TELKOMNIKA Indonesian Journal of Electrical Engineering
Vol.12, No.7, July 2014, pp. 5678 ~ 5684
DOI: 10.11591/telkomnika.v12i7.5276

Framework of Software Testing Based on Cloud Computing

Bensheng Yang[1], Xiangmeng Yuan*[2], Xiaoguang Huang[3]
[1]College of Recourses, Hebei University of Engineering,
Hebei Handan 056038, China
[2,3]College of Information and Electrical Engineering, Hebei University of Engineering,
Hebei Handan 056038, China
*Corresponding author, e-mail address: yuan.xiangmeng@163.com

Abstract

For the problem that efficiency is low and cost high exists in the traditional software testing method, the paper tested software using cloud testing technology. It introduced related technologies including cloud testing, and described the design of overall architecture of the system in details, designed and implemented the scheduling module using a high priority first scheduling based on dynamic priority. The results of the Matlab simulation experiment show that this scheduling algorithm can reduce test cost and realize the automation of software testing under the condition of significantly improving test efficiency and resource utilization.

Keywords: *cloud computing, TaaS, Hadoop, cloud testing, coal mine web system*

1. Introduction

Software testing is the process of executing a program with the intent of finding errors. Software testing as an important means to guarantee the quality of the software gets more and more people's attention in the field of software engineering. The traditional approach of manually creating in-house testing environments that fully mirror these complexities and multiplicities consumes huge capital and resources, it is seriously restricted the development of testing technology [1]. Cloud computing is an emerging paradigm which opens a new door for software testing. Cloud testing are allocated dynamically to create a highly flexible and scalable computing environment through using virtualization technology. James Whittaker prospected the future of software testing in the book *Exploratory Software Testing*, which referred to the software testing as a services based on cloud computing [2]. Distinguished by the way they are utilized for testing, four different types role can be played by cloud in the testing [3]. They are cloud as system under test, cloud as testware utility, cloud as test environment, and cloud as test logistics. In concrete contexts, when the cloud plays on several roles simultaneously these types may overlap. Cloud testing can be classified into three following types accounting to the roles cloud played in the testing process. (1) Test for the cloud, it involved the testing problems about internal structure, resource configuration and function extension and so on in cloud computing [4]. (2) The migration of the testing, the traditional test methods, management, process and framework are migrated to the cloud [5]. (3) Test other software system using cloud computing. The third kind is mainly introduced in this paper. Test based on cloud platform as a service provided to customers by Cloud testing service providers across the Internet can well solve the problems existing in the traditional test method. In recent years, cloud testing began attracting the attention of academia, and many research results are obtained. D-Cloud is a large-scale software cloud computing testing environment model with fault injection for dependable distributed system [6]. Cloud9 is a parallel symbolic execution of computer clusters on public cloud infrastructures such as Amazon EC2 [7]. YETI is an automated random cloud –based testing tool for Java with the ability to test programs written in different programming languages [8].

The rest of this paper is structured as follows. Section 2 introduces basic concepts about cloud testing and its scope, features, benefits and the function modules included in a nature cloud testing system. Section 3 presents cloud testing solution aim at software testing

Received December 2, 2013; Revised March 10, 2014; Accepted April 2, 2014

and designs the overall structure of the system and system architecture. Section 4 analyses and implements the scheduling module of system. Section 5 Verify the performance of scheduling subsystem through the experiment, shows the conclusion remarks and scratches the future work.

2. Cloud Testing Overview

With the expansion of the scale of software and the complexity of hardware, software testing encountered unprecedented challenges. The characteristics of cloud computing includes the following points, allocation of resources is dynamic, demand for services is customized, the services can be quantified, resources are pooled and transparent, which can solve the problems. Cloud testing is based on the cloud and the cloud infrastructure, using cloud technology and solutions for software testing [9].

The entire testing environments can be configured from the cloud on-demand at a cost that is practical and reasonable due to the pay-to-use nature of cloud computing and with a lead-time that is near impossible within a company's own data center [10]. Cloud testing service providers provide 24/7/365 on-demand automated testing services, it realizes the resource sharing; More importantly, the time to market of the product is shorten on the premise of guarantee the quality of the Web system in cloud testing environment [11, 12]

Not all applications are suitable for testing in the cloud. For some, the cost of migration may outweigh the amortized benefits. Characteristics of an application that can make it feasible for its testing process to migrate to the cloud include: (1) test tasks are independent from one another or whose dependencies are easily identified, this is because concurrent execution is only possible in this situation, And speedup will be came true through concurrent execution; (2) a self-contained and easily identifiable operational environment, in order to know software and hardware environment needed in the testing process; and (3) a programmatically accessible interface suitable for automated testing [13].

A mature cloud testing system should include the following eight types of function modules: scalable test environment service, multi-tenant test modeling and adequacy service, digital test management service, on-demand automated test and control service, test solution integration and composition service, test tracking and monitor service, large-scale test simulation service, testing contracting and billing service [14].

3. System Overall Design

3.1. Cloud Testing Steps

In the process of testing other software in the cloud, first of all, the test requests are made by user and sent through the Internet to cloud testing system, and then they are accepted by system, in the next place, test tasks are scheduled and dispatched, middleware services are provided to tasks, virtual resources are matched and test tasks are executed and supervised. In the end, test results and analytics are collecting and delivering to user using web interface [15]. Cloud testing steps is shown in Figure 1.

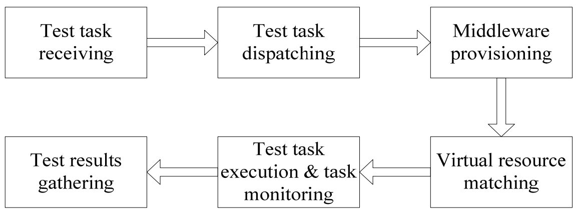

Figure 1. Cloud Testing Steps

3.2. System Architecture Design

Three service models are defined in cloud computing, Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS). Cloud testing can be regarded as SaaS service model using in the field of software testing. We design the cloud software testing system (CSTS) architecture in the light of that three service models with combining with the step of cloud testing. Cloud testing system architecture is shown in Figure 2.

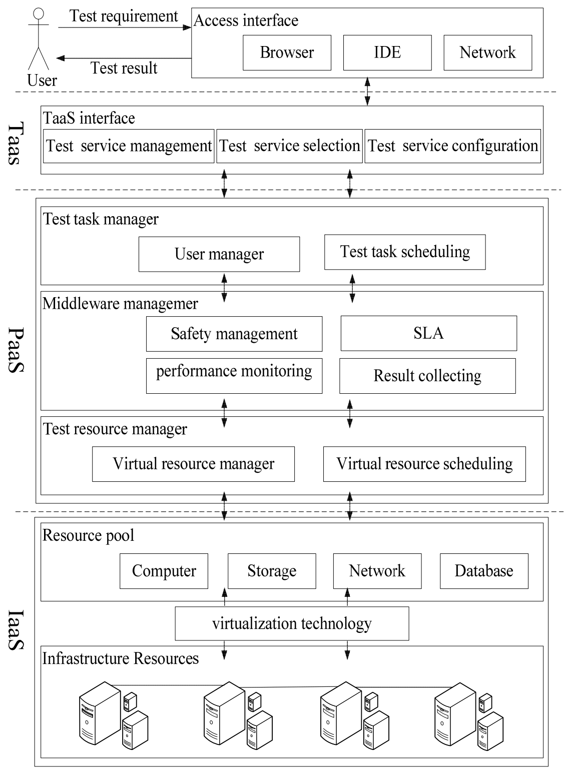

Figure 2. System Architecture Diagram

(1) The IaaS layer

Processor, virtual storage, network and other infrastructure resources are processed to logical resource pool by using virtualization technology to provide to user in the form of services, which is in a unified, centralized pattern. Users make the requests to the CSCT for all kinds of resources according to their own requirement, and do not need to care about how the resources are allocated and a scheduled. In this way, the utilization of hardware and software resources are improved and testing process becomes more intelligent and automated.

(2) The PaaS layer

PaaS is a business infrastructure platform of software development with the purpose of providing customers with unified and customized development of middleware platform, and includes the management of infrastructure resources and test tasks submitted by user at the same time. This layer is comprised of test task management module, middleware management module and test resource management module. It implements the scheduling and allocation of virtualization infrastructure in the virtual resource management module. Middleware management module achieves the following functions, safety management, SLA (Service Level Agreement) performance monitoring and result collecting and analyzing. Test task management module is divided into user management and test tasks scheduling management.

In the virtual resource management module, all kinds of virtual resources may be in two states: idle state and running state. All kinds of resources are stored into two lists according to their states. The virtual resources conform to the requirements are allocated to the test tasks completing the scheduling process based on resources state.

(3)The TaaS layer

According to the test requirement submitted through access interface, free matching software and hardware are chosen and installed to build the target test environment, which is maintained and updated by CSTS. Users enjoy the right to use the software, and can also continuously upgrade.

In addition, user interact with CSTC through the web browser, thin clients explore clients or programming and other ways, submit the test request, consume the test service provided by CSCT, and finally get the test results.

4. Scheduling Model Design and Implement

During the course of resources management in cloud testing, parallel test task scheduling and dispatching algorithms is complicated because of the dynamic nature of the infrastructure used in cloud environment, which do not occur in other testing methods [16]. Various scheduling strategy will lead to complex technical, security issues and determine the execution sequence of test tasks and which virtual machine will be dispatched to the task sorted. The scheduling module was designed and implemented of CSTS in this paper.

4.1. Scheduling Module Structure

After user submits test tasks, it is an indispensable part to predict the number of virtual machine needed according to the number of tasks and the analysis of the test project. In this way, the waste in requesting excessive virtual machines and applying to the cloud resource manager because virtual machine is not enough in the process of test execution can be reduced and avoid. Scheduling subsystem module structure is shown in Figure 3.

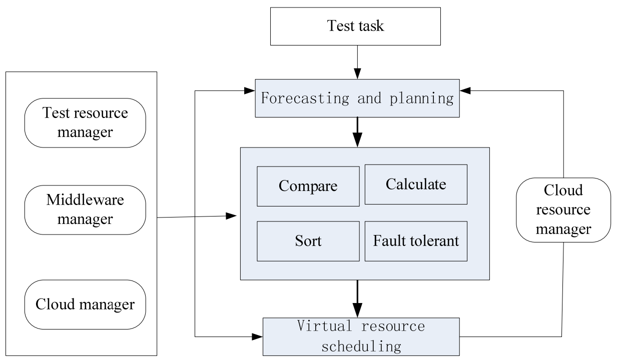

Figure 3. Scheduling Module Structure

4.2. The Algorithm Process

Test task suited to cloud testing are independent from one another, so we take no account of the dependency relationship between tasks during the course of test task scheduling in the cloud testing platform, this is a simplification to the scheduling algorithm.

The scheduling algorithm we used in this paper is that the task with high priority is first executed, which is based on dynamic priority, we called it HDPF algorithm, and takes the task waiting time, task execution time and task weight into consideration. That is to say the priority of tasks will be changed for the task executive condition or the increase of waiting time, and the high priority task is first executed.

(1) The number of tasks and the number of virtual machine are assumed to be finite. There is a list of tasks T , $T = \{T_1, T_2, ..., T_n\}$ performed by a set of machines M , $M = \{M_1, M_2, ..., M_m\}$ .Any tasks can be executed in any virtual machine. However, interruption is unwarrantable before the test process is completed. Any task cannot be split into smaller subtasks. Every time after the first scheduling process, the number of waiting execution tasks are assigned to n, and the number of virtual machine are assigned to m.

Given a definition of two queues: WaitList points the test task waiting queue and FreeVMList denotes free virtual machine queue.

(2) Test tasks are submitted by user through Internet interface, and the weight value is assigned to task by user according to the prompt at the same time. Then the tasks are added into task waiting queue WaitList.

The value of a task is decided by users when they submit test job. According to the waiting time, the execution time and the weight of task, its priority can be calculated by the following formula.

$$p_j = \frac{q_j + d_j}{d_j} + \pi w_j$$

The term j denotes the jth task and i is the ith virtual machine. q_j denotes the waiting time of the jth task and the subscript j refers to a task, d_j denotes the execution time of the jth task, w_j denotes the weight of the jth task and p_j denotes the jth task priority. π denotes a parameter which is imported to adjust the weight between $\frac{q_j + d_j}{d_j}$ and w_j to make them in the same order of magnitude.

(3) The number and configuration of virtual machine is forecasted based on the analysis of project needed to test requirement and the number of test task user submit. CSCT sends message to the cloud resource manager to apply for the virtual machine. The construction of test environment is completed through deploying the operating system, software system and hardware system and so on in light of environment configuration template and this will reduce the request of the new resources in the process of test execution. Of course, there are errors in this prediction on the basis of experience and template.

(4) The number of tasks in WaitList queue in assigned to the n, and the number of tasks in FreeVMList is assigned to the m.

(5) If $n \le m$, CSCT releases the redundant processors, the virtual machines can be assigned to the tasks, and be released immediately after the test tasks performed.

(6) If $n > m$, CTCS calculates priority of all tasks in accordance with the priority formula, sorts them according to their priority, selects the top m tasks and assigns them to m virtual machines. Tasks which were chosen are removed from the queue WaitList and virtual machines which are executing are removed from the queue FreeVMList.

(7) The new tasks are added to the queue WaitList, which are submitted by user and tasks there are any fault occurred in their execution process. Free virtual machines are added to the queue FreeVMList when they turn up. Execute step 4 until all tasks are carried out properly.

4.3. Algorithm Flow Chart

The advantages of the HDPF algorithm: (1) If $n > m$, there is no need to consider the priority of each task, the total execution time is the running time of the task with the longest running time, in other words, it needs $O(1)$ time; If $n > m$, the sorting takes $O(n \log n)$ time and the loop needs $O(n \log \mathrm{m})$, as a consequence, the total of the amount of time is $O(n \log n + n \log m) = O(n \log n)$. (2) Virtual machines are released as soon as each task are completed resulting in the resource utilization of cloud platform are improved. The tasks with long waiting time, short execution time and heavy weight take precedence on execution. Of course, the priority of all tasks must be recounted before every scheduling process, and this will increase the cost of the system. The algorithm flow chart is shown in Figure 4.

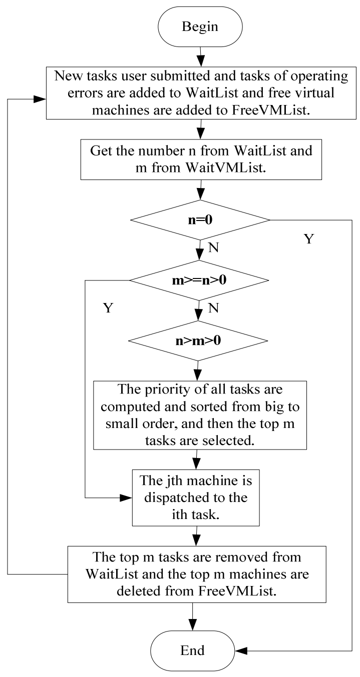
Begin
New tasks user submitted and tasks of operating errors are added to WaitList and free virtual machines are added to FreeVMList.
Get the number n from WaitList and m from WaitVMList.
n=0
Y
N
m>=n>0
Y
N
n>m>0
The priority of all tasks are computed and sorted from big to small order, and then the top m tasks are selected.
The jth machine is dispatched to the ith task.
The top m tasks are removed from WaitList and the top m machines are deleted from FreeVMList.
End

Figure 4. Algorithm Flow Chart

5. Experimental Verification

The algorithm HDPF proposed in this paper and FCFS algorithm are coded in Matlab and analyzed operation efficiency from the perspective of mathematical through the Simulation Experiment. Assuming that the number of virtual machines is 250, the number of test tasks n is valued respectively {50, 150, 250, 350, 450}, the execution time of each test task is not the same, the time of the task with the longest execution time is 12ms, As can be seen from the Figure 5: (1) If the number of the virtual machines is greater than the number of tasks, the total turnaround time of the HDPF scheduling algorithm are constant, and FCFS algorithm is increased when running the same amount of functional test and performance testing. (2) If the number of virtual machines less than the number of tasks, the total turnaround time of HDPF scheduling algorithm increases gently and FCFS algorithm quickly. (3) With the increase of amount of testing tasks, the executive efficiency of the HDPF algorithm is better than FCFS of almost 20%, having a distinct advantage.

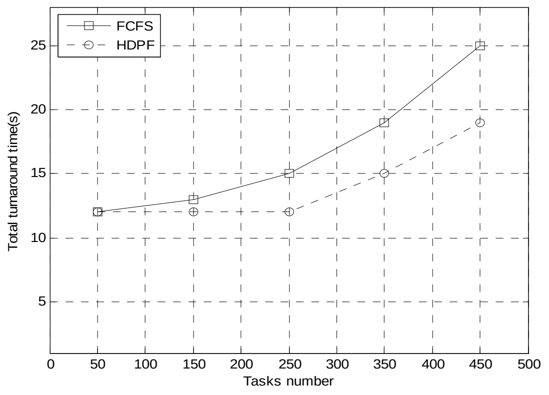

Figure 5. Total Turnover Time Compared

Conclusion remarks can be given from the analysis results: compared with the traditional test methods, there is no need to make an investment in software testing, such as expensive testing tools, the building, maintenance and upgrade of the test environment and so on, when run the same amount of functionality testing tasks and performance testing task. In addition, the test result report can be got soon after the test tasks are submitted. Test efficiency are improved significantly.

6. Conclusion

Combined with cloud computing technology, this paper introduces the structure of the cloud testing system based on cloud computing model framework in detail, and implement the scheduling modules of the system by using a high priority tasks fist scheduling based on the dynamic priority. Compared with the classic FCFS scheduling algorithm, HDPF algorithm can significantly improve the test efficiency of the software system, and reduce test cost through the test experiments in Matlab. In CSTS, the implementation of other modules and the safety of the system is the focus of the next step work, further research should be deal with.

Acknowledgements

This paper supported by the construct program of the key discipline in Hebei province and the natural science foundation of Hebei provice(Grant No. E2011402046)

References

[1] Vinaya Kumar Mylavarapu. Taking testing to the cloud. Cognizant. http://www.cognizant.com/insights/perspectives/taking-testing-to- the-cloud.
[2] James A, Whittaker. Exploratory Software Testing. Addison-Wesley Professional. 2009.
[3] Victou Czenter, Performance testing meets the cloud opportunities and challenges. SQS, http://www.sqs.com/ en-group/_download/White_Paper_Performance_Testing_Cloud_EN.pdf
[4] Chan WK, Lijun Mei, Zhenyu Zhang. Modeling and testing of cloud applications. In Services Computing Conference. *IEEE Asia-Pacific*; 2009; 111-118.
[5] Tauhida Parveen, Scott Tilley. *When to Migrate Software Testing to the Cloud.* In the Third International Conference on Software Testing, Verification, and Validation Workshops (ICSTW). 2010; 424-427.
[6] Takayuki Banzai, Hitoshi Koizumi, Ryo Kanbayashi. *D- Cloud: Design of a Software Testing Environment for Reliable Distributed Systems Using Cloud Computing Technology.* Proceedings of IEEE/ACM International Conference on Cluster, Cloud and Grid Computing, 2010, pp. 631-636.
[7] Liviu Ciortea, Cristian Zamfir, Stefan Bucur. Cloud9: A Software Testing Service. *ACM SIGOPS Operating Systems Review.* 2010; 43(4): 5-10.
[8] Manuel Oriol, Faheem Ullah. YETI on the cloud. Proceedings of the International Workshop on Software Testing in the Cloud. 2010: 434-437.
[9] Wikipedia. Cloud testing. http://en.wikipediaorg/wiki/Cloud _testing. 2013; 8.
[10] A Vanitha Katherine, K Alagarsamy. Software testing in cloud platform: a survey*. International Journal of Computer Applications*. 2012; 46(6): 21-24.
[11] Priyanka, Inderveer Chana, Ajay Rana. Empirical Evaluation of Cloud-based Testing Techniques: A Systematic Review. *ACM SIGSOFT Software Engineering Notes.* 2012; 37(3): 1-4.
[12] Jerry Gao, Xiaoying Bai, Wer-Tek Tsai. Cloud Testing- Issues, Challenges, Needs and Practice. *Software Engineering: An International Journal (SEIJ).* 2011; 1(1): 9-23.
[13] Scott Tilley, Tauhida Parveen. Software Testing in the Cloud: Migration and Execution. Springer, 2012.
[14] Jerry Gao, Xiaoying Bai, Wer-Tek Tsai. *Testing as a Service (TaaS) on Clouds.* Proceedings of Service- Oriented System Engineering. 2013: 212-223.
[15] LVD Aalst, "Software testing as a service (STaaS). http://www.tmap.net/Images/Paper%20STaaS_ tcm8-47910.pdf.
[16] Michael L Pinedo. Scheduling: Theory, Algorithms and Systems. Third Edition. *Springer.* 2008: 118-130.