

**<<LUA >>**

**测试需求说明书**



北京航空航天大学

2015-04

小组成员

|  |  |
| --- | --- |
| 名字 | 学号 |
| 黎功辉 | SY1406232 |
| 白瑞雪 | SY1406233 |
| 颜世增 | SY1406219 |

版本变更历史

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 版本 | 提交日期 | 主要编制人 | 审核人 | 版本说明 |
| 1.0 | 2015/2/26 | 全体 | 黎功辉 | 该文档的最初版本 |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

目 录

[1. 引言 4](#_Toc418690930)

[1.1 编写目的 4](#_Toc418690931)

[1.2 过程分析 4](#_Toc418690932)

[1.3 参考文献 5](#_Toc418690933)

[2. 测试计划 5](#_Toc418690934)

[2.1 目标 5](#_Toc418690935)

[2.2 测试手段 5](#_Toc418690936)

[2.2.1 单元测试 5](#_Toc418690937)

[2.2.2 集成测试 6](#_Toc418690938)

[2.2.3 确认测试 6](#_Toc418690939)

[2.2.4 测试工具 6](#_Toc418690940)

[2.2.4.1 Xavante 6](#_Toc418690941)

[2.2.4.2 Apache 7](#_Toc418690942)

[2.2.4.3 Nginx 7](#_Toc418690943)

[2.2.4.4 ngx\_lua 7](#_Toc418690944)

[2.2.4.5 PHP 7](#_Toc418690945)

[2.2.4.6 Python 8](#_Toc418690946)

[2.3 测试数据 8](#_Toc418690947)

[2.4 测试策略 8](#_Toc418690948)

[2.5 测试通过准则 9](#_Toc418690949)

[3. 功能测试以及功能测试点分析 9](#_Toc418690950)

[3.1 概述 9](#_Toc418690951)

[3.2 Lua 嵌入到其他应用 9](#_Toc418690952)

[3.2.1 搭建动态网页测试 10](#_Toc418690953)

[3.2.1.1 测试方案 10](#_Toc418690954)

[3.3 独立使用Lua 10](#_Toc418690955)

[3.3.1 文件复制 10](#_Toc418690956)

[3.3.1.1 测试方案 10](#_Toc418690957)

[3.3.2 只运行一次的小程序 11](#_Toc418690958)

[3.3.2.1 测试方案 11](#_Toc418690959)

[3.3.3 模式匹配测试 11](#_Toc418690960)

[3.3.3.1 测试方案 11](#_Toc418690961)

[3.4 Lua 和C++ 混合使用 12](#_Toc418690962)

[3.4.1 调用lua打印“hello world” 12](#_Toc418690963)

[3.4.1.1 测试方案 12](#_Toc418690964)

[4. 性能测试以及性能测试点分析 12](#_Toc418690965)

[4.1 Table性能测试 13](#_Toc418690966)

[4.1.1 测试方案 13](#_Toc418690967)

[4.2 字符串性能测试 13](#_Toc418690968)

[4.2.1 测试方案 13](#_Toc418690969)

[4.3 WEB应用的性能测试 13](#_Toc418690970)

[4.3.1 测试方案 14](#_Toc418690971)

# 引言

## 编写目的

本文档是系统初步验收过程中对测试进行分析的需求方案，其主要描述了测试方案、测试用例产生原因和项目中的测试用例设计说明等。测试人员可通过文档的测试方案结合测试用例，对系统进行测试。

本文档的读者范围是：

 LUA项目的测试技术人员，LUA项目的技术人员，LUA项目的需求分析人员。

## 过程分析

小组成员在讨论后决定主要通过对Lua功能和性能两方面的主要特征进行测试，测试计划的指定依据为《需求规格说明书》。由于Lua整个代码量较小，所以阅读源码后，我们不仅对功能进行了测试，还对Lua如何共性能的运作进行了测试。由于这些特征点较多，我们主要选取了最有代表性的特征进行测试。

为了使得测试用例更加规范，我们小组约定了几条在编写测试用例时需要遵守的几条规则：首先，测试用例的前置条件应该与对应需求用例中的条件相对应；其次，测试用例的主流程应该与对应需求用例的主流程相对应，至少包括用例中参与者与系统的交互过程；最后测试用例中应该给出明确的判断测试用例是否通过的标准。

## 参考文献

1) LUA需求说明书

2) LUA官方手册

3) LUA官方版本更新日志

# 测试计划

## 目标

通过系统测试检查LUA是否符合《需求说明书》的要求，程序是否具备《需求说明书》要求的功能，LUA是否实现了《需求说明书》所要求的脚本语言功能，在执行过程中LUA的性能是否符合要求，LUA中标准库是否实现了所需功能等。同时针对LUA在测试过程中发现的问题进行相应的回归测试，保证已发现问题已解决。

## 测试手段

## 单元测试

在需求规格说明书中描述的所有功能，都将在选定的LUA脚本或C++代码上进行测试。测试中需要考虑正常的功能实现和恶意测试，即使用设计者不期望的数据进行操作。

## 集成测试

集成测试的目的是确保各单元组合在一起后能够按既定意图协作运行，并确保增量的行为正确。它所测试的内容包括单元间的接口以及集成后的功能。使用黑盒测试方法测试集成的功能。集成测试中没完成的功能进行回归测试。

## 确认测试

确认测试又称为有效性测试。其任务是检查软件的功能和性能是否与需求说明书中确定的指标相符。确认测试阶段有进行确认测试与软件配置审查两项工作。

确认测试一般是在模拟环境下运用黑盒测试方法，由专门测试人员和用户参加的测试。确认测试需要需求说明书、用户手册等文档，要制定测试计划，确定测试的项目，说明测试的内容描述具体的测试用例。集成测试中没完成的功能进行回归测试。

软件配置审查的任务是检查软件的所有文档资料的完整性、正确性。如发现遗漏和错误，应补充和改正。同时要编排好目录，为以后的软件维护工作奠定基础。

## 测试工具

本测试工具主要说明，在以后的实际的测试过程中有可能用到的工具。

## Xavante

Xavante是一个 Lua HTTP 1.1 Web服务器，采用一个模块化的架构基于URI mapped 处理器。Xavante当前提供一个文件处理器，一个Redirect处理器和一个WSAPI处理器。

## Apache

Apache HTTP Server（简称Apache）是Apache软件基金会的一个开放源码的网页服务器，可以在大多数计算机操作系统中运行，由于其多平台和安全性被广泛使用，是最流行的Web服务器端软件之一。它快速、可靠并且可通过简单的API扩展，将Perl/Python等解释器编译到服务器中。

Apache HTTP服务器是一个模块化的服务器，源于NCSAhttpd服务器，经过多次修改，成为世界使用排名第一的Web服务器软件。

## Nginx

Nginx ("engine x") 是一个高性能的 HTTP 和 反向代理 服务器，也是一个 IMAP/POP3/SMTP 代理服务器。Nginx 是由 Igor Sysoev 为俄罗斯访问量第二的 Rambler.ru 站点开发的，第一个公开版本0.1.0发布于2004年10月4日。其将源代码以类BSD许可证的形式发布，因它的稳定性、丰富的功能集、示例配置文件和低系统资源的消耗而闻名。

## ngx\_[lua](http://www.ttlsa.com/monitor/lua/)

ngx\_[lua](http://www.ttlsa.com/monitor/lua/)是Nginx的一个功能模块，该模块的主要作用是把lua语言嵌入nginx中,使其支持lua来快速开发基于nginx下的业务逻辑

## PHP

PHP（外文名:PHP: Hypertext Preprocessor，中文名：“超文本预处理器”）是一种通用开源脚本语言。语法吸收了C语言、Java和Perl的特点，利于学习，使用广泛，主要适用于Web开发领域。PHP 独特的语法混合了C、Java、Perl以及PHP自创的语法。它可以比CGI或者Perl更快速地执行动态网页。用PHP做出的动态页面与其他的编程语言相比，PHP是将程序嵌入到HTML（标准通用标记语言下的一个应用）文档中去执行，执行效率比完全生成HTML标记的CGI要高许多；PHP还可以执行编译后代码，编译可以达到加密和优化代码运行，使代码运行更快。

## Python

Python（英语发音：/ˈpaɪθən/）, 是一种面向对象、解释型计算机程序设计语言，由Guido van Rossum于1989年底发明，第一个公开发行版发行于1991年，Python 源代码同样遵循 GPL(GNU General Public License)协议[1] 。Python语法简洁而清晰，具有丰富和强大的类库。它常被昵称为胶水语言，能够把用其他语言制作的各种模块（尤其是C/C++）很轻松地联结在一起。常见的一种应用情形是，使用Python快速生成程序的原型（有时甚至是程序的最终界面），然后对其中有特别要求的部分，用更合适的语言改写，比如3D游戏中的图形渲染模块，性能要求特别高，就可以用C/C++重写，而后封装为Python可以调用的扩展类库。需要注意的是在您使用扩展类库时可能需要考虑平台问题，某些可能不提供跨平台的实现。

## 测试数据

本测试实践的测试数据来源，主要有我们自己编写的LUA脚本和C++代码，以及LUA官方提供的一些性能方面的数据和部分测试结果。

## 测试策略

我们准备进行测试的过程为LUA系统测试，系统测试的测试策略如下：

系统测试的主要目的是在于验证软件的功能和性能及其他特性是否与用户的要求一致，主要是下列类型的测试:

1. 安装测试：确定LUA在各种操作系统（32位，64位）中都能够被正常编译，安装。
2. 功能测试：确定LUA是否完全的实现了《需求说明书》中的功能点。主要内容详见第三节。
3. 性能测试：确定LUA在一定条件下，性能的曲线图，包括强度测试，容量测试，安全性测试等。主要内容详见第四节。

## 测试通过准则

LUA系统通过测试的准则，即当依据测试用例执行者测试结果与预期结果相符，或测试结果与预期结果虽有不符但不可归咎于LUA本身时为测试通过，反之测试失败。

# 功能测试以及功能测试点分析

## 概述

功能测试就是对产品的各功能进行验证，根据功能测试用例，逐项测试，检查产品是否达到用户要求的功能

Functional testing（功能测试），也称为behavioral testing（行为测试），根据产品特性、操作描述和用户方案，测试一个产品的特性和可操作行为以确定它们满足设计需求。本地化软件的功能测试，用于验证应用程序或网站对目标用户能正确工作。使用适当的平台、浏览器和测试脚本，以保证目标用户的体验将足够好，就像应用程序是专门为该市场开发的一样。功能测试是为了确保程序以期望的方式运行而按功能要求对软件进行的测试，通过对一个系统的所有的特性和功能都进行测试确保符合需求和规范。

功能测试也叫黑盒测试或数据驱动测试，只需考虑需要测试的各个功能，不需要考虑整个软件的内部结构及代码.一般从软件产品的界面、架构出发，按照需求编写出来的测试用例，输入数据在预期结果和实际结果之间进行评测，进而提出更加使产品达到用户使用的要求。

## Lua 嵌入到其他应用

该方面测试主要是针对将lua嵌入到其他应用中的功能实现情况，根据软件需求说明书在该方面所提的需求，我们设计了如下一个功能测试点：搭建动态网页测试。

## 搭建动态网页测试

在LUA嵌入到其他应用方面，我们选取的功能测试点是搭建动态网页测试，LUA作为脚本语言可以嵌入到很多其他不同类型的应用程序中，比如WEB应用，因为WEB的发展极为迅速，在当今的所有应用中，系统软件和WEB应用是两个重要的方面，我们认为该功能测试点能够完全的测试出LUA在该方面所具有的所有功能，所以我们选择了该功能测试点。

搭建动态网页测试主要是测试在WEB页面中嵌入LUA代码。WEB页面包括HTML代码、CSS代码、JS代码。

## 测试方案

首先编写一个用于测试的简单的lua脚本。然后通过Xavante或者Apache等带有LUA模块的WEB服务器，运行起来，最后在浏览器中查看测试结果。进行结果对比是否符合预期结果。

## 独立使用Lua

该方面测试主要是针对独立使用lua的功能实现情况，根据软件需求说明书在该方面所提的需求，我们设计了如下三个测试功能点：文件复制，只运行一次的小程序，模式匹配。

## 文件复制

文件复制实际上也是一种只运行一次的小程序的一种，之所以把文件操作相关的内容拿出来，是关注一下Lua对一些其他文件的处理能力。Lua对文件的处理也提供了打开，关闭，读取，写入等一些操作，而文件复制过程中，能基本上把这些函数都用到，所以我们测试这个用例。

## 测试方案

编写文件复制的脚本，基本步骤是打开文件，读取内容，关闭文件，创建并打开新文件，写文件，关闭文件。主要的变量是文件的内容，通过变化文件的大小看Lua的支持情况。

## 只运行一次的小程序

只运行一次的小程序是Lua编写脚本运行的过程。Lua脚本有自己的编码风格，我们希望通过编写一个简单的脚本，看看其运行情况即可。代码的编写样式也是各式各样的，不可能将Lua的脚本做全部的测试，所以我们只测试满足条件的部分脚本，例如顺序，分支，循环，递归的脚本各一个，测试Lua基本脚本可以执行。

## 测试方案

编写四类脚本，顺序执行，分支执行，循环执行和递归执行的脚本各一个(如果可能的话，可以编写多个)。分别在Lua最新版本的环境中执行，查看并分析执行情况。

## 模式匹配测试

Lua并不使用POSIX规范的正则表达式（也写作regexp）来进行模式匹配。主要的原因出于程序大小方面的考虑：实现一个典型的符合POSIX标准的regexp大概需要4000行代码，这比整个Lua标准库加在一起都大。权衡之下，Lua中的模式匹配的实现只用了500行代码，当然这意味着不可能实现POSIX所规范的所有更能。然而，Lua中的模式匹配功能是很强大的，并且包含了一些使用标准POSIX模式匹配不容易实现的功能。通常使用string.gmatch（str,pattern）函数实现模式匹配。

## 测试方案

通过Lua的string.getMatch（str,pattern）进行字符串的匹配。循环输出匹配结果，查看匹配功能的效果

## Lua 和C++ 混合使用

该方面测试主要是针对lua调用C++和C++调用lua的功能实现情况，根据软件需求说明书在该方面所提的需求，我们设计了如下一个测试功能点：调用lua打印“hello world”。

## 调用lua打印“hello world”

在LUA和C++混合使用方面，我们选择了C++调用Lua打印”hello world”这个功能测试点，因为lua这门语言最开始产生的原因就是想做一门脚本嵌入到C程序当中，提供灵活的扩展和定制功能。Lua由标准C编写而成，几乎在所有操作系统和平台上都可以编译，运行。Lua脚本可以很容易的被C/C++ 代码调用，也可以反过来调用C/C++的函数，这使得Lua在应用程序中可以被广泛应用。综上所述，我们认为这个功能测试点能够完全的测试lua在被C++调用方面的全部功能。所以我们选择该功能测试点。

## 测试方案

首先在DEVC++中编写一个C++程序，在这个程序中引入lua.h，lauxlib.h和lualib.h这三个标准库，然后在C++中利用控制台输入LUA代码，最后编译，解释执行输入的LUA代码，最后通过C++在控制台中输入结果。进行结果对比是否符合预期结果。

# 性能测试以及性能测试点分析

性能测试是通过自动化的测试工具模拟多种正常、峰值以及异常负载条件来对系统的各项性能指标进行测试。[负载测试](http://baike.baidu.com/view/651437.htm)和[压力测试](http://baike.baidu.com/view/565536.htm)都属于性能测试，两者可以结合进行。通过[负载测试](http://baike.baidu.com/view/651437.htm)，确定在各种工作负载下系统的性能，目标是测试当负载逐渐增加时，系统各项性能指标的变化情况。[压力测试](http://baike.baidu.com/view/565536.htm)是通过确定一个系统的瓶颈或者不能接受的性能点，来获得系统能提供的最大服务级别的测试。

从广泛意义上讲性能测试包括：压力测试、稳定性测试、负载测试和可扩展测试等。在不同应用系统的性能测试中，需要根据应用系统的特点和测试目的的不同来选择具体的测试方案。

本研究课题选择的是lua，根据需求说明书中的非功能需求，我们选择了以下性能测试点进行测试。

## Table性能测试

当需要创建非常多的小size表时，应该预先填充好表的大小。

问题产生：当我们把一个新键值赋给表时，若数组和哈希表已经满了，则会触发一个再哈希(rehash)。再哈希的代价是高昂的。首先会在内存中分配一个新的长度的数组，然后将所有记录再全部哈希一遍，将原来的记录转移到新数组中。新哈希表的长度是最接近于所有元素数目的2的乘方。

## 测试方案

在非常多的小标创建的时候，预先填充以避免rehash。

## 字符串性能测试

大字符串连接中，应该避免..。应用table来模拟buffer，然后concat得到最终字符串。

问题产生：所有的字符串变量，只保存字符串引用，而不保存它的buffer。

## 测试方案

问题解决：应用table来模拟buffer，然后concat得到最终字符串。

## WEB应用的性能测试

ngx\_lua是基于nginx的扩展 nginx已经有无数家验证过，是个可靠的WEB应用。而且ngx\_lua 已经有很多广泛的应用。那么与其他脚本语言php, lua, python 和 node.js在性能上有何优势。ngx\_lua模型也是单线程的异步的事件驱动的，工作原理和nodejs相同，代码甚至比nodejs的异步回调更好写一些。通过并发我们对三个脚本语言进行了测试。

## 测试方案

搭建好Red hat linux的server 和client端，分别使用PHP,NODE.JS,LUA环境编写测试脚本，通过100个并发对比时间，CPU，内存的性能优势