

基于云服务的自动化应用巡检研究与应用

设计了一套基于云服务的自动化应用巡检工具，对巡检脚本进行重新设计，并在云服务平台进行部署，采用任务动态分配和并行操作的方式分配任务脚本，保证了巡检任务的连续、稳定且高效执行，提高了巡检工作的可靠性。

马思硕¹ 汪明² 吕进² 赵宇亮^{2/1}. 国网北京市电力公司信息通信分公司 2. 国家电网公司信息通信分公司

0 引言

随着国家电网公司信息化建设进程的不断加快，国家电网公司的信息系统部署模式逐步由两级部署转变为集中部署或一级部署，而运维模式也逐渐向一级运维模式转变，信通公司承接的一级运维系统数量也越来越多，这对信息系统运维工作的精益化水平提出了更高的要求。目前在信息系统进入试运行环节后，需要进行大量的测试以保证软件质量，而针对系统应用情况，更需要的是保证其运行状态正常，于是，只需要采取测试工作的一个子集——巡检，即较好地解决问题。

然而目前大部分测试软件注重的是对软件的整体测试，并没有达到系统巡检所需的定时、稳定及可扩展性强等特点。因此国网信通公司目前的主要巡检方式仍然为人工巡检，但由于系统规模大、应用功能多，巡检工作占用大量的人力资源和时间资源。每次巡检都是大规模的重复性劳动，误检、漏检现象也时有发生。

针对这种情况，我们决定采取自动化的方式对系统进行定期巡检。然而现有的自动化工具大多以单台或多台服务

器形式运行，具有如下的缺点：

1) 自动巡检过程中，不能对机器进行其他操作。每个系统都需要准备一台机器或服务器，进行定时自动化巡检，在自动化巡检中如果有人为的干预，会降低机器性能，并会导致巡检中断、失败等现象。

2) 巡检客户端只部署在一台服务器。如果已设置定时巡检的机器出现故障，会导致定时任务无法启动。如果在巡检中机器出现故障，则巡检会失败。

3) 巡检只能按录制脚本顺序执行。录制的脚本是最小的执行单位，且只能顺序执行，不能打断或选取部分进行执行。

针对以上问题，我们决定对巡检脚本进行重新改造设计，并以云计算的架构部署巡检工具，采用任务动态分配和并行操作的方式分配脚本任务，在大幅度提升效率的同时，确保了巡检准确率达到100%。本工具在集成过程中使用了Python语言，采用了开源的Selenium自动测试工具，利用开源的Robot Framework自动化框架进行架构。

1 自动化应用巡检工具架构

根据实际的应用场景，基于云服务的自动化应用巡检工具分为如下任务模块：

1.1 任务提交模块

任务提交模块主要处理用户端相关操作，用户对需要巡检的内容进行录制，生成巡检脚本套件，套件内会包含可独立执行的巡检脚本用例。选择需要的脚本用例，自定义新的脚本套件后，将脚本套件提交给云服务。



马思硕/工程师

关键词/Keywords

自动化应用巡检 ·
云服务 ·
动态分配 ·
并行化 ·

1.2 任务管理模块

该模块主要包含任务的接收、任务的动态分配、任务结果的汇总以及任务报告的发送及备份四个主要功能。

1) 任务接收。获得提交上来的脚本套件，并将脚本套件按脚本用例的为单元进行独立存储。

2) 任务的动态分配。按照上述的动态分配机制，维护处理模块的编号列表和状态列表，并由任务管理模块选取合适的任务处理模块，将脚本用例以并行执行的方式发送到任务处理模块，等待其返回执行结果。

3) 任务结果汇总。一个脚本套件的所有用例都执行完成后，会返回到任务管理模块，由其进行结果汇总，并重新组装成一个完整的脚本套件任务报告。

4) 任务报告的发送及备份。将任务报告返回给用户，并进行备份存储。

1.3 任务处理模块

接收任务管理模块分配的任务并执行，完成后返回执行结果。同时按照预先设定的时间，与任务管理模块进行通信，保证其编号列表和状态列表的及时更新。

3个模块构成本自动化应用巡检工具的总体架构，如图1所示。



图1 自动化应用巡检工具结构图

2 单元化的巡检脚本设计

目前的巡检工具大多使用基于时间顺序的线性巡检脚本，以录制或者手动编写的方式，按照时间节点，形成一套完整的巡检脚本，然而这种脚本形成方式，固化了脚本形式，缺乏灵活性，不利于大规模的分布式巡检。

因此，我们设计了一种单元化的巡检脚本，提出了脚本套件和脚本用例两个概念。我们通过分析网页元素对脚本进行定位，当录制巡检脚本结束后，会自动生成单元化的巡检脚本用例，如定位巡检点、点击和翻页等，这些脚本用例集合

就构成一个脚本套件。用户可以选择相应的脚本用例，构筑自己独特的脚本套件。

比如，如果一个脚本套件为对网页中的20个标签都进行单次点击，查看其功能。则该套件中的一个用例就是对其中的一个标签进行单次点击。则该脚本套件是由20个脚本用例构成。而用户可以根据需要选取其中10个用例，组成新的套件。

使用单元化的脚本用例，增加了脚本设计的灵活性，在云服务平台下，利于其分布式的并行运行。

3 基于云服务的任务动态分配机制

我们根据实际应用场景搭建了云计算平台，由多台服务器集群构成。为了保证任务的动态分配，将其中两台机器选为Master服务器，其他为Slave服务器。任务分配采用Master-Slave的任务调配机制。

两台Master服务器采用负载均衡，以热备方式共同维护两个列表，一个是Slave服务器的编号列表，一个是Slave服务器的状态列表。当用户提交巡检任务时，任务将提交到Master服务器，由Master服务器按照数据库中维护的Slave列表进行任务分配。

1) 编号列表。当一台Slave服务器启动后，会主动与Master进行通信（发出信号，等待返回值1或0，1代表成功，0代表失败），通信成功后，Master服务器就会对该slave服务器进行注册并编号，编号按照0、1、2...n的方式进行，Master可以按照预先设定的时间（如5 s）对该编号列表中的所有Slave服务器进行通信遍历，如通信失败，则代表Slave服务终端，将该slave服务器移除列表。

2) 状态列表。针对各Slave服务器的状态，Master服务器会维护一个状态列表，Master服务器与slave服务器可以按照预先设定的时间（如5 s）进行心跳通信，Slave服务器会主动将自己的状态信息返回给Master服务器的状态列表。我们设计了三类状态：空闲返回0，忙碌返回1，如果预定时间没有返回，则Master服务器将该Slave服务器状态记录为通信失败2。

综上，Master服务器会根据以上两个列表，生成任务分配的算法，将任务分配给在编号列表中的空闲Slave服务器中。

4 并行化的脚本用例执行方法

通过将巡检任务单元化为巡检脚本用例，巡检任务中的所有巡检脚本用例可以根据Slave服务器的数量进行并行处理，Master服务器对整个并行处理过程进行管理，提高巡检效率。如：一个巡检任务中有20个巡检脚本用例，而云服务架构中有10台Slave服务器，则每个Slave服务器只需运行2个巡检脚本用例，理论上巡检时间大体上可以缩短 $20/2=10$ 倍，而且巡检脚本用例越多，Slave服务器越多，巡检效率提高越明显。

Slave服务器只需对分配到的脚本用例进行操作执行，将结果反馈回Master服务器，Master服务器对所有结果进行组装，形成报告，最终反馈给用户，并进行备份存档。

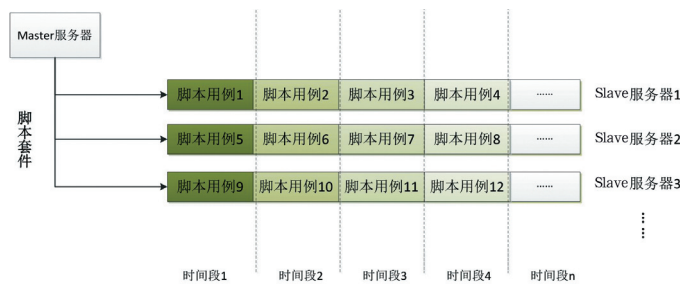


图2 脚本用例的并行执行过程

5 结束语

该自动化应用巡检工具在对现有开源的自动化测试工具进行集成的基础上，设计了云服务的动态分配机制，并定义了脚本套件和单元化的脚本用例，通过对单元化用例的动态任务调用，实现了高效和稳定的脚本执行方式。主要优点体现在以下几方面：

1) 设计脚本套件和脚本用例的概念及实现方法，使得应用巡检更加单元化和标准化，而单元化的巡检任务会更加灵活。我们设计的实现方法可以将一个录制的脚本自动分解为多个单元化的脚本用例，简化了脚本的生成方式，增加了灵活性。巡检人员可以利用单元化的脚本用例生成任意的脚本套件，减轻了巡检人员的工作量，提高

了脚本的准确性。

2) 通过基于云服务的任务动态分配方法，将硬件架构可以无限扩充，从而可以实现高速执行一个大型巡检任务的目的。

3) 通过Master-Slave的动态分配方法与并行化的脚本用例执行方式，使得本方案具备了以下的优点：①实现了高速的应用巡检，由于采用并行的脚本用例方式，如果Slave服务器足够多，就可以高效地对其进行巡检。②实现了高可用性的应用巡检，通过在Master和Slave服务器中使用心跳通信机制，使得Slave服务器的状态可以及时获得，保证了脚本用例可以分配到合适的Slave服务器中，排除了硬件对应用巡检任务的影响。

本文设计的信息系统自动化应用巡检工具通过重新设计巡检脚本并采用云服务的平台架构，实现了任务的动态分配和并行操作，保证了巡检任务的连续、稳定且高效执行，极大地提高了巡检工作的可靠性。

参考文献

[1] Karl E Wiegers. 《Software Requirements》[M]. 北京：机械工业出版社.

[2] Stephen R Schach. 《Object-Oriented and Classical Software Engineering》(Fifth Edition) [M]. 北京：机械工业出版社.

[3] Craig Larman. 《Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design and Iterative Development》(Fifth Edition) [M]. Addison Wesley Professional.

[4] “Software Engineering” (6th Edition), Ian Sommerville, 2000 Addison-Wesley, ISBN 020139815X 英文版，北京：机械工业出版社，2003.

作者简介

马思硕（1986-），男，硕士研究生，工程师，从事电力信息通信、信息系统运维及信息安全工作。

(收稿日期：2015-12-10) **EA**
(编辑：崔文静)