ENHANCEDDYNAMICINFRASTRUCTURE AS A SERVICEMODEL (EDIAAS) FOR SELECTING SERVICE PROVIDER

Yazeed Al Moaiad[1], Zainab Abu Bakar[2], and Najeeb Abbas Al-Sammaraie[3]

Al-Madinah International University, Malaysia
[1]yazeed.alsayed@mediu.edu.my
[2]zainab.abubakar@mediu.edu.my
[3]dr.najeed@mediu.edu.my

ABSTRACT. The number of cloud service providers are expanding their services differ from one another. These have caused difficulty for user to choose the best services for a particular application. It is a tedious process for a user to search and select service providers before a chosen service can be extracted and compiled according to user preference. This process will repeat until all the desired services by user are compiled. Then the user has to rank the providers and make decision which provider fulfilled his need. Unfortunately user has to search the information again as it will be updated by the service provider. In this paper, we propose an Enhanced Dynamic Infrastructure as a Service Model (EDIAAS) for selecting service providers based on user need that incorporate important techniques such as worker role, cache redis and SignalR. The prominent services considered will be infrastructure as a service focusing on the speed of central processing unit (CPU), the size of random-access memory (RAM), the size solid-state drive (SSD), the bandwidth in bits per second (bit/s), and the cost of service. The experiments conducted shows that incorporating techniques such as worker role, cache redis and SignalR has increase system efficiency and reduce the time to display up to date information and the results instantly. This will ease the efficiency of user's search and ranking task in selecting cloud service provider.

Keywords: cloud service providers, user preference, infrastructure as a service, efficient search

INTRODUCTION

Goscinski and Brock (2010) has proposed a general structure of Cloud service publication, discovery and selection however there is no methodology proposed in their work. The Cloud service selection can usually be solved by Cloud service comparison in view of target execution investigation (Li et al., 2011). CloudCmp (Li et al., 2010), an efficient comparator, can be connected to look at three parts of the execution and expense of a Cloud, that is, flexible registering, diligent capacity, and intra-Cloud and wide range organizing (Li et al., 2010). These examinations are acknowledged by an arrangement of standard benchmark instruments, whose outcomes show the target evaluation of a Cloud.

Binnig et al. (2009) consider the differences of Cloud the measurements of versatility, cost, top load and adaptation to internal failure. Another discourse of Cloud benchmark testing is introduced by Lenk et al. (2011). In their work, they call attention to that the execution markers gave by Cloud suppliers may not be sufficient to judge the genuine execution of a virtual machine, and propose another execution estimation system, which considers the sorts of services executed on a virtual machine for Infrastructure-as-a-Service mists. As of late, some outsider associations such as, CloudHarmony have begun to offer Cloud observing and benchmarking services (Leitner & Cito, 2016). In customary e-trade or e-service situations, service choice for the most part relies on upon the notoriety based trust assessment of services. Contrasting with right on time trust assessment approaches in view of processing a solitary trust esteem for an service (Li and Wang, 2008; Li, Wang and Varadharajan, 2009) proposed a few trust vector based assessment approaches, where a trust vector is ascertained to reflect both the present dependability of an service and its trust trend. Such trust values or vectors are all evaluated from evaluations which speak to the subjective appraisal of services given by service buyers, keeping in mind the end goal to consider subjective parts of a Cloud service(Li and Wang, 2010).

Rehman et al. (2012) proposed a basic system for checking Cloud execution in view of client criticism, in which the execution of a Cloud service is observed and anticipated by clients' input. Their methodology just considers Cloud clients' subjective evaluation. There is no component to check the dependability of clients' criticism. What's more, the target appraisal of a Cloud service is not considered in their structure. Another answer for Cloud service choice issue is to demonstrate the issue as a multi-criteria choice making (MCDM) issue(Jain, Tanniru & Fazlollahi, 1991), which can be regularly fathomed by Analytic Hierarchy Process (AHP) (Muralidhar, Santhanam and Wilson, 1990).

Godse and Mulik (2009) concentrated on the choice of Software-as-a-Service mists in light of AHP. Five elements, that is, usefulness, construction modeling, and ease of use, seller notoriety and expense, are considered in their methodology. It ought to be noticed that every one of these elements aside from expense can barely be quantified by a goal measure. Hence, their methodology is still for the most part in view of subjective appraisal. Another AHP-based Cloud examination methodology is proposed by Garg, Versteeg and Buyya (2013), In their work, they endeavor to institutionalize the execution qualities for Cloud correlation. Be that as it may, the institutionalization for a few qualities, for example supportability and straightforwardness, are excessively basic, making it impossible to reflect the intricate circumstances of Cloud services in this present real world.

The services provided are subject to changes and thus the user cannot rely on the decision made earlier. Thus, the first objective of this research is to construct a Dynamic Infrastructure, as a Service Model (DIAAS) for selecting service providers based on user preferencesthat is up-to-date. Secondly, in order to improve time efficiency, the DIAAS model need to be incorporated important techniques such as worker role, cache redis and SignalR. The new model is known as Enhanced Dynamic Infrastructure as a Service Model (EDIAAS). Finally to show the efficiency of EDIAAS.

CONSTRUCTION OF DYNAMIC IAAS MODEL (DIASS)

The steps in the construction of DIAAS are shown in Figure 1. The data values for service providers are dynamically retrieved and grabbed using web services. This is automatically carried out by an intelligent tool, ITOOL since there is no standard tool available. ITOOL will visit the data page on each site, do intelligent searching and recording for the highest value plan. This will ensure to capture any changes in the arbitrary data and processed the data to the standard data. The tool will continue looping to examine the values of each row on each table, inside each provider site. The intelligence tool algorithm will perform the following steps;

1. Fetching the pricing web page and detect all the tables in that page.
2. Detecting the needed pricing tables using regular expressions to find the needed data.
3. Looping over all of the tables and data sets until reaching the table with the highest values in CPU, RAM, SSD, bandwidth and cost.
4. Storing the highest value inside a new variable and populating using standard way JSON.
5. Processing all the html tags using special libraries. Importing the data in text format, and keeping track the location of the table and cell.

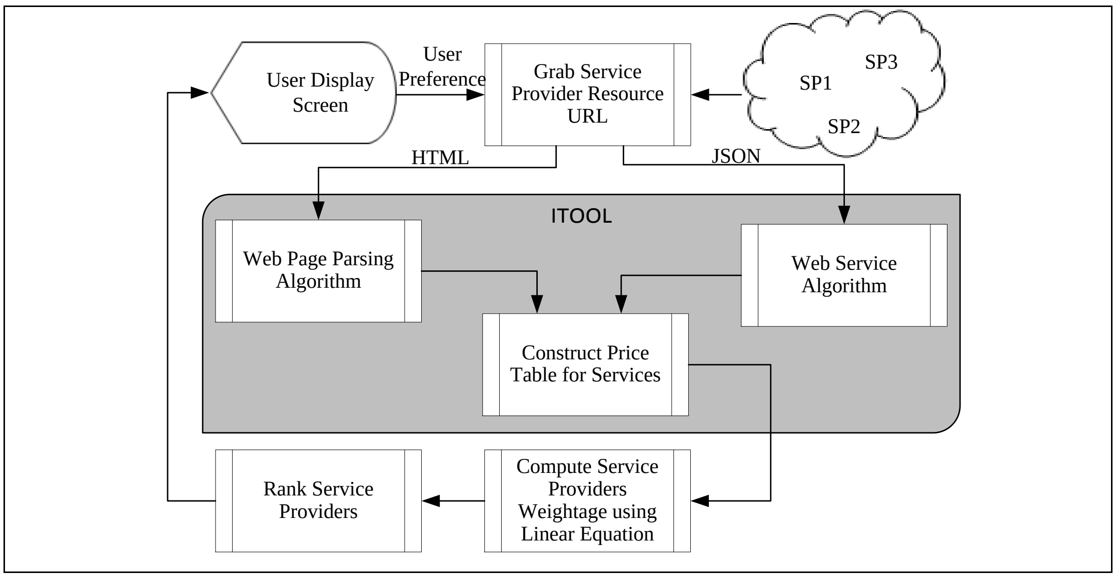
User Display Screen
User Preference
Grab Service Provider Resource URL
SP1
SP2
SP3
HTML
JSON
ITOOL
Web Page Parsing Algorithm
Web Service Algorithm
Construct Price Table for Services
Rank Service Providers
Compute Service Providers Weightage using Linear Equation

Figure 1. Dynamic Infrastructure as a Service Model (DIAAS).

CONSTRUCTION OF ENHANCED DYNAMIC IAAS MODEL (EDIASS)

As mentioned earlier, DIASS searches and ranks providers according to user providers dynamically. This model can be further improved by reducing time using important techniques such as worker role, cache redis and SignalR. Figure 2 shows the phases and steps in the construction of EDIAAS. There are four phases in implementing EDIAAS. Note the usage of ITOOL constructed in DIAAS is employed in EDIASS.

Phase 1: Connect to server

1. Connect to the cloud provider and retrieve data during server startup.
2. Store the retrieved data into the cache redis.
3. Retrieve the data from the cache redis.
4. Display the retrieved data to the user.
5. Trigger the worker role to perform the following tasks in the background in parallel.

Phase 2: If web service

1. Invoke the cloud provider web service.
2. Compare the retrieved data to the data stored in the cache redis.
3. If there is mismatch;
   a. Update the cache redis
   b. Notify the user with the updated data using SignalR technology

Phase 3: If web page parsing

1. Connect to the cloud provider and retrieve the pricing html page.
2. Store the retrieved pricing html page in memory.
3. Traverse the pricing html page.
4. Find the location of the required values based on the location of the table containing those values.
5. Compare the values found in the specified location to the values in the cache redis.
6. If there is mismatch;
   a. Update the cache redis.
   b. Notify the user with the updated data using SignalR technology.

Phase 4: Selecting

1. The user selects data preference.
2. User submits the data to the server.
3. Use linear equation algorithm to calculate the cloud provider's rankings.
4. Display the cloud provider's rankings to the user.

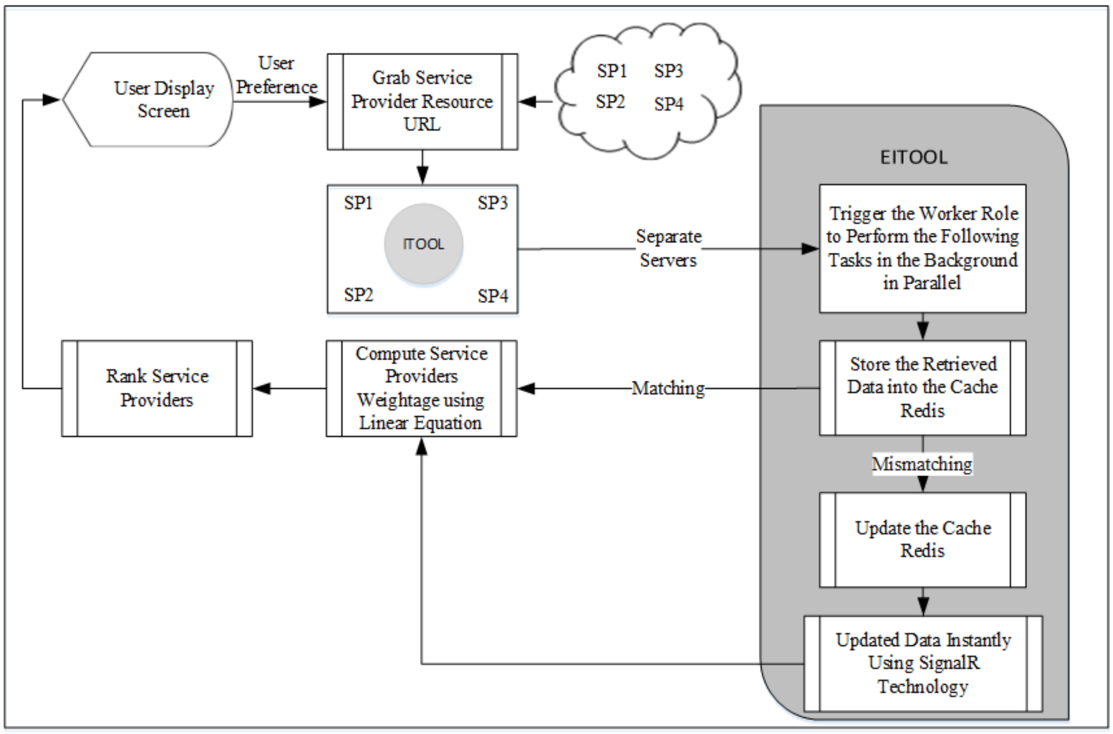
User Display Screen
User Preference
Grab Service Provider Resource URL
SP1
SP3
SP2
SP4
EITOOL
ITOOL
Separate Servers
Trigger the Worker Role to Perform the Following Tasks in the Background in Parallel
Rank Service Providers
Compute Service Providers Weightage using Linear Equation
Matching
Store the Retrieved Data into the Cache Redis
Mismatching
Update the Cache Redis
Updated Data Instantly Using SignalR Technology

Figure 2.Enhanced Dynamic Infrastructure as a Service Model (EDIAAS).

RESULT AND DISCUSSION

The two dynamic models, DIAAS and EDIAAS are compared in terms of three factors, that is, capacity, load and time for a user whose that's score are values of 1-3. A score 1 indicates user requiring in high level of the service; 2 implies medium level and 3 indicates low level. Table 1 shows an example of user requirement.

Table 1. Requirement Level of Service Needs of Experimental User.

	SSD	RAM	CPU	Bandwidth	Cost/Month
User	3	2	2	2	3

Table 2 shows the performance measures of DIAAS and EDIAAS in terms of capacity, load and time based on experimental user's requirement. Capacity, load and time are measured using JMeter tool. Table 3 summarizes the differences between static and dynamic models.

Table 2. Performance Measures between DIAAS and EDIAAS.

Factors	**DIAAS Model**	**EDIAAS Model**
Capacity	68%	93%
Load	500/responds	500/ responds
Time/Average	23.424/s	11.834/s

Table 3. Performance between DIAAS and EDIAAS.

Factors	**DIAAS Model**	**EDIAAS Model**
Capacity	Acceptable	Capability
Load	Reasonable	Sustainable
Time	Fast	Instantly

The capacity designed is to handle the number of users and processes. DIAAS has the capacity of 68%, which is acceptable. In contrast, EDIAAS model has the capacity of 93%. The load testing for 500 concurrent users is considered reasonable as the number approaching 450 loading become slower and the errors increased to 32% when the users reaches 490. Thus, DIAAS is considered a failure. While EDIAAS is able to load, continuosly with only 7% error. Finally, the response time is the average time to fetch the homepage is 23.424/s for DIAAS and 11.834/s for EDIAAS. This implies that the user instantly receive the result although the number of users is less than 500.

CONCLUSION AND FUTURE WORKS

The experiments conducted shows that incorporating techniques such as worker role, cache redis and SignalR has increase system efficiency and reduce the time to display up to date information and the results instantly. This will ease the efficiency of user's search and ranking task in selecting cloud service provider. The model is not only help users to select the best service provider but also to understand their own requirements and serves to assign a ranking on provider companies to make the process of researching for providers easier.

Future works will be more comprehensive and take into account the reputation of providers, which contributes to a high trust efficiency. It is intended to apply this in the model in a practical way. It also helps provider companies to judge their performance by trying to face the challenges that comes with increased users with various priority of services.

REFERENCES

Binnig, C., Kossmann, D., Kraska, T., &Loesing, S. (2009). How is the weather tomorrow?: towards a benchmark for the Cloud. In *Proceedings of the Second International Workshop on Testing Database Systems* (p. 9).

Garg, S. K., Versteeg, S., & Buyya, R. (2013). A framework for ranking of cloud computing services. *Future Generation Computer Systems*, *29*(4), 1012-1023.

Godse, M., & Mulik, S. (2009, September). An approach for selecting software-as-a-service (SaaS) product. In *Cloud Computing, 2009. CLOUD'09. IEEE International Conference on* (pp. 155-158). IEEE.

Goscinski, A., & Brock, M. (2010). Toward dynamic and attribute based publication, discovery and selection for cloud computing. *Future generation computer systems*, *26*(7), 947-970.

Jain, H. K., Tanniru, M. R., & Fazlollahi, B. (1991). MCDM approach for generating and evaluating alternatives in requirement analysis. *Information Systems Research*, *2*(3), 223-239.

Leitner, P., & Cito, J. (2016). Patterns in the Chaos—A study of performance variation and predictability in public IaaS clouds. *ACM Transactions on Internet Technology (TOIT)*, *16*(3), 15.

Lenk, A., Menzel, M., Lipsky, J., Tai, S., & Offermann, P. (2011, July). What are you paying for? performance benchmarking for infrastructure-as-a-service offerings. In *Cloud Computing (CLOUD), 2011 IEEE International Conference on* (pp. 484-491). IEEE.

Li, L., & Wang, Y. (2008, September). A trust vector approach to service-oriented applications. In *Web Services, 2008. ICWS'08. IEEE International Conference on* (pp. 270-277). IEEE.

Li, L., & Wang, Y. (2010). *Subjective trust inference in composite services*. In *AAAI* (pp. 1377–1384).

Li, A., Yang, X., Kandula, S., & Zhang, M. (2011). Comparing public-cloud providers. *IEEE Internet Computing*, *15*(2), 50-53.

Li, A., Yang, X., Kandula, S., & Zhang, M. (2010, November). CloudCmp: comparing public cloud providers. In *Proceedings of the 10th ACM SIGCOMM conference on Internet measurement* (pp. 1-14). ACM.

Li, L., Wang, Y., & Varadharajan, V. (2009, July). Fuzzy regression based trust prediction in service-oriented applications. In *International Conference on Autonomic and Trusted Computing* (pp. 221-235). Springer Berlin Heidelberg.

Muralidhar, K., Santhanam, R., & Wilson, R. L. (1990). *Using the analytic hierarchy process for information system project selection. Information & Management*, *18*(2), 87-95.

Rehman, Z., Hussain, O. K., Parvin, S., & Hussain, F. K. (2012, July). A framework for user feedback based cloud service monitoring. In *Complex, Intelligent and Software Intensive Systems (CISIS), 2012 Sixth International Conference on* (pp. 257-262). IEEE.