RESEARCH ON DISTRIBUTED SOFTWARE TESTING PLATFORM BASED ON CLOUD RESOURCE

Shi Hengliang[1], Zhao Changwei[1], Huang Tao[1] and Dong Yongsheng[1]

[1]Computer School of Henan University of Science & Technology
shi_hl@sina.com.cn

ABSTRACT

In order to solve the low efficiency problem of large-scale distributed software testing , CBDSTP(Cloud-Based Distributed Software Testing Platform) is put forward.This platform can provide continous integration and automation of testing for large software systems, which can make full use of resources on the cloud clients, achieving testing result s in the real environment and reasonable allocating testing jobs, to resolve the Web application software configuration test, compatibility test and distributed test problems, to reduce costs, improve efficiency. Through making MySQL testing on this prototype system, the verification is made for platform architecture and job allocation effectiveness.

KEYWORDS

software testing platform, distributed job scheduling,ACO

1. INTRODUCTION

Compared with traditional software products, the degree of current software products' integration,modulization and complication is becoming higher and higher, which makes the testing of large-scale distributed software as a research hotspot. There are two reasons: the first is the development of Internet makes Web application system's complexity and scale increasing. The traditional centralized software testing methods have certain limitations, and the virtual testing environment and data is difficult to be realistic. The second is that global software development has become a trend, making software development is from centralized to the distributed method to realize software continuous integration and test anywhere and anytime.

Aiming at the problems of software testing, domestic and foreign scholars have put forward a series of test tools and frameworks. In accordance with the realization way,software testing platform can be mainly divided into centralized testing and distributed test. Based on the basis of local testing, centralized software testing framework usually is only for certain type or structure of software testing, which is characterized as higher testing cost and lower efficiency. Li[1] proposed a testing framework for model transformation, Stocks and Carrington[2] put forward a testing framework based on specification introduction, Hartman proposed a model based testing tools AGEDIS[3], Marinov proposed a new automation testing in Java program test framework, realizing automatic generation of test cases and operation [4].

The distributed software testing framework, will test the different jobs assigned to the different test termina. After the test is completed, the testing framework will collect intermediate test results, and generate the final test report. Porter[5] proposed a distributed software testing

framework based on Skoll, making full use of Internet free client resources and establishment of software automation test platform. In the industry, there are many companies who have developed its test framework, such as Cloud Testing, Kite and SOASTA[6], but these frames can only be used for the Web application performance testing and load testing. Zhu[7] proposed a distributed environment for real-time software automated regression testing framework, this framework can complete automatic regression testing according to the specific needs of dynamic configuration system information.

In view of the existing software testing tools and frameworks in solving large distributed software testing exists insufficient, the author put forward cloud resources-based distributed software testing platform of CBDSTP(Cloud-Based Distributed Software Testing Platform), which provides a large software system continueous integration and test automation, making full use of cloud resources on the client resources, and realized test job allocation according to resources reasonably, achieved software testing objectives in the real environment , resolved the Web application software configuration test, compatibility test and distributed test question, which reduces the test cost, and improves the test efficiency.

2. CONSTRUCTION OF CBDSTP

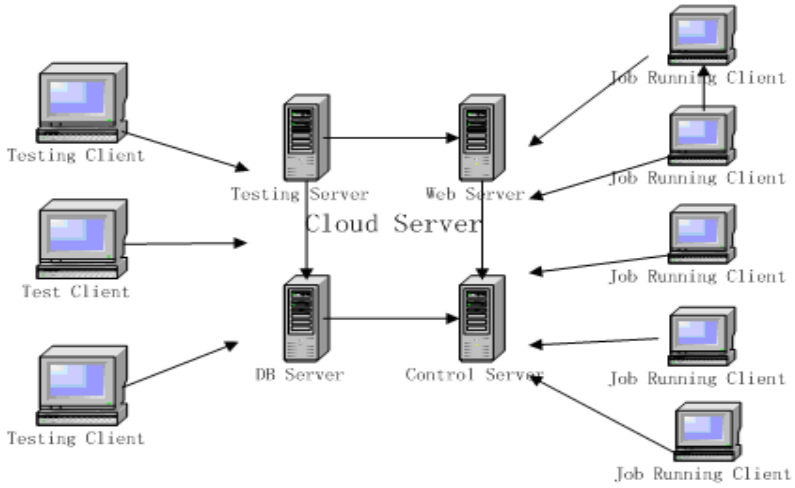

Figure1.CBDSTP Deployment Graph

CBDSTP (Cloud-Based Distributed Software Testing Platform) is based on cloud computing resources (computing resources, storage resources, cyber source), which testing personnel submit test job through the Web browser, and then platform receives the test job, generates test case, and then assigns the testing job to different test job running clients, finally collects different test records and intermediate results for test result analysis and test report generation. Job running client is a cloud platform terminal volunteers, they can apply to join and leave testing platform according to its own terminal performance, and obtain certain economic benefits on the basis of their working hours. During the test job execution, the server will manage and monitor all clients running testing job dynamically. As shown in Figure 1.

The system structure of CBDSTP is shown as Figure2. The platform consists of testing client, job running client and servers. The structure between server and testing client takes B/S, and the structure between server and job runniing client takes C/S architecture. Test staff client provide the test tasks, test process and test results view view interactive pages. Job running client is responsible for excetuing testing jobs and uploading test ing intermediate result. The server is the core of the test platform, consisting by Web server, test server and database server component, responsible for the reception and management of test job, generation of test subtasks and scheduling of jobs, running the client dynamic monitoring, collection and analysis of test results, adjustment of testiing plan.

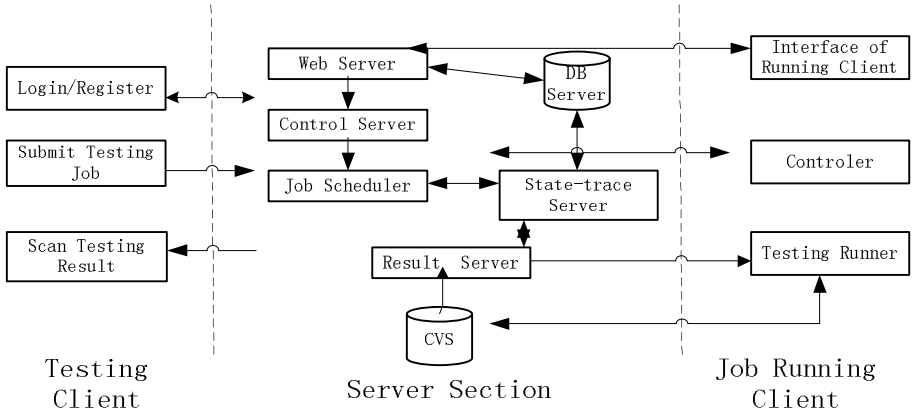

Figure 2. CBDSTP System Architecture Graph

Web server in servers section provides testing personnel 4 user serive interfaces such as registration/login, submitting testing tasks, querying testing results and registration of job running client. Database server is for storage static and dynamic data of system which is the resource sharing platform. Testing server consists of test task partition, test task scheduler, test results manager and a test task state tracker. Test task scheduler can implement task allocation fully based on dynamic parameters of running client resource. Test results management analysis test results uploading from operation end, which will produce test report by analysis of testing results through using software's statistical measure criterion. State tracker is responsible for real-time tracking task running end state to call exception handling module according to state information correspondingly.

2. CBDSTP PROCESSING FLOW

Testing personnel login and upload test task. Server invokes the task partition to generate subtasks. Task running client registrates in and installs client software, sends testing task request according to their own needs. Server task assigns test task for running client. Task running client establishes testing environment, runs test task and upload test results. Server analysis and collects the intermediate testing results to generate the final testing report. Testing personnel views the test results report through browser.

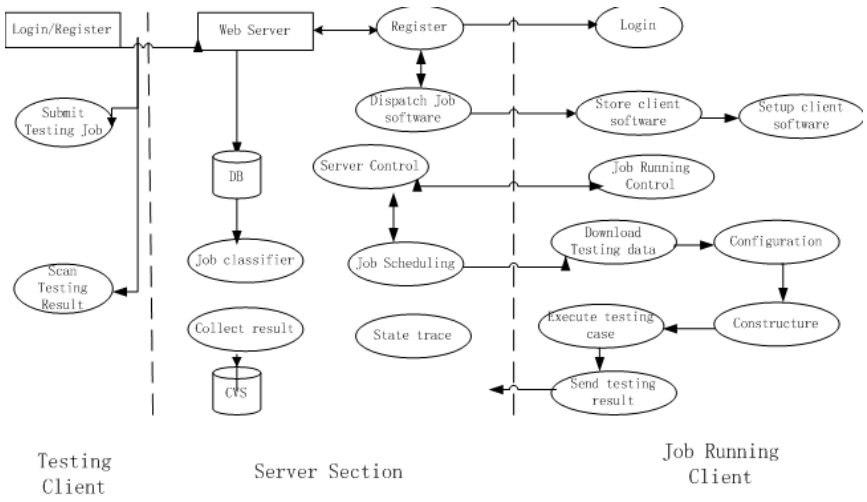

Figure 3. CBDSTP processing flow

3. TESTING TASK PARTITION AND ALLOCATION ON CBDSTP

As a random search optimization algorithm, ACO algorithm and other simulated evolutionary algorithm have defects, whose positive feedback makes the search results are prone to fall into local optimal prematurally, i.e. when the search going into a certain extent, the solution of all individual ants are basically as the same, but has been unable to obtain the optimal solution stagnation from bigger range, that is to say it is unable to find the global optimal solution, it may appear unbalanced phenomenon of load. Node resources overload easily result in cloud computing task scheduling bottlenecks, which effects of task minimum completion time; node resources under-load, the resources can not be fully utilized [8,9]. In swarm intelligence optimization algorithms, there are PSO (Particle Swarm Optimization) algorithm[10], but PSO algorithm is is difficult to understand,comparing with ACO algorithm, meanwhile the algorithm is needed to encode for a large number of particles, so it requires larger work for the resource node and task particles[11,12].

A typical job scheduling problem is described as the following: n jobs need to bedispatched onto n node machines, and one node machine only processes one job, and one job only can be implemented by one node machine[13]. So different dispatch plans have different execution cost and resource consumption, job scheduling is to find one plan to make jobs completing smoothly to ensure availability, reliability and optimality[14]. How to make a allocation to complete all tasks with minimal cost is conversed into how to obtain task assignment matrix Rn*n. [15]

Suppose there are n ants to complete all the submitted jobs. Ant's one trip stands for one job dispatching procedure in which ant needs n walks which demonstrates dispatching one job, tagging the walks as s; when all the ants complete one trip, can be thought as one loop finished; Nc stands for the times of loop. And introduce two sets ask={task1,task2……,taskn} and Node={node1,…,noden};Task stands for job set that is to be dispatched; and Node stands for slave node machine set that is to be given jobs.

The traditional ACO algorithm was used in local pheromone renewal, the releasing pheromones Q everytime is a constant. If the pheromone right of arc (x, y) is greater, the number of ant who passes through the arc are more, so updating times of local pheromone is more and more, which eventually leads to the information amount gap between different

arcs is too big, resulting in the algorithm searches trapped into local optimal. Therefore, along with the algorithm search status changes, Q value should be constantly revised, and not be a constant.

Definition 3.1 Suppose the number of ant passing by node X is m; and the number of ant passing through arc (x, y) is M, then M / m is ant pheromone weight of arc (x, y).

Correction principle: Q(t)=Q*(1-(M / m)),and if m=0, Q(t) =0.

3.1 Global pheromone updating

During the Kth ant in the Q iteration, supposed there are a total of Rk ants passed by node X, and there are rk ants who chose arc (x, y) as their following path. Algorithm for global update rule is as the followging:

$$\tau_{ij}^{new} = (1-\rho) * \tau_{iji}^{old} + \Delta\tau_{ij}; \forall i, j, i \neq j \tag{3-1}$$

$$\Delta\tau_{ij} = \sum_{k=1}^{m} \Delta\tau_{ij}^{k} \tag{3-2}$$

$$\Delta\tau_{ij}^{k} = \begin{cases} Q * (1 - R_k / \gamma_k), ant(k) is from i to j \\ 0, else \end{cases} \tag{3-3}$$

3.2 Local pheromone updating

In order to make the load more evenly, volatile balance factor is added in local pheromone renewal rule. The balance factor is mainly used to describe the pheromone volatilization changes, but is no longer constant.

Ordered:
$$\omega = length\_this / length\_\max, \tag{3-4}$$

length_this represents the scheduling length, length_max for the iterative scheduling length this job scheduling.

$$\tau_{ij}(t+1) = (1-\omega*\rho)*(\tau_{ij}(t)) + \Delta\tau_{ij}(t+1) \tag{3-5}$$

3.3 Pheromone quantity control

In order to avoid algorithm stagnation, the quantity of pheromone on each path is limited in a certain range[$\tau_{\min}$, $\tau_{\max}$], the value beyond of the range will be limited to $\tau_{\min}$ or $\tau_{\max}$ which can be used to avoid the pheromone on a path is much larger than or is far less than the pheromone on any other rest paths, causing all ants concentrate on the same path or away from a path. And the setting of $\tau_{\min}$ or $\tau_{\max}$ is crucial, because each iteration path increases the

maximum information for $1/L(S^{gb})$, which corresponds to a global optimal solution to this path length, so when updating the optimal solution, it needs to update $\tau_{\min}$ or $\tau_{\max}$ simultaneously, and the pheromone $\tau_{\min}$ or $\tau_{\max}$ is inversely propotional to ρ and $L(S^{gb})$, with proportional to the number of excellent ant. The following rules are used to update dynamically.

$$\tau_{\max}(t) = \frac{1}{(2(1-\rho)*L(S^{gb}))} + \frac{\sigma}{L(S^{gb})} \quad (3\text{-}6)$$

$$\tau_{\min}(t) = \frac{\tau_{\max}(t)}{20} \quad (3\text{-}7)$$

Where σ is specified as the number of excellent ants.

4. IMPLEMENTATION AND VERIFICATION OF CBDSTP

CBDSTP is a software testing platform based on Web, using MyEclipse as development environment, with Java as the development of language. System server is running in Linux environment, and testing personnel client and test task running client run on Linux or Windows OS. The system adopts MySQL as background database, for storing system dynamic information, such as the test task information, test task running client information.

In order to verify the feasibility, experiments select MySQL, DTK2.0(EN) and DTP2.0(EN) as measured software, included with a test script as test data. The goal is to validate the feasibility of the platform architecture, so the simulation of some processing part ignores some problems that should be considered in the actual environment, such as confidentiality and security.

project name	machine	expected total(Sec.)
mysql7.0	P1, P2, P3, P4	25600
DTK2.0(EN)	P2, P3, P4, P9, P10	30267
DTP2.0(EN)	P5, P6, P7, P8	45298

Figure4. Tested task assignment graph

Figure4 shows the tested task assignment graph according to the ACO algorithm. We can see that mysql7.0 is dispatched onto P1,P2,P3, and P4 physical machine. Respectively DTK2.0(EN) and DTP2.0(EN) are also dispatched to some machines according to the burden of tested task.

The experimental conditions are as follows: 2 Linux servers connected by SSH seamlessly. 5 PC with Windows and Linux OS of different configurations for running the testing task respectively; 3 ordinary PCs for testing personnel to use.

Testing personnel client: testers upload test tasks via the test task submission interface, including testing task name, testing software code, test data, testing task running environment information, test case list information. The test data contains test script and test cases. Testing task running environment information includes testing task configuration information (such as the operating system type, version), required testing tools and tool installation script. Testing case table contains all cases name in tasks, the estimated execution time.

From Figure5, we can see the week average output or daily average output,and team output level. Through the defects we can even trace back to the source problem.At the same time,we also can grasp the team's key tasks before distribution, and the task complete state in last period. we can not only master the final results, but also the implementation process, even as the qulity and quantity of tested softwares during certain period.

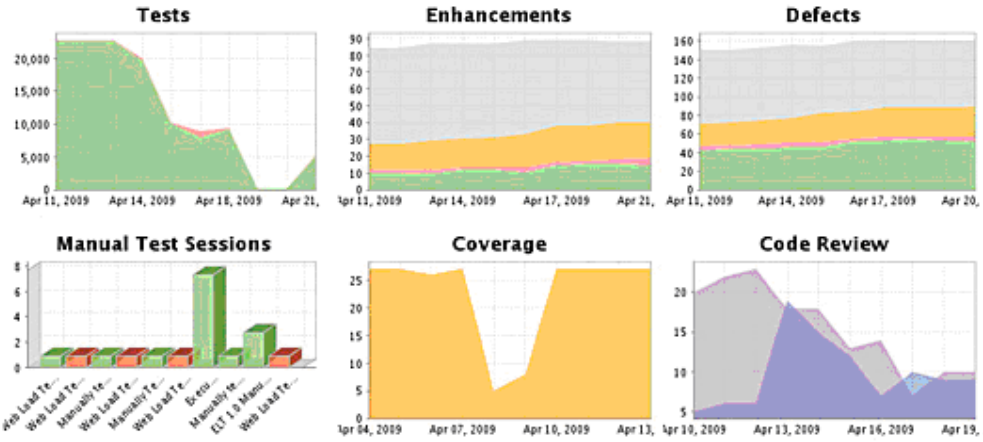

Figure5. Tested software monitor graph

Testing Module	Testing Type	Testing Case	Testing PC	Testing Result
Create table	implement	im_create_table	PC1	success
Create table	implement	im_create_default_tab	PC1	success
Insert data	implement	im_opt_insert	PC2	failed
Querydata	implement	im_opt_query	PC2	success

Figure6. Intermediate Testing Result

Testing-task Name	Testing Module	Testing Type	Testing Case	Testing Result
MySQL	Create table	implement	im_create_table	success
MySQL	Create table	implement	im_create_default_tab	success
MySQL	Insert data	Implement	im_opt_insert_crct_data	success
MySQL	Insert data	implement	im_opt_insert_wrg_data	failed
MySQL	Querydata	implement	im_opt_query	success
MySQL	Create fold	installation	im_create_fold	success

Figure7.Final Testing Result

The intermediate testing result and the final testing results are shown as Figure 6 and Figure 7 respectively. According to the experimental results, the tester is only required to submit the test task, system can automatically implement sub-task partitioning, generating test subtask, allocating sub-task, collecting and analysis test results,and shape the final test report.

5. CONCLUSION

Based on the idea of making full use of cloud resources, this paper presents a distributed software test platform,which supports contineous online testing for large-scale software systems. And it can achieve test task division and scheduling, collection and analysis and other functions which provides a well solution for distributed software test. And many experiments verificate the feasibility and reliability of prototype design and architecture.

ACKNOWLEDGEMENTS

Thanks for the Cloud_Test Group of Henan University of Science & Technology!It is their deligent work that made this paper complete.

REFERENCES

[1] Lin Yuehua,Zhang Jing,Gray J.(2005) "A testing framework for model transformations" [M],Modeldriven Software Development, Research and Practice in Software Engineering.Springer,Vol.32, No.5,pp.219:-236

[2] Stocks P A,Carrington D A.(2010) "Test templates: A specification-based testing framework", Proc.of15th International Conference on Software Engineering,pp.405-414,1993.

[3] Hartman A, Nagin K. "The AGEDIS tools for model based testing"[C], International Symposium on Software Testing and Analysis, ,Vol.29,No.4,pp.129-132,2004.

[4] Marinov D,Khurshid S. "TestEra: A novel framework for automated testing of Java program" [C].Proc of 16th IEEE International Conference on Automated Software Engineering.San Diego, CA,pp.22-32, 2001.

[5] Porter A,Yilmaz C, Memon A M. "Skoll: Distributed continuous quality assurance" [C]. Proc of the 26th IEEE ACM International Conference on Software Engineering. Edingburgh, Scotlang, pp.129-127, 2004.

[6] Rankin C. "The software testing automation framework" [J].IBM System Journal:Software Testing and Verification, ,Vol.41,No.1,pp.126-139,2002.

[7] Zhu F,Ryadurgam S, Tsai W T. "Automating regression testing for real-time software in a distributed environment" [C].Proc of 1st International Sysposium on Object-oriented Real-time distributed computing.pp.373-382,1998.

[8] P. Maechling, H. Chalupsky, etc. "Simplifying Construction of Complex Workflows for Non-expert Users of the Southern California Earthquake Center Community Modeling Environment". ACM SIGMOD Record, ,Vol.34,No.3,pp.24~30,2005.

[9] H.-M. Lee, T.-Y. Lee, and M.-H. Hsu. "A process schedule analyzing model based on grid environment", Artificial Intelligence, Vol.53,No.42,pp. 938~947,2006.

[10] Xu Bowei,Li Junjun, "Discrete PSO algorithm based on pheromone", Journal of Shanghai Maritime Univeristy, Vol.32,No.3,pp.74~78,2011.

[11] Nakajima T, Tezuka H. "A continuous media application supporting dynamic QoS control on real-time mach". Proceedings of the ACM Multimedia. ACM Press,pp.289~297,1994.

[12] M. Chinnadurai, M. Joseph, （2012） "Survey on Scheduling and Allocation in High Level Synthesis",International Journal of Computer Science & Engineering Survey (IJCSES) Vol.3, No.5, pp:43-54.

[13] Zahra Askarinejad Amiri,(2012). "Challenges and Weaknesses of Agile Method in Enterprise Architecture", International Journal of Computer Science & Engineering Survey (IJCSES), Vol.3, No.6, pp:37-45.
[14] Aguilar J,Gelenbe E. "Task assignment and transcation clustering heuristics for distributed system" [J], Information Science,Vol.32,No.9,pp.199-219,1997.
[15] Md. Firoj Ali, Rafiqul Zaman Khan,(2012) " The Study on Load Balancing Strategies in Distributed Computing System " International Journal of Computer Science & Engineering Survey (IJCSES), Vol.3, No.2, pp:19-30.

Authors

Shi Hengliang, Ph. D, associate processor, research intrest includs cloud computing, distributedcoumputing, software testing method and engineering.