INFORMATION TECHNOLOGY JOURNAL

Information Technology Journal 11 (7): 868-875, 2012
ISSN 1812-5638 / DOI: 10.3923/itj.2012.868.875
© 2012 Asian Network for Scientific Information

Efficient Service Selection Middleware using ELECTRE Methodology for Cloud Environments

[1]Salaja Silas, [1]Elijah Blessing Rajsingh and [2]Kirubakaran Ezra
[1]Karunya University, Coimbatore, India
[2]BHEL, Trichy, India

Abstract: Cloud computing delivers on-demand services over the network. There are many service providers that provide service with similar functionality in the cloud. The problem of satisfying the user's request as per their requirements is a vital process. Service Selection in a Cloud can be modeled as a multi-criteria decision making problem since many requirements such as turnaround time, service cost, trust, reliability, etc., influence the selection of the cloud service. In this study, an efficient service selection middleware for Cloud environment is proposed adopting ELECTRE methodology. Experimental analysis has been performed on the proposed middleware. The results show that the proposed middleware is efficient.

Key words: Cloud service discovery, multi-criteria decision, cloud service selection, ELECTRE

INTRODUCTION

The cloud computing is a promising and a challenging domain of the future delivering on-demand services as a utility over the internet. Cloud computing provides various services to the users such as computation, software, data access and storage. Based on the type of services provided, cloud computing can be classified into three categories. They are Infrastructure as a Service (IaaS), Platform as a Service (PaaS) and Software as a Service (SaaS) (Kecskemeti *et al*., 2011). Based on the deployment models, the cloud has been classified into private, community, public and hybrid clouds (Zissis and Lekkas, 2012). Few examples of cloud are Amazon EC2, Google App Engine, Salesforce, Window Azure, IBM cloud (Zhao *et al*., 2012). The features of cloud computing is that the user does not require knowledge of the physical location and configuration of the system that delivers the services. The users only pay for the amount of time they use the services that reduces the cost to a great extent. The main characteristics of the cloud are flexibility, scalability, location independence and reliability.

Many users tend to use the services provided by the cloud. But since multiple service providers may provide similar services in the cloud with different characteristics, the users find difficulty in identifying their best service based on their requirements. The preferences also vary from one user to another based on their requirements. Selecting the best service based on user's requirements is a complex process due to the multiple user requirements. This article proposes an efficient service selection middleware using ELECTRE methodology for cloud environments.

In recent years, many researchers have focused on proposing feasible mechanisms to select appropriate service providers from the cloud. Some of the service selection schemes used replication (Wendell *et al*., 2010), ranking (Zhao *et al*., 2012) and cluster based searching (Goscinski and Brock, 2010) techniques.

Wendell *et al*. (2010) proposed DONAR, a distributed system that provided an interface to specify the mapping policies. The distributed mapping nodes executed an algorithm that coordinated the replica selection decisions for users. The protocol considered both client performance and server load and proved the algorithm was stable and effective. But DONAR did not consider the user preferences such as time, cost etc.

Zhao *et al*. (2012) proposed a Service Provider Search Engine (SPSE) innovative service selection algorithm that would find the appropriate service considering the user's multiple QoS requirements. The QoS requirements considered were service's response time, trust degree and monetary cost. It was found that it could capture user's preferences value in less than six times of job submissions. Even if the user preferences had changed, the preference value would be recaptured. SPSE had not considered the fault tolerance and preemptible issues in the service scheduling problem for SOA systems. SPSE had not discussed about the performance when more QoS factors were considered.

Goscinski and Brock (2010) proposed a technology for publication, discovery and selection based on dynamic attributes. The dynamic attributes expressed the

Corresponding Author: Salaja Silas, Department of Information Technology, Karunya University, Coimbatore, Tamil Nadu, 641114, India

current state and characteristics of cloud services and resources. The technique was an application of the Resources Via Web Services middleware (RVWS) to offer higher level abstraction of clouds in the form of a new technology. Cluster concept allowed easy publication, discovery and selection via a simple interface using Web pages. Instead of spending time and effort locating, evaluating and learning about clusters, clients were able to easily discover, select and use the required resources was considered as an important feature for considering this new technique. RVWS middleware had not discussed about user personalization.

Zhou *et al.* (2011) proposed unstructured P2P paradigm for service discovery in cloud. A hybrid search scheme was proposed for service query routing that couples with a number of components including one-hop replication, semantic-aware message routing, topology reorganization and supernodes for enhanced system performance. The concept of semantics to both query message routing and topology reorganization played an important role. QoS parameters were not considered in the P2P paradigm.

Kecskemeti *et al.* (2011) proposed an automated virtual appliance creation service that aided the service developers to create efficiently deployable virtual appliances. The effectiveness of service deployment systems affected initial service response times. The algorithm decomposed the virtual appliances in order to replicate the common virtual appliance parts in IaaS systems. Those parts were used to reduce the deployment time of the service by rebuilding the virtual appliance of the service on the deployment target site. In order to reduce the service cost a solution was proposed based on the partial replication of virtual appliance contents where the replicated parts were defined automatically by a decomposition algorithm and a mechanism to rebuild the decomposed virtual appliances on the target site. The automated virtual appliance creation helped the reduction of the service cost from user perspective and stated that service cost plays a major role in service selection.

Han and Sim (2010) proposed cloud service discovery that consisted of cloud service reasoning agent, an agent-based discovery system that consulted ontology when retrieving information about cloud services. Cloud service reasoning agent enabled the service discovery to reason about the relations of cloud services and rates the search results. Cloud ontology enabled the agent to determine the relations of Cloud services using three service reasoning methods namely, Similarity, Equivalent and Numerical. Cloud ontology was successful in finding Cloud services that were closer to users' requirements. Multiple criteria were not considered.

Ruiz-Alvarez and Humphrey (2011) proposed an automated approach that selected the best cloud storage service of a given application. The approach relied on a machine readable description of the capabilities of each storage system processed together with the user's specified requirements. The result was an assignment of datasets to storage systems that had multiple advantages. They were the resulting match met the performance requirements and estimated cost. The users were allowed to express their storage requirements using high-level concepts rather than reading the documentation from different cloud providers. Since the approach was based on machine readable description, it was suitable for storage service alone.

Wang *et al.* (2011a) proposed a mixed integer programming model to select optimal services. The model first computed the QoS uncertainty to prune redundant services in order to extract reliable services. The next best match is not considered by the model. Wang *et al.* (2012) proposed a fast service selection approach by using fuzzy logic control. The approach had adopted fuzzy logic control to support fast and dynamic service selection and mixed integer programming assisted users to obtain suitable services.

Wang *et al.* (2011b) proposed service selection constraint model that used the dynamic configuration method of web service composition. The functional dependency relationships between component services were defined as the service selection constraints. The Pareto optimal solutions were searched by a special ant colony optimization algorithm for web service. Gao and Kang (2012) proposed cloud simulation scheduling algorithm based on multi-dimension QoS. Analytic hierarchy process was introduced into the Resource scheduling algorithm to compute every dimensional parameters weight and then the tasks were allocated to appropriate resource according to customer satisfaction, QoS distance and loading equilibrium, etc. Dingguo *et al.* (2011) proposed a trust cloud-based subjective trust assessment and management model. The model provided the design of trust cloud, the policy of the obtainment to compute the trust information and supplied trust decision based on trust cloud model.

Talib *et al.* (2011) proposed Multi Agent System (MAS) architecture using prometheus, an agent-oriented software engineering methodology. The methodology propagated information that ensured consistency between various parts of the design. The proposed MAS architecture includes five types of agents:cloud service provider agent, cloud data confidentiality agent, cloud data correctness agent, cloud data availability agent and cloud data integrity agent.

Fan and Fang (2010) proposed Niche particle swarm optimization algorithm. The algorithm was an integration of Simulated Annealing (SA) and niche technique into Particle Swarm Optimization (PSO) that inherited the rapid local search ability of PSO and global convergence of SA. Yin *et al*. (2010) proposed QoS-aware services selection for web service composition in multi network environment. The service selection approach was based on Integer Programming that met the requirements in multi-network environment. Hussain *et al*. (2008) proposed service discovery and controlling of services with ubiquitous remote manager helped to find the appropriate service according to the user preference.

Hwang *et al*. (2012) proposed semantic-based service substitution that overcame service substitution and service similarity problems that occur in the service-oriented community after the service discovery and service composition problems such as unavailability. Liu (2010) proposed Vitrual device that focused on the autonomy of a device. Vitrual device had function agents that had the ability for auto-configuration, auto-announcement, auto-service discovery.

USER REQUIREMENTS THAT INFLUENCE SERVICE SELECTION IN CLOUD

It is obvious from the literature that many requirements such as cost, time, trust, responsiveness etc influence the selection of the service providers in the cloud. In fact, different service providers in a cloud can provide similar services with same functionality but can possess different criteria. Moreover, the requirements and the preferences of the users will also vary. In this section, few factors that influence the selection of the service provider in a cloud have been provided.

Turnaround time: Turnaround time is the time span from request submission by a user to request completion. It includes the service selection time, execution time of the job and also the network transmission time (Zhao *et al*., 2012).

Trust degree: More often, the user prefers higher trust relationships with the cloud provider and the service provider before exchange of any information from the user's side. However, for some service requirements where confidentiality is not an important preference the user may select a lower level of trust degree. (Zhou *et al*., 2011; Ahamed and Sharmin, 2008; Wang and Vassileva, 2007).

Service cost: Service cost is the cost for utilizing the services of the service provider. Most providers have fixed cost for their service (Zhao *et al*., 2012; Kecskemeti *et al*., 2011).

Capability: The number of services that the service provider will be able to handle at a particular instance of time (Ahamed and Sharmin, 2008).

Responsiveness: The time taken for the service provider to respond to the service request (Wendell *et al*., 2010; Dabrowski *et al*., 2007).

Reliability: The ability of the service provider to satisfy the requirements of the user without any service disruption (Zissis and Lekkas, 2012; Zhu *et al*., 2007; Sailhan and Issarny, 2005).

Scalability: The middleware should be capable to support a large number of service providers and a large number of users (Wendell *et al*., 2010; Zhao *et al*., 2012; Zhou *et al*., 2011).

Flexibility: The ability to meet the requirements of heterogeneous users (Wendell *et al*., 2010; Zhao *et al*., 2012).

The above factors influence the service selection in cloud environments. The requirements of each user may also vary from time to time. Many service providers in the cloud can also provide identical services. Therefore selecting a service based on user requirements is a challenge. The user can have multiple requirements. Therefore selecting the best service can be modeled as a multi-criteria decision making problem. Hence providing the user with the appropriate best service efficiently and effectively based on the user's requirements and preferences is a major research problem in the cloud. Therefore an attempt has been made to propose a middleware that provides service to the user.

PROPOSED SERVICE SELECTION MIDDLEWARE USING ELECTRE METHODOLOGY IN CLOUD (SSM_EC)

The proposed service selection middleware using ELECTRE methodology is shown in Fig. 1. The Service Selection Middleware provides Request Processing Unit, Service Selection Unit, Job Monitoring Unit and Service Delivery Unit. The Request Processing unit helps the users to submit the request for the desired service along with the user requirements and preferences for the criteria. This unit assigns a job ID for the user request and stores as job request for further processing. Service Selection unit receives the job request, selects the best service provider in the cloud as per users' requirement and

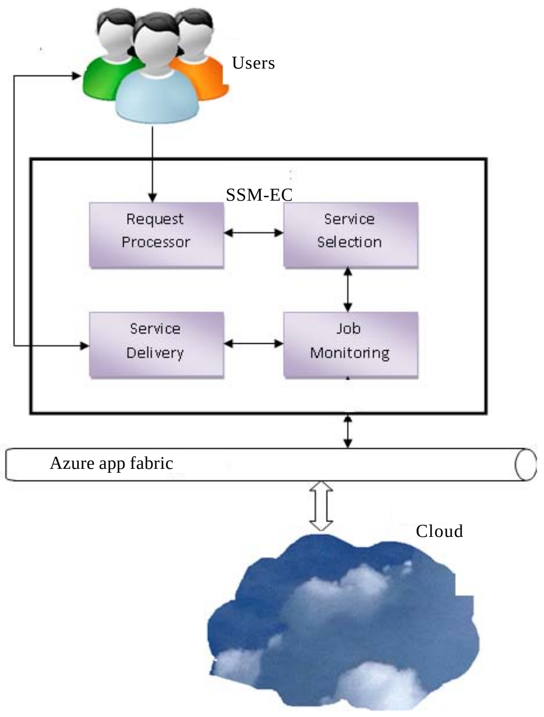

Fig. 1: Proposed service selection middleware for cloud environments

assigns the job. The services in the cloud are obtained using Azure AppFabric. The service selection unit provides the information of the job details and the information of the selected service provider to the job monitoring unit. The execution of the job is been monitored. Once the job has been completed the information is sent to the service delivery unit which also calculates the cost for the service and gives the information to the user.

Selection of the best service based on user requirements in a cloud is a multi criteria decision making problem. Along with the type of service, preference values of user for every criterion plays a major role in service selection. Our proposed Middleware implements ELECTRE Methodology (Giannoulis and Ishizaka, 2010; Triantaphyllou *et al*., 1998; Leyva-Lopez and Fernandez-Gonzalez, 2003; Doumpos *et al*., 2009) (Elimination and Choice Translating Reality) a multi criteria analysis method used to solve multi criteria problems which are considered to be NP- complete. ELECTRE method is one of the methodologies of Multi Utility Attribute Theory that works on pair wise comparisons to rank the service providers of the cloud. Among many forms of ELECTRE III (Marzouk, 2011; De Almeida, 2007) is used for outranking relation.

The methodology requires two types of inputs for processing. They are information from the user and the service providers of the cloud.

S	$cr_1(.)$	$cr_2(.)$	…	$cr_j(.)$	…	$cr_k(.)$
s_1	$cr_1(s_1)$	$cr_2(s_1)$	…	$cr_j(s_1)$	…	$cr_k(s_1)$
s_2	$cr_1(s_2)$	$cr_2(s_2)$	…	$cr_j(s_2)$	…	$cr_k(s_2)$
…	…	…	⋱	…	⋱	…
s_i	$cr_1(s_i)$	$cr_2(s_i)$	…	$cr_j(s_i)$	…	$cr_k(s_i)$
…	…	…	⋱	…	⋱	…
s_n	$cr_1(s_n)$	$cr_2(s_n)$	…	$cr_j(s_n)$	…	$cr_k(s_n)$

Fig. 2: Evaluation table

- **Step 1:**
- **Information from the user:** The user's preference for each criterion cr is obtained from the user. All the preference values are normalized to 1 and are represented as weights $\{w_j, j = 1, 2, \ldots k\}$

$cr_1(.)$	$cr_2(.)$	…	$cr_j(.)$	…	$cr_k(.)$
w_1	w_2	…	w_j	…	w_k

$$\sum_{j=1}^{k} w_j = 1$$

- **Information from the service providers of the cloud:** Based on the user's requirements, the information of the set of relevant service providers are retrieved from the cloud to form the evaluation table. An evaluation table (Fig. 2) with corresponding values for each criterion is generated based on the information provided by the service providers in the cloud that influences the selection of a particular service from feasible service providers.

- **Step 2: Calculation of the concordance index (C):** The concordance index indicates the truthfulness of the assertion " s_x outranks s_y". If C is equal to 1 it indicates full truthfulness and if C equals 0 then it indicates the assertion is wrong.

$$C(s_x, s_y) = \frac{1}{W}\sum_{i=1}^{k} w_i c_i(s_x, s_y) \tag{1}$$

$$c_i(s_x, s_y) = \begin{cases} 1, & \text{if}\,(Cr_i(s_y) - Cr_i(s_x)) \le q_i \\ \dfrac{p_i(Cr_i(s_x)) + Cr_i(s_x) - Cr_i(s_y)}{p_i(Cr_i(s_x)) - q_i(Cr_i(s_y))}, & \text{if}\,(q_i < Cr_i(s_y) - Cr_i(s_x) < p_i) \\ 0, & \text{if}\,(Cr_i(s_y) - Cr_i(s_x)) \ge p_i \end{cases} \tag{2}$$

where, p is the preference threshold (Rogers and Bruen, 1998) that justifies the preference of the service provider over the other and q is the indifference when the threshold (Rogers and Bruen, 1998) that defines the value service provider can be neglected from the selection process.

- **Step 3: Calculation of the discordance index (D):** The discordance index (D) is defined to make the system cautious to refuse the assertion "s_x outranks s_y" when the difference of performance between the service provider alternative s_x and s_y on the preference cr_i is higher than the veto threshold (Rogers and Bruen, 1998) v_i.

$$D_i(s_x, s_y) = \begin{cases} 0, & \text{if}(Cr_i(s_y) - Cr_i(s_x)) \le p_i \\ \dfrac{Cr_i(s_y) - [Cr_i(s_x) + p_i]}{v_i - p_i}, & \text{if}(p_i < Cr_i(s_y) - Cr_i(s_x) < v_i) \\ 1, & \text{if}(Cr_i(s_y) - Cr_i(s_x)) \ge v_i \end{cases} \quad (3)$$

- **Step 4: Calculation of the credibility degree (CD):** Credibility degree indicates whether the outranking relationship is true or false:

$$CD(s_x, s_y) = \begin{cases} C(s_x, s_y), & \text{if}(D_i(s_x, s_y) \le C(s_x, s_y)) \forall i \\ C(s_x, s_y) \Pi_{i \in J(s_x, s_y)} \dfrac{1 - D_i(s_x, s_y)}{1 - C(s_x, s_y)}, & \text{otherwise} \end{cases} \quad (4)$$

Where J (s_x,s_y) is the set of criteria where D_i (s_x,s_y),

- **Step 5: Exploitation of the outranking relation**

- **Distillation procedure:** The distillation procedure is used to rank the feasible service providers of the cloud. Two forms of distillation are performed. First pre order is obtained with descending distillation which rates the service provider from best downwards. The second pre order is obtained with the ascending distillation which rates the service provider based on the service provider with least weakness onwards. s_x outranks s_y can be defined if the following condition holds true:

$$\left(CD(s_x, s_y) > \lambda_2\right) \text{AND} \left(CD(s_x, s_y) - CD(s_y, s_x) > f(\lambda_0)\right) \quad (5)$$

where, λ_0 = max s_x, $s_y \in s$ CD(s_x, s_y) and λ_2 is the largest credibility index just below the cut off level λ_1 λ_2= max (CD (s_x, s_y)$\le\lambda$), CD(s_x, s_y)$\forall s_x$, $s_y \in S$ such that $\lambda_1 = \lambda_0 - f$ (λ_0) is the following cut off level and f (λ_0) = $\alpha+\beta\lambda_0$ is following discrimination threshold:

Where:
α = 0.3
β = -0.15

- **Extraction:** When s_x outranks s_x is given the score+1 to represent its strength and s_Y is given -1 to represent its weakness. For each service provider alternative, the individual strengths and weakness are added together to give the final qualification score. Within the descending distillation, the service provider alternative with the highest qualification score is assigned to a rank and removed from the credibility matrix. The process is repeated with the remaining service provider alternatives until all service provider alternatives of the cloud are ranked. The ascending distillation procedure works in a similar way to the descending distillation with the exception that the procedure assigns the service provider alternative having the lowest qualification score

- **Final ranking:** The final ranking is obtained through the combination of the descending and ascending distillation pre-orders. There are four different cases which are described below:

- s_x is higher ranked than s_y in both distillations or s_x is better than s_y in one distillation and has the same ranking in the other distillation then s_x is better than s_y:

$$s_x P s_y$$

- s_x is higher ranked than s_y in one distillation but s_y is better ranked than s_x in the other distillation then s_x is incomparable to s_y:

$$s_x R s_y$$

- s_x has the same ranking than s_y in both distillations then s_x is indifferent to s_y:

$$s_x I s_y$$

- s_x is lower ranked than s_y in both distillations or s_x is lower ranked than s_y in one distillation and has the same rank in the other distillation then s_x is worst than s_y:

$$s_x Q s_y$$

Then the sum of P for every feasible service provider of the cloud is calculated. The feasible service provider which has the highest ranking is selected as the best service provider in the cloud.

RESULTS AND DISCUSSION

The cloud environment was simulated by using an ASP. NET MVC Web Application with SQL Azure using Microsoft Visual Studio 2010 in .NET Framework 4.0. Windows Azure provides high availability, scalability and manageability as the key benefits. Fifty service providers were created in a cloud to provide various services.

Experiment 1: The users who require cloud services were permitted to provide their requirements such as service type and their preference values for the criteria. The number of criteria was varied by 2, 4, 6, 8 and 10. The service selection time taken to detect the best service as required by the user was determined by varying the feasible number of service providers in the cloud as 10, 20, 30, 40 and 50 and is shown in Fig. 3. The turn-around time experienced by the user is shown in Fig. 4.

It was found that the service selection time in cloud environments increases with the increase in the number of service providers in the cloud. This is due to the fact that when the number of service providers in the cloud is greater, the evaluation table is bigger and the calculation of Concordance Index, Discordance index, Credibility degree and the exploitation of the outranking relation consumes more time. It was observed in Fig. 4 that the turn-around time for a particular job also increases with the increase in the number of service providers in the cloud. This is due to the fact that the turn-around time includes the service selection time, execution time of the job and also the network transmission time. It was also found that the number of requirements affect the cloud service Selection time as well as the turn-around time.

Experiment 2: The users who were in need of the Cloud services were permitted to provide their requirements such as service type and their preference values for the criteria. The number of criteria was varied by 2, 4, 6, 8 and 10. The service selection overhead to detect the best service as required by the user was determined by varying the feasible number of service providers in the cloud as 5, 10, 15, 20 and 25 and is shown in Fig. 5.

It was observed from Fig. 5 that the average selection overhead increases with the number of service providers in the cloud. This is due to the fact that as the number of service providers in the cloud increases the number of computational messages required to identify the best service provider in the cloud increases.

Experiment 3: The users who require cloud services were permitted to provide their requirements such as service type and their preference values for the criteria. The number of criteria was varied by 2, 4, 6, 8 and 10. The number of feasible service providers was set to 10. The service selection time to detect the best service in the cloud as required by the user was determined by varying the number of users as 200, 400, 600, 800 and 1000 and is shown in Fig. 6.

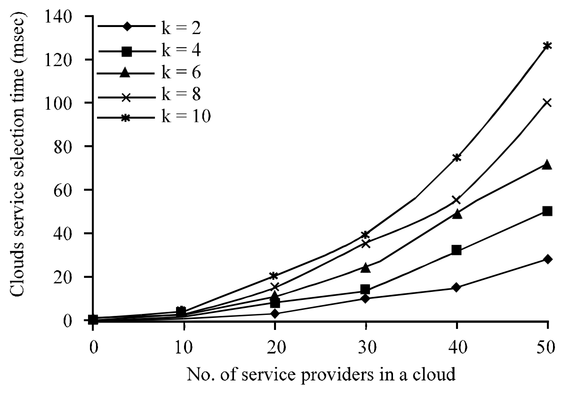

Fig. 3: Cloud service selection time

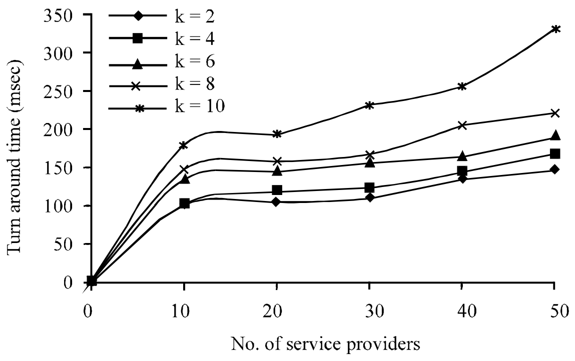

Fig. 4: Turn around time

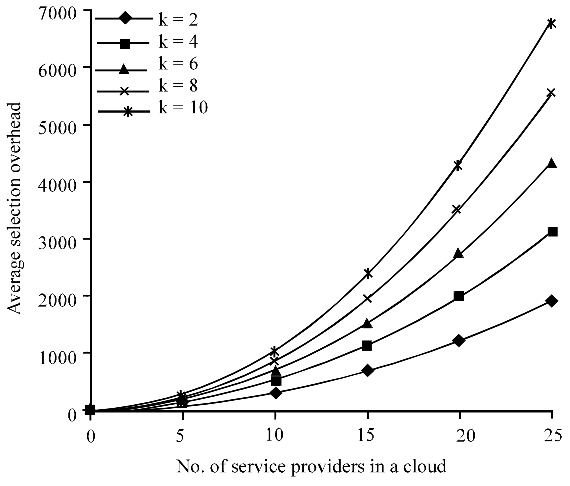

Fig. 5: Service selection overhead

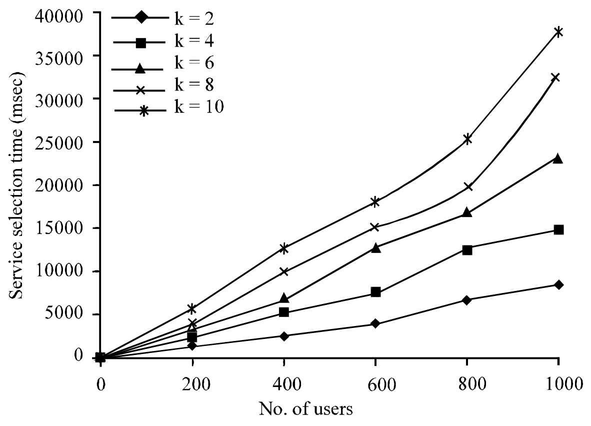

Fig. 6: No. of users vs. service selection time

It was observed from Fig. 6 that the proposed service selection middleware is scalable.

CONCLUSIONS

In this study, an efficient service selection middleware for cloud environments has been proposed. The service selection has been as modeled as multi-criteria decision making problem. Many criteria such as turnaround time, service cost, responsiveness, trust reliant, scalability, capability, flexibility, etc., influencing the service selection were considered. The experimental results show that the proposed middleware was effective.

REFERENCES

Ahamed, S.I. and M. Sharmin, 2008. A Trust-based Secure Service Discovery (TSSD) model for pervasive computing. Comput. Commun., 31: 4281-4293.

Dabrowski, C., K. Mills and S. Quirolgico, 2007. Understanding failure response in service discovery systems. J. Syst. Software, 80: 896-914.

De Almeida, A.T., 2007. Multicriteria decision model for outsourcing contracts selection based on utility function and ELECTRE method. Comp. Operat. Res., 34: 3569-3574.

Dingguo, Y., C. Nan and T. Chengxiang, 2011. Research on trust cloud-based subjective trust management model under open network environment. Inform. Technol. J., 10: 759-768.

Doumpos, M., Y. Marinakis, M. Marinaki and C. Zopounidis, 2009. An evolutionary approach to construction of outranking models for multicriteria classification: The case of the ELECTRE TRI method. Eur. J. Oper. Res., 199: 496-505.

Fan, X. and X. Fang, 2010. On optimal decision for QoS-aware composite service selection. Inform. Technol. J., 9: 1207-1211.

Gao, W. and F. Kang, 2012. Cloud simulation resource scheduling algorithm based on multi-dimension quality of service. Inform. Technol. J., 11: 94-101.

Giannoulis, C. and A. Ishizaka, 2010. A web-based decision support system with ELECTRE III for a personalised ranking of British universities. Decis. Support Syst., 48: 488-497.

Goscinski, A. and M. Brock, 2010. Toward dynamic and attribute based publication, discovery and selection for cloud computing. Future Gener. Comput. Syst., 26: 947-970.

Han, T. and K.M. Sim, 2010. An ontology-enhanced cloud service discovery system. Proc. Int. MultiConf. Eng. Comput. Sci., 1: 644-649.

Hussain, C.S., C.S. Ahmed, A.H. Akbar, A.K. Bashir, K.H. Kim and W.S. Yoon, 2008. Ubiquitous service discovery in pervasive computing environment. Inform. Technol. J., 7: 533-536.

Hwang, Y.Y., Y.S. Kim and K.C. Lee, 2012. The method for semantic service substitution in ubiquitous computing. Inform. Technol. J., 11: 540-543.

Kecskemeti, G., G. Terstyanszky, P. Kacsuk and Z. Nemeth, 2011. An approach for virtual appliance distribution for service deployment. Future Gener. Comput. Syst., 27: 280-289.

Leyva-Lopez, J.C. and E. Fernandez-Gonzalez, 2003. A new method for group decision support based on ELECTRE III methodology. Eur. J. Operat. Res., 148: 14-27.

Liu, L., 2010. Autonomy on intelligent home network. Inform. Technol. J., 9: 282-289.

Marzouk, M.M., 2011. ELECTRE III model for value engineering applications. Autom. Constr., 20: 596-600.

Rogers, M. and M. Bruen, 1998. Choosing realistic values of indifference, preference and veto thresholds for use with environmental criteria within ELECTRE. Eur. J. Oper. Res., 107: 542-551.

Ruiz-Alvarez, A. and M. Humphrey, 2011. An automated approach to cloud storage service selection. Proceedings of the 2nd International Workshop on Scientific Cloud Computing, June 8-11, 2011, San Jose, CA., USA., pp: 39-48.

Sailhan, F. and V. Issarny, 2005. Scalable service discovery for MANET. Proceedings of the 3rd IEEE International Conference on Pervasive Computing and Communications, March 8-12, 2005, Kauai Island, HI., USA., pp: 235-244.

Talib, A.M., R. Atan, R. Abdullah and M.A.A. Murad, 2011. Multi agent system architecture oriented prometheus methodology design to facilitate security of cloud data storage. J. Software Eng., 5: 78-90.

Triantaphyllou, E., B. Shu, S. Nieto Sanchez and T. Ray, 1998. Multi-Criteria Decision Making: An Operations Research Approach. In: Encyclopedia of Electrical and Electronics Engineering, Webster, J.G. (Ed.). Vol. 15. John Wiley and Sons Inc., New York, USA., pp: 175-186.

Wang, Y. and J. Vassileva, 2007. Toward trust and reputation based web service selection: A survey. Int. Trans. Syst. Sci. Appl., 3: 118-132.

Wang, S., Z. Zheng, Q. Sun, H. Zou and F. Yang, 2011a. Cloud model for service selection. Proceedings of the IEEE Conference on Computer Communications Workshops, April 10-15, 2011, Shanghai, China, pp: 666-671.

Wang, X.L., Z. Jing and H.Z. Yang, 2011b. Service selection constraint model and optimization algorithm for web service composition. Inform. Technol. J., 10: 1024-1030.

Wang, S., Q. Sun, H. Zou and F. Yang, 2012. Fuzzy logic control for web service selection. Inform. Technol. J., 11: 399-401.

Wendell, P., J.W. Jiang, M.J. Freedman and J. Rexford, 2010. DONAR: Decentralized server selection for cloud services. Proceedings of the ACM SIGCOMM 2010 Conference on SIGCOMM, August 30-September 3, 2010, New Delhi, India, pp: 231-242.

Yin, K., B. Zhou, S. Zhang, H. Jiang and J. Cristoforo, 2010. Optimizing services composition in multi-network environment. Inform. Technol. J., 9: 399-411.

Zhao, L., Y. Ren, M. Li and K. Sakurai, 2012. Flexible service selection with user-specific QoS support in service-oriented architecture. J. Network Comput. Appl., (In Press).

Zhou, J., N.A. Abdullah and Z. Shi, 2011. A hybrid P2P approach to service discovery in the cloud. Int. J. Inform. Technol. Comput. Sci., 1: 1-9.

Zhu, F., W. Zhu, M.W. Mutka and L.M. Ni, 2007. Private and secure service discovery via progressive and probabilistic exposure. IEEE Trans. Parallel Distrib. Syst., 18: 1565-1577.

Zissis, D. and D. Lekkas, 2012. Addressing cloud computing security issues. Future Gener. Comput. Syst., 28: 583-592.