Robot Framework模拟器组件介绍

V1.0.0

2015-12-17

|  |  |  |  |
| --- | --- | --- | --- |
| **作者** | **版本** | **时间** | **内容** |
| 王绵杰 | V1.0.0 | 2015-12-17 | 创建 |
|  |  |  |  |
|  |  |  |  |

目录

[1 自动化测试面临的问题 4](#_Toc438218494)

[1.1 原有自动化测试工具面临挑战 4](#_Toc438218495)

[1.2 新的测试工具选型 4](#_Toc438218496)

[1.3 模拟器组件开发的必要性 5](#_Toc438218497)

[1.3.1 测试工作需要各种模拟器 5](#_Toc438218498)

[1.3.2 模拟器开发方式代价大 5](#_Toc438218499)

[2 模拟器组件的功能特色 6](#_Toc438218500)

[3 模拟器组件架构 6](#_Toc438218501)

[3.1 模拟器组件组成 6](#_Toc438218502)

[3.1.1 SimLibrary： 7](#_Toc438218503)

[3.1.2 SimServer 7](#_Toc438218504)

[3.2 整合技术 8](#_Toc438218505)

[3.2.1 Twisted 8](#_Toc438218506)

[3.2.2 Zope 8](#_Toc438218507)

[3.2.3 Lxml 8](#_Toc438218508)

[3.3 功能实现原理 8](#_Toc438218509)

[4 组件使用场景 9](#_Toc438218510)

[5 附录 9](#_Toc438218511)

# 自动化测试面临的问题

## 原有自动化测试工具面临挑战

测试部原来的自动化工具使用ATP。ATP是一个接口测试的自动化平台，接口测试功能强大。但具备以下局限性：

1. ATP不支持WEB、APP测试；
2. 在接口测试方面，ATP对MISC产品相关协议支持良好，其他协议需要二次开发；
3. ATP系统实现较为复杂，目前代码行近40万，二次开发复杂度高，开发周期和投入都无法保证测试的快速支撑；
4. 其他商业自动化测试工具，不仅存在购置成本高的问题，而且支持的测试类型有限，工具灵活性差，脚本开发成本也高。

## 新的测试工具选型

根据工作需要，结合业界开源自动化测试技术的发展，我们选择了开源测试框架Robot Framework。Robot Framework具备以下特点：

1. Robot Framework是一款自动化测试框架，具备良好的可扩展性，可以使用不同的测试库来实现多种类型测试，**支持WEB自动化，接口测试自动化，APP自动化，windows应用程序自动化**；
2. Robot Framework是开源的，有Nokia、ThoughtWorks、google等公司支持，发展前景良好；
3. 支持关键字驱动、数据驱动。 利用已有的关键字，测试人员可以封装自己需要的关键字；自动化案例易于维护，上手快；
4. 测试执行报告和日志是HTML格式，容易阅读；
5. 可以同时测试多种类型的客户端或者接口，可以进行分布式测试执行；
6. 测试人员可以使用Python和java创建自己需要的测试库；
7. Robot Framework有非常丰富的类库，支持Robot Framework的库很多，开源的标准库加扩展库有几十个：

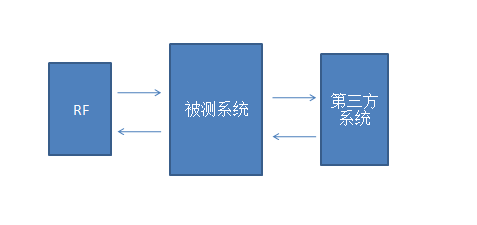
* web自动化测试：SeleniumLibrary，Selenium2Library，Selenium2Library for Java、watir-robot等。
* Windows GUI测试：AutoItLibrary。
* 移动测试：Android library、iOS library、AppiumLibrary等。
* 数据库测试：Database Library (Java)、Database Library (Python)、MongoDB library等。
* 文件对比测试：Diff Library。
* HTTP测试：HTTP library (livetest)、HTTP library (Requests)等。

1. Robot Framework能与持续集成工具Jenkins集成，支持敏捷开发；

## 模拟器组件开发的必要性

### 测试工作需要各种模拟器

* 在项目测试过程中，存在的被测系统与第三方系统交互的情况，需要一个模拟器模拟第三方系统的行为；



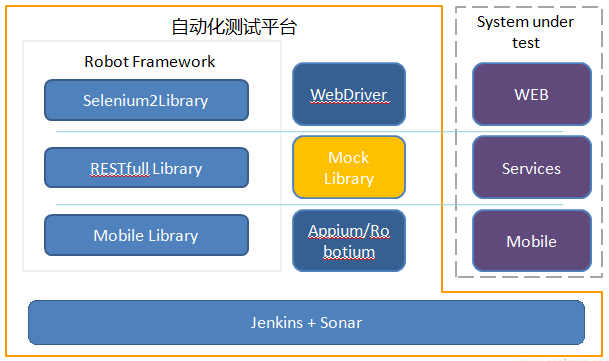
* 多个产品项目都存在这种需求；某些产品和多个外部系统，与每一个第三方系统交互可能存在一种或多种协议接口，每种接口协议可能存在多个接口；

### 原有模拟器开发方式代价大

* 原来的工作方式是针对每种协议都需要开发一个模拟器。针对每种协议，每个接口编写代码实现，代码开发工作量大，且复用度低；
* 有些复杂的模拟器的开发，对人员技能要求较高；
* 整体而言，这些都增加了测试准备的工作量；

针对以上问题，我们规划了一个**通用**的模拟器的**公共组件**，该公共组件具备以下优点：

* 能与Robot Framework集成；
* 模拟器使用时无代码开发工作量，对协议的适配能通过配置文件的方式实现，如果遇到新的协议接口的测试，无需进行代码开发；
* 有利于形成统一风格的工作方式，有利于内部人员互备和流动；



# 模拟器组件的功能特色

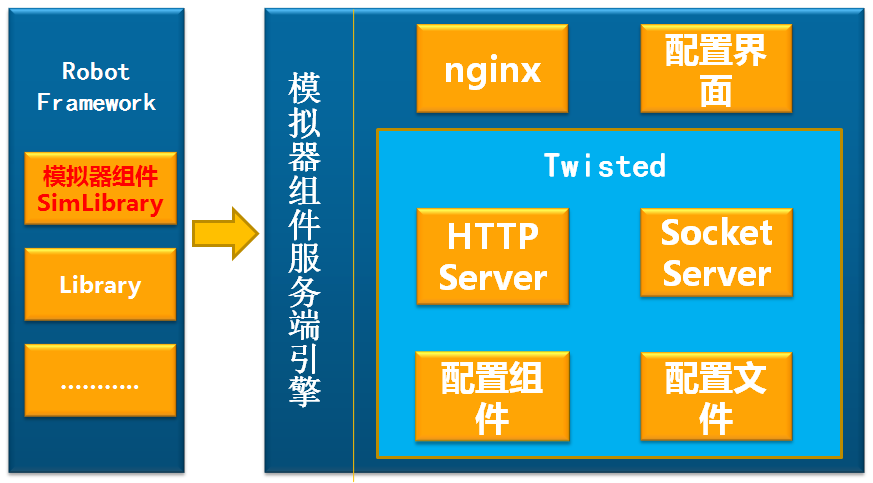
服务器端模拟器库是一个通用的，模拟第三方系统的公共组件，用来配合Robot Framework实现接口自动化测试。功能特点如下：

* 支持HTTP协议，HTTP GET，HTTP POST。如基于HTTP的xml协议，如soap，　基于http的json协议，如RESTFull API；
* 支持长连接协议，如常见的CMPP2.0/3.0协议；
* 与Robot Framework的集成，以关键字封装的方式使用；
* 提供丰富的日志信息；
* 能够通过配置决定需要模拟的服务器的预期行为，按需返回应答；
* 对被测应用的请求包进行强校验的能力；

# 模拟器组件架构

## 模拟器组件组成

模拟器组件由两个部分组成：模拟器的Robot Framework调用库SimLibrary，服务器端引擎SimServer。



### SimLibrary：

SimLibrary作为Robot Framework的库提供，与RIDE配合使用，通过关键字封装使用。SimLibrary与SimServer交互，告诉SimServer，对被测系统的消息接口校验规则，返回应答消息的规则。

### SimServer

SimServer在基于事件驱动的网络引擎Twisted基础上实现，根据配置启动了一个或多个HTTP Server或TCP Server，这些Server能够模拟第三方系统，接收被测系统的请求，根据SimLibrary设定的校验规则和应答规则，对被测系统请求进行校验，校验通过再根据返回规则发送应答包。

对被测系统的消息接口校验规则支持：IP地址校验，超时控制，消息内容校验；消息内容校验支持：

1. 完全字段匹配：字段值等于期望值；
2. 空字段：期望消息中存在该字段，但为空值；
3. 字段不存在：期望消息中不存在该字段；
4. 字段非空校验：期望消息中存在该字段，且值不为空；
5. 字段格式校验：对字段格式校验，如时间格式，长度；

返回应答消息的规则控制支持：

1. 不返回；
2. 延迟返回；
3. 返回内容可以根据测试案例需要返回成功，返回不同错误码的失败；

## 整合技术

### Twisted

Twisted是用Python实现的基于事件驱动，开源网络引擎框架，可扩展性高、支持异步通信。

Twisted广泛使用于生产环境中，从Google、卢卡斯电影到Justin.TV以及Launchpad软件协作平台都有在使用

Twisted实现了设计模式中的反应堆（reactor）模式，这种模式在单线程环境中调度多个事件源产生的事件到它们各自的事件处理例程中去。

Twisted的核心就是reactor事件循环。Reactor可以感知网络、文件系统以及定时器事件。它等待然后处理这些事件，从特定于平台的行为中抽象出来，并提供统一的接口，使得在网络协议栈的任何位置对事件做出响应都变得简单。

### Zope

Zope是一个开源的web应用构建的框架。python支持多继承，但是不支持接口，zope.inteface是其三方的接口实现库，在twisted中有大量使用。

### Lxml

Lxml是Python语言里和XML以及HTML工作的功能最丰富和最容易使用的库，特点是功能强大，性能优异。

## 功能实现原理

一个典型的接口自动化过程由以下步骤组成：

1. 数据环境准备：准备被测系统的测试数据，启动一个模拟器，通知模拟器接收何种消息，如何应答；
2. 发起测试：测试工具向被测系统发起请求，或通过测试工具启动被测系统一个Job；
3. 数据检验：检查被测系统应答，数据库，日志或文件；
4. 环境数据清理：清理脏数据，避免对其他测试案例的运行有影响；

模拟器组件在自动化测试中的工作过程为：

1. Robot Framework通过调用SimLibrary库，通知模拟器服务端SimServer对被测系统请求的校验规则，应答规则；
2. Robot Framework向被测系统发起测试；
3. 被测系统向模拟器服务端SimServer发送请求；
4. SimServer根据SimLibrary设定的校验规则，检验请求消息；
5. 校验通过，SimServer根据SimLibrary设定的应答规则，返回应答；

功能原理图

# 组件使用场景

在微服务盛行的年代，原有的大型ＩＴ服务已经在向微服务模式进行转变，原来单一的application应用也被众多微小的组件架构代替，以提供更强的系统可伸缩能力和提供更迅速的产品变更能力。组件的分层是架构方面的工作，但分离为多个可被单独部署的组件，各组件都依赖或者向外部组件提供各式接口，单一组件提供的接口不多，但一套系统所有组件加起来的内外部接口，数量较多，其测试成本居高不下。

本组件与Robot Framework结合，适用于接口类型的测试，能极大地减轻人员编写和执行接口类测试案例的压力，并能大量减少未来的维护成本，是微服务系统快速上线，以及应对未来的快速变更的测试公用组件。该组件能明显减少项目测试人员对模拟器写作的能力要求。适合快速推广。

# 附录

《通用模拟器安装部署手册\_v1.0.0.0》