STAF体系中。

STAF比较适应需要构造复杂测试环境的场合,复杂测试环境通常是分布式的,通过STAF将测试任务分发到不同的测试环境去执行,可以方便测试机测试脚本和收集测试结果。

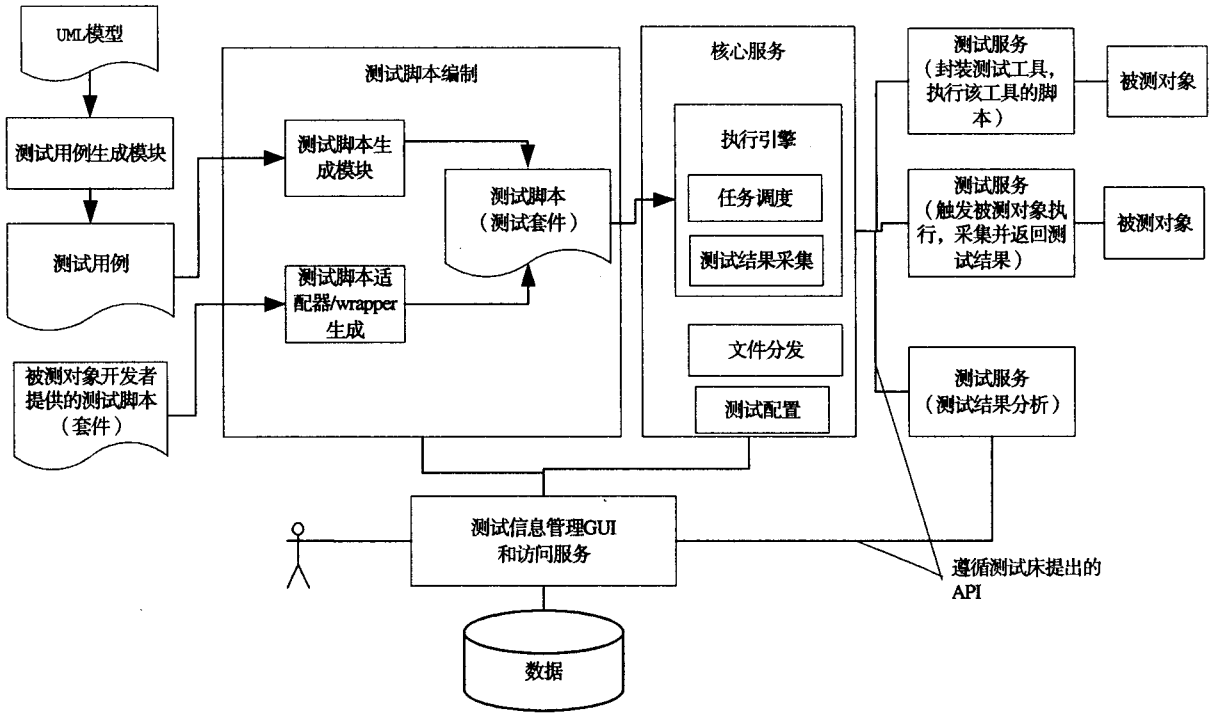


图 1 测试床概貌

2.2 基于 STAF 框架的自动化测试平台的系统架构

由于 STAF 的以上特点，又针对于复杂软件的多重实现技术，文中提出一种基于 STAF 框架的测试床。首先，用驱动测试引擎的方法调用不同的 STAF 服务来满足对于不同实现技术在同一平台内的测试；其次用 STAF 封装底层通信的特性，实现分布式测试；最后，调用 STAF 的 Log 服务来有效地记录测试结果供测试人员分析。其总体结构图如图 1 所示。

(1)测试用例的生成：有两个途径生成测试用例，一个是通过 UML 图产生测试用例，另一个是被测对象的开发者所提供的包含测试用例的测试脚本。

(2)测试脚本的生成：设计一个测试脚本生成模块以接收通过 UML 图产生的测试用例生成测试脚本。将被测对象的开发者提供的测试脚本放入一个测试脚本适配器生成可执行的一个测试脚本。

(3)测试引擎：此引擎是本测试床的核心部分，采用在 STAF 框架下，调用 STAF 的一个内部服务 STAX 的运行的方式来实现测试引擎的主要工作：任务的调度和测试结果的采集。对测试引擎的内部设计如图 2 所示。

脚本解析模块[8]：将得到的由测试用例产生的执行脚本解析为一个个的命令，每个命令被定义为(TS, DriverName, Keyword, Arguments, ExpectedResults, VariableNames)。

TS：测试用例执行次序编号。

DriverName：不同的开发技术对应于不同的测试方法，用来表现驱动何种测试方法的信息。

Keyword：定义一个基于关键字的测试原子操作。

Arguments：执行此步测试需要的入口参数。

ExpectedResults：此步测试的预期结果。

VariableNames：如需要的话，在测试执行中接收变量值。

一个命令序列由一个个命令所组成，脚本解析模块将脚本解析为一个个命令，然后再将命令存放在命令序列当中。

具体执行模块：在 STAF 环境下，具体的执行工作是调用 STAF 框架的一个内部服务 STAX，STAX 是基于 STAF 的执行引擎，它提供了一种 XML 格式的工作流语言。用户可以编写 XML 的脚本文件来通过 STAX 调用 STAF 的服务以完成自动化测试。用户可以不需要和编程语言打交道就可以开发出自己的自动化测试环境。STAX 提供如下的功能：支持并行运行，用户自定义的运行控制粒度，嵌套测试用例，控制运行时间，支持现有的 Java 和 Python 模块等。STAX 还提供了一个图形化的监控工具，通过这个工具，用户可以清晰地看出测试运行的位置、状态和出错信息等。

为了使用强大的 STAX 服务，首先，要将一些开源的测试工具封装为 STAF 可以识别的服务，就是创建 STAF 的外部服务。或者创建一个外部服务来触发 STU 的执行，并在 STAF 的配置文件中注册此服务，以便于以后要使用时的调用。其次，要根据脚本解析模块所产生的命令序列的信息，构建一个 STAX 可识

别和运行的驱动XML文件。

测试结果采集:在本平台中使用STAX中的分级日志功能来实现测试结果的采集,采集采用两种方式:一种是在控制端集中采集测试结果,测试人员可以直接在STAX Job Monitor上查阅测试结果;一种是在分布的测试代理上采集测试结果,并将测试结果输入到数据库当中。由测试人员到数据库中取测试结果,进行分析。

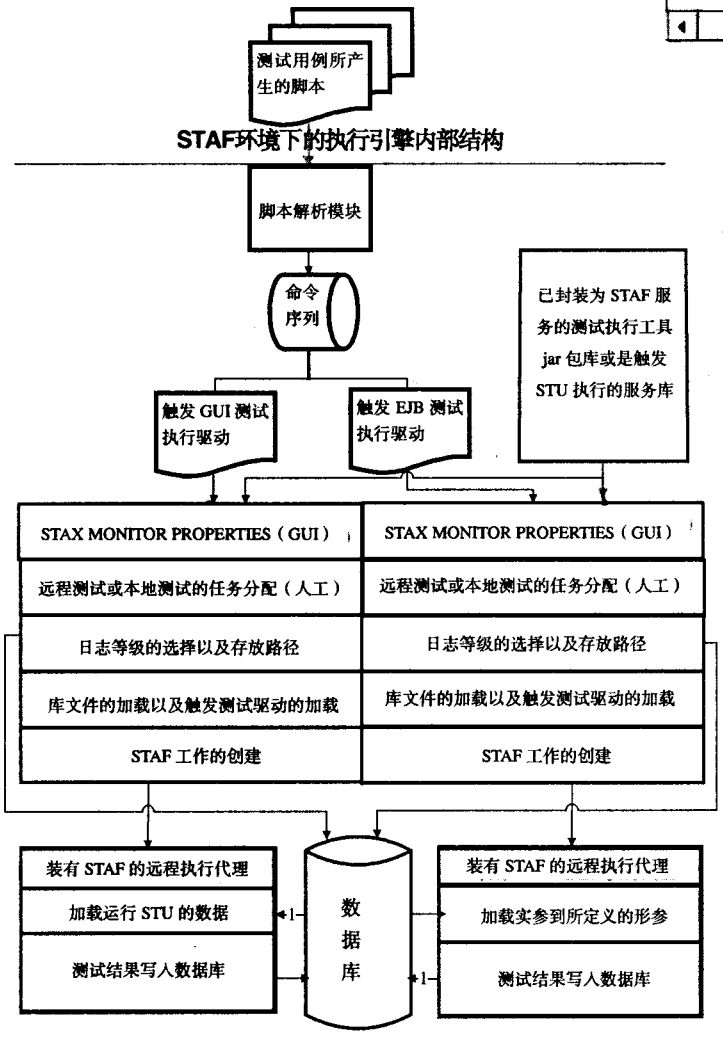


图2　测试引擎内部结构

3　基于STAF框架的自动化测试平台应用

通过以上介绍,实现一个监测本地TEST服务的一个应用,首先创建一个本地监测服务,然后在STAF配置文件中进行配置;其次,创建基于XML的STAX可识别的测试执行驱动,然后加载到STAX Monitor Properties中,最后在STAX Job Monitor中查阅测试结果。首先,启动STAFProc 3.3.3,如图3所示。

其次,通过命令行中输入:Java -jarSTAXMon.jar来启动STAX Monitor Properties,在此GUI窗口定义

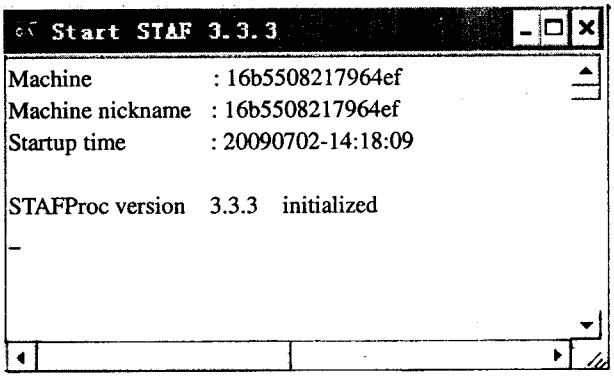


图3　启动STAFProc3.3.3

的内容:

1)是本地测试还是远程测试,如果是远程测试,给出远程测试的机器名(需要在STAF配置文件中说明信任等级等)。

2)日志所展现内容的选择以及在本机中存放的路径。日志所展现内容:Pass,Fail,Info,Status Data-Time,等。

3)已封装好放在外部服务库中的服务加载,已经驱动服务调用执行的XML文件加载。

然后执行测试,在测试过程中可以进行实时的监测以观看执行情况。

最后,在STAX Job Monitor中打开日志,查阅测试结果,如图4所示。

4　结束语

本测试平台是在IBM的开源测试框架STAF/STAX下提出的,具有很强的实用性。当面对不同开发技术,加载不同的测试驱动,使得测试人员关注于测试服务的封装以及驱动程序的编写,提高了驱动的可复用性,并且支持一种远程的分布式测试,以提高自动化测试的效率。

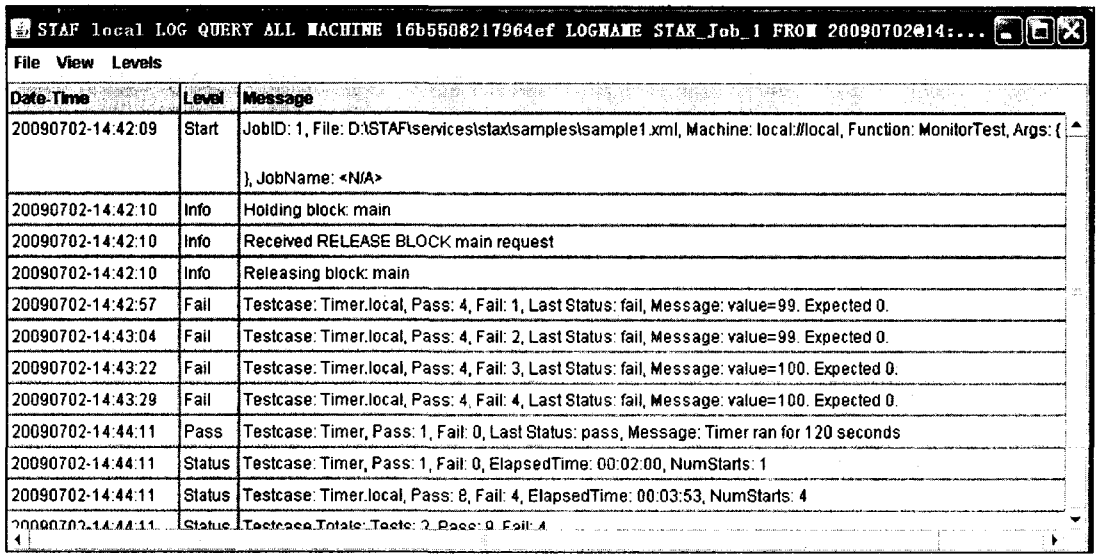
当然此自动化测试平台还有许多工作需要探讨。例如如何将回放工具封装为服务,来分布式地测试不同环境下的GUI测试,以及如何用脚本解析模块解析EJB测试脚本命令,通过后续研究使之成为一个具有通用机制的简单、易用的自动化测试方法。

参考文献:

[1] 朱　菊,王志坚,杨　雪.基于数据驱动的软件自动化测试框架[J].计算机技术与发展,2006,16(5):68-70.

[2] 朱　芳,李　曦,赵振西.一种多平台自动化测试工具的设计和实现[J].计算机工程,2004,30(24):186-188.

[3] 严少清,陈　革,万年红.软件测试自动化管理系统的设计


STAF local LOG QUERY ALL MACHINE 16b5508217964ef LOGNAME STAX_Job_1 FROM 20090702@14:...

File View Levels

Date-Time	Level	Message
20090702-14:42:09	Start	JobID: 1, File: D:\STAF\services\stax\samples\sample1.xml, Machine: local://local, Function: MonitorTest, Args: { }, JobName: <N/A>
20090702-14:42:10	Info	Holding block: main
20090702-14:42:10	Info	Received RELEASE BLOCK main request
20090702-14:42:10	Info	Releasing block: main
20090702-14:42:57	Fail	Testcase: Timer.local, Pass: 4, Fail: 1, Last Status: fail, Message: value=99. Expected 0.
20090702-14:43:04	Fail	Testcase: Timer.local, Pass: 4, Fail: 2, Last Status: fail, Message: value=99. Expected 0.
20090702-14:43:22	Fail	Testcase: Timer.local, Pass: 4, Fail: 3, Last Status: fail, Message: value=100. Expected 0.
20090702-14:43:29	Fail	Testcase: Timer.local, Pass: 4, Fail: 4, Last Status: fail, Message: value=100. Expected 0.
20090702-14:44:11	Pass	Testcase: Timer, Pass: 1, Fail: 0, Last Status: pass, Message: Timer ran for 120 seconds
20090702-14:44:11	Status	Testcase: Timer, Pass: 1, Fail: 0, ElapsedTime: 00:02:00, NumStarts: 1
20090702-14:44:11	Status	Testcase: Timer.local, Pass: 8, Fail: 4, ElapsedTime: 00:03:53, NumStarts: 4
20090702-14:44:11	Status	Testcase Totals: Tests: 2 Pass: 9 Fail: 4

图 4 日志查询界面

与实现[J]. 计算机工程,2002,28(9):152－157.

[4] Xin Feng, Marr S, O'Callaghan T. ESTP: An experimental software testing platform[C]//Testing: Academic and Industrial Conference Practice and Research Techniques, TAIC PART2008. [s.l.]:[s.n.],2008:59－63.

[5] PEI Songwen, WU Baifeng, ZHU Kun, et al. Novel Software Automated Testing System Based on J2EE[J]. Tsinghua Science and Technology,2007,12(s1):51－56.

[6] 接 卉,兰雨晴,骆 沛. 一种关键字驱动的自动化测试框架[J]. 计算机应用研究,2009,26(3):927－929.

[7] 李夏安,陈志泊. 基于 STAF 的软件自动化测试系统的研究和实现[J]. 计算机应用,2009,29(3):699－704.

[8] Tang Jingfan, Cao Xiaohua. Towards Adaptive Framework of Keyword Driven Automation Testing[C]//Proceedings of the IEEE International Conference on Automation and Logistics. Qingdao, China:[s.n.],2008:1631－1636.

(上接第 115 页)

线程实时控制软件的实时性能,对于多线程实时软件的开发具有实用价值。

```
elapsed_time = 153000013984
elapsed_time = 153002014496
elapsed_time = 153004014208
elapsed_time = 153006014432
elapsed_time = 153008014656
elapsed_time = 153010014080
elapsed_time = 153012013952
elapsed_time = 153014014496
elapsed_time = 153016014208
elapsed_time = 153018013920
elapsed_time = 153020014464
elapsed_time = 153022014144
elapsed_time = 153024014720
elapsed_time = 153026014432
elapsed_time = 153028014080
```

图 3 扫描耗时

参考文献:

[1] Hilton E F. Real－Time Applications with RTLinux[J/OL]. 2001 － 01 － 20. http://www. linuxjournal. com/article/4444.

[2] Ji Hua, Li Yan, Wang Jian. A software oriented CNC system based on Linux/RTLinux[J]. International Journal of Advanced Manufacturing Technology,2008,39(10):291－301.

[3] Li B, Zhou Yin－fei, Tang Xiao－qi. Research on open CNC system based on architecture/component software reuse technology[J]. Computers in Industry, 2004,55:73－85.

[4] 广 涛,施 华,尤晋元. 远程机器人控制系统的实时性能评估[J]. 计算机工程,2004,30(16):99－101.

[5] 陈晓明,王治森,董伯麟,等. 基于 Windows CE 5.0 的嵌入式数控系统实时性研究[J]. 工业仪表与自动化装置,2007(6):6－10.

[6] 李 江,戴胜华. Linux 操作系统实时性测试及分析[J]. 计算机应用,2005,25(7):1679－1680.

[7] 褚文奎,张凤鸣,樊晓光. 嵌入式 Linux 系统实时性能测试研究[J]. 系统工程与电子技术,2007,29(8):1385－1388.

[8] Stillerman J A, Ferrara M, Fredian T W, et al. Digital real－time plasma control system for Alcator C－Mod[J]. Fusion Engineering and Design,2006, 81:1905－1910.

[9] 朱响斌,涂时亮. Linux 的实时性能测试[J]. 微电子学与计算机,2004,21(11):85－88.

[10] 张 帆. 基于 RTLinux 的硬实时性研究[J]. 武汉理工大学学报:信息与管理工程版,2006,28(7):165－167.