Article can be accessed online at http://www.publishingindia.com

Automation of Discovery and Aggregation of Cloud Services

Sreedevi R. Nagarmunoli*, Nandini S. Sidnal**

Abstract

Cloud computing is a technology where IT-related resources are dynamically provided "as a service" to the customers through Internet. The customer can demand for the services dynamically from Cloud Service Provider (CSP)s, take them on lease based on Service Level Agreement (SLA), release the resources after completion of task and pay for what is used. The required services may not be available from a single CSP. There are many CSPs providing multiple services with different Quality of Service (QoS). The customer has to discover the available services with the expected QoS which is one of the major challenges to be solved in cloud computing today. In this paper we dynamically create a repository of the cloud services and aggregate them whenever there is a demand for service and then derive that the services obtained from the repository are time efficient as compared to direct service discovery.

Keywords: Repository, Aggregation

Introduction

Cloud is a very popular technology today that provides "Everything as a Service" to the customers. It has become one of the important utility. There has been an exponential increase in the number of Cloud Service Providers. They can potentially publish various Cloud services and the consumers can access these services through the Internet. The customer can demand for the services dynamically, get them on lease based on Service Level Agreement (SLA), release them after completion of task and pay for what they utilize. Cloud delivers services on demand at a very reasonable price thus huge capital investment is not required. However, there are no standards, open protocols or search mechanisms for discovering different kinds of Cloud services and consumers have to discover the Cloud services manually. Hence, there is a need for a repository (directory) that automatically discovers the available services, aggregates them based on the QoS requested, negotiate with the SLA, schedule the services to the customers.

Aggregation here is about providing the data integration, process integrity or intermediation needed to bring multiple services together. We combine multiple point services into one aggregated service. It ensures that data is modeled across all component services and integrated as well as ensuring the movement and security of data between the service consumer and multiple providers. Gartner (2015) opines that cloud computing looks similar to the already existing paradigms: World Wide Web, grid computing, service computing, and cluster computing. But the differences are the characteristics such as on-demand resource pooling, self-service, high scalability, end-to-end virtualisation support, and robust support of resource usage metering and billing, services that are offered under pay-as-you-go-model, guaranteed SLA, faster time to deployments, lower upfront costs, little or no maintenance overhead, and environment friendliness.

Research is going on in cloud computing by several researchers all over the globe. Mainly research is

* Associate Professor, Department of Computer Science and Engineering, KLE DR MSS College of Engineering and Technology, Belaeavi, Karnataka, India. Email: srnmunoli@gmail.com

** Professor, Department of Computer Science and Engineering, KLE DR MSS College of Engineering and Technology, Belaeavi, Karnataka, India. Email: sidnal.nandini@gmail.com

being conducted in areas like resource management, load balancing, security and trust, scheduling services, scalability, elasticity, data management, energy management, cloud interoperability and some non-technical issues. Research in automation of cloud service discovery and repository development is not much paid attention. This paper focuses on implementing automated service discovery and further more aggregation of cloud services based on users requirements. It presents literature survey in the second section, followed by some issues and challenges. The third section details the proposed system and implementation. Further it presents some test cases of the actual implementations and result analysis. Finally the future scope follows the conclusion.

Literature Survey

This section explores the related works in cloud service discovery and their aggregation based on users requirements and demands. Cloud computing looks similar to the existing paradigms and mention that human familiarity is required with different types of cloud resources available. In an effort to solve the problem of resource allocation, resource scheduling, and resource provisioning Maurer, Brandic and Sakellariou (2013) say little human interaction is required for better resource utilisation than complete automation.

Manvi and Shyam (2014) mainly focus on resource management techniques such as resource provisioning, resource allocation, resource mapping and resource adaptation. Qu, Wangand Orgun (2013) propose a model of cloud service selection by aggregating the information from both the feedback and from cloud users and objective performance analysis from trusted third party using simple fuzzy adaptive weighting system. Tserpes, Aisopos, Kyriazis and Varvarigou (2010) propose service selection model based on collaborative filtering techniques.

Huang (2013) has designed a sub-optimal resource allocation system that redistributes the resources based on the current status of all virtual machines installed in physical machine. Mattess (2011) has discussed several resources provisioning algorithm to handle the demand for resources by the enterprises.

Jain, Rane and Patidar (2012) and Nair (2010) discuss cloud bursting and brokerage. Cloud bursting is a deployment model in which an application runs in a private cloud and bursts into a public cloud when the demand for computing capacity spikes. According to Gartner, Cloud Services Brokerage (CSB) makes it easier for organisations to consume and maintain cloud services Gartner (2015).

Taekgyeong and Kwang (2010) opine that the cloud service discovery system interacts with cloud ontology to determine the similarities between and among services. It uses agents to reason about relations of cloud services using similarity, equivalent and numerical reasoning. This paper does not speak of automating the entire service discovery. Leyli and Jahani (2014) rank the cloud services provided by various vendors based on QoS. It does not mention about discovering the services rather focuses only on aggregating the services. Toosi, Calheiros and Buyya (2014) discuss all relevant aspects such as architectures, challenges and issues of Cloud Interoperability.

Cao, Wen and Li (2011) construct a QoS-aware and qualitative trust model to discover and aggregate the cloud services. Cavalcante (2012) presents optimised cloud service selection approach that evaluates each alternative set of services that composed an execution plan considering cost and quality parameters for each service in the execution plan. Researchers Modica, Petralia, and Tomarchio (2012). Mındruta, and Fortis (2013) Di Modica, Petralia andTomarchio (2013) have tried to match the customers' service request with the CSP's services using ontology and semantic search algorithm. Garg, Versteeg, and Buyya (2011) have proposed a framework to automatically index services provided by the cloud service providers based on customers' needs. They measure QoS required and prioritize the cloud services and make use of the Analytical Hierarchy Process (AHP) to prioritize the services.

Tserpes, Aisopos, Kyriazis and arvarigou (2010) have proposed that the customer comes against the problem of selecting a service from a plethora of available ones. Service Level Agreement (SLA) is a fundamental tool for governing the customer-provider relationship, defining and quantifying the required levels of the Quality of Service (QoS). This paper exploits the concept of the QoS in order to allow for customers to evaluate providers' services and use these evaluations for giving recommendations to customers with similar experiences. The proposed service selection model is based on collaborative filtering techniques and especially on measuring the correlation between customer ratings using the Pearson's product-moment correlation coefficient.

Hence there is a need to have a cloud repository (directory of services) to help the customers to receive the services on lease and release. The repository should also be capable of scheduling the services and metering. Cloud aggregator is a platform or service that combines multiple clouds with similar characteristics (geographic area, cost, technology size, etc.) into a single point of access, format, and structure. Buyya, Ranjan and Calheiros (2010) propose an environment to scale the applications across multiple cloud vendors to provide the QoS expected by the customer. Mapping processes and services should maximise the system efficiency and utilisation. The key elements defined by the authors for enabling the federation of clouds and auto scaling application services are – cloud coordinators, brokers and an exchange.

Manvi and Shyam (2014) say that the resource management problems include resource allocation, resource provisioning, resource requirement mapping, resource adaptation, resource discovery, brokering, estimation and modeling. Resource management for IaaS in cloud computing offers following benefits: scalability, quality of service, optimal utility, reduced overheads, improved throughput, reduced latency, specialised environment, cost effectiveness and simplified interface. It mainly focuses on resource management techniques such as resource provisioning, resource allocation, resource mapping and resource adaptation.

Mladen (2008) investigates the challenges of developing a Campus Cloud based on aggregating resources from multiple universities. The requirements and architecture model of this cloud environment are presented along with an implementation methodology using open source cloud middleware. Kousiouris, Kyriazis, Varvarigou, Oliveros and Mandic (2011) extract a taxonomy of the characteristics found in modern service discovery systems and produce a categorisation of existing implementations in a grouped and comparative way, based on these features. Furthermore, the mapping of these characteristics to the Cloud Business model is produced, in order to assist in selecting the suitable solutions for each provider based on his/her location in the value chain or identify gaps in the existing implementations.

Zhang, He, Feiyi and Bin (2013) proposed a service discovery architecture that supports division of services into different Cloud Service Domains (CSD) according to the characteristics of the functions of cloud service directories and each CSD has a sub-domain with a common root directory. Zhang, Ranjan, Haller, Georgakopoulos, Menzel, and Nepal (2012) present Web Ontology Language (OWL) based ontology, the Cloud Computing Ontology (CoCoOn) that defines functional and non-functional concepts, attributes and relations of infrastructure services. They also present a system, Cloud Recommender-that implements the domain ontology in a relational model. The system uses regular expressions and SQL for matching user requests to service descriptions. It briefly describes the architecture of the Cloud Recommender system, and demonstrates its effectiveness and scalability through a service configuration selection experiment based on a set of prominent Cloud providers' descriptions including Amazon, Azure, and Go Grid.

The survey depicts that service discovery; repository development and aggregation as a package are not automated completely. Based on the survey it can be concluded that building an automated cloud repository is essential as it may reduce the cloud service search time and provide better QoS. Thus this paper attempts to accomplish the aforementioned task of service discovering, repository building and aggregation of services. From the survey some of the following issues were inferred with partial solutions.

Unavailability of a common service platform that can customize or aggregate the cloud services as per Small and Medium-sized Business (SMB) or enterprise business requirements.

1. Unable to offer multiple cloud services (IaaS, SaaS and PaaS) on a single platform.
2. Unable to deliver new services faster as per changing market trends.
3. Underutilised capacity of existing infrastructure that reduces Return on Investment (ROI).
4. Unable to offer pay-per-use model for SMBs.
5. Excess time and resources spent on aggregating and managing services from multiple platforms.
6. Algorithm to predict the future request.
7. Algorithm to dynamically discover the cloud services.
8. Deploying multidimensional data structure to store the cloud services in the repository.
9. To design unsupervised learning mechanism for the agents to negotiate intelligently to aggregate the cloud services for demanded QoS.

10. Scheduling techniques to distribute the services maximizing the availability and utility of services.
11. Cloud service utility metering algorithms is to be developed for the delivered services.

Issues 1-6 have been overcome to some extent through the proposed solution in this paper. Issues from 7-12 are yet to be addressed.

Proposed System and Implementation

In Cloud environment, service providers can potentially publish various Cloud services through Internet, and consumers can access these services by Cloud application layer through web-portals. However, there are no standards, open protocols and search mechanisms for discovering different kinds of Cloud services and consumers have to search the Cloud services manually. In addition, there has been tremendous increase in the number of cloud services and service providers, hence in an effort to help the end users to identify their required service a framework has been designed and implemented. This frame work has been tested with some inputs such as service requirements from consumers - type of service, subscription plan, hardware requirements, etc. to create an aggregated list of required services.

Architecture

The architectural pattern used here is the three tier architecture and the independent tiers of the system are the sub-repositories, main repository and an interface to the user as shown in Figure 1.

Figure 1: Architectural Pattern

Sub-repository

The sub-repository is an intermediate database that is a part of the main repository. This repository is updated automatically by using standalone application which accesses the web to get the Cloud services. These services are fetched in real time and updated at regular intervals of time. Most of the types of different Cloud services available on the web are registered in the sub-repository.

All the data fetched using this application from the web may not be relevant to build the repository. The data received has to be processed and the required information has to be extracted. The received data is parsed to extract Name of the CSP, Base plan cost, Base plan details, Subscription type, Computing category, Compatible operating system, Control interface, Free support, Software license. These fields are populated into the database of the sub-repository.

At regular intervals of time the task of auto fetching of cloud services from web is initiated. This helps to check if there are any additional services found to be populated to the intermediate database. Thus the sub-repository is always updated.

Main Repository

Main repository is the central repository which is used to serve customers' requests. The website which serves as the frontend of the system retrieves data from this repository. This repository is a union of all the sub-repositories and is populated using aglets which are a type of software agents. The software agents that are mobile in nature are responsible for collecting the data base from all repositories and merging them in main repository.

The benefit of using aglets is that the database is populated asynchronously and there is no need for a connection to be established and hence the bandwidth is utilized optimally and efficiently. Aglets are capable of carrying the data from one system to another. Hence, the data collected from the intermediate databases of sub-repositories by the agents is populated into the main repository. The main repository can be queried through the Web Interface of the system to get the services based on the required QoS such as duration of service, cost of service etc. The proposed system for the implementation of the sub-repository can be depicted as in Figure 2.

For demonstration purpose of the project, three sub-repositories were used. These repositories contain different sets of data and also act as backup repositories in case of the failure of the main repository, as the main repository can be reproduced using all the sub repositories. The proposed system for the implementation of the main

Figure 2: Architecture of Sub-Repository

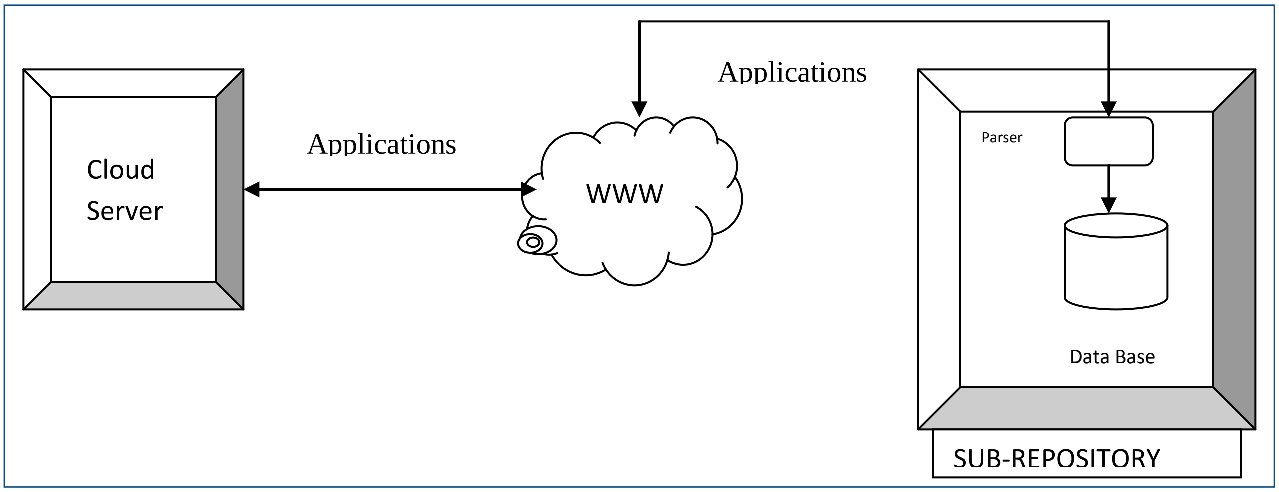
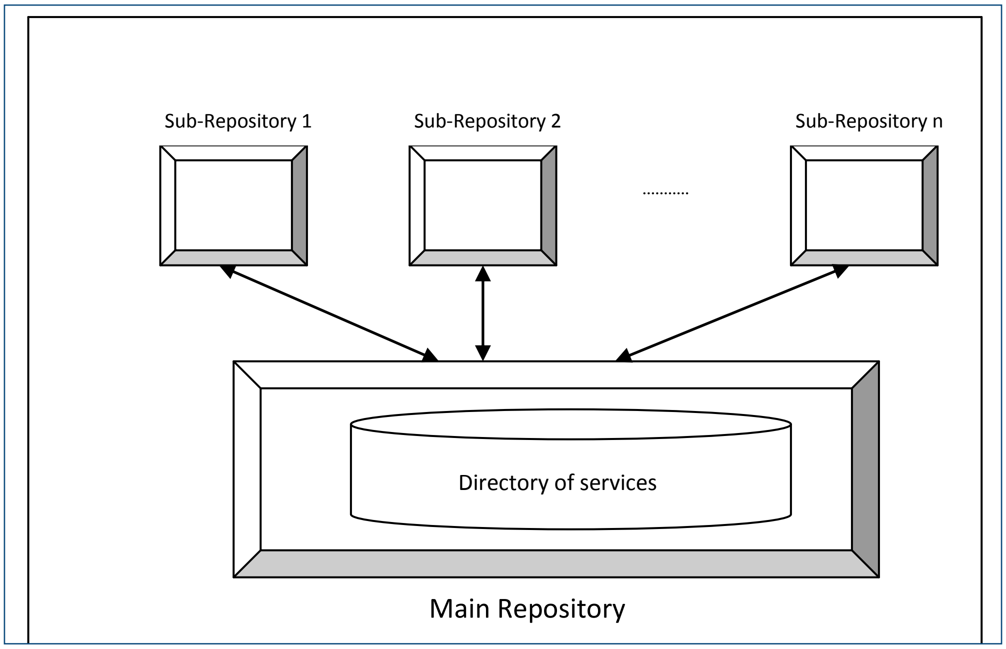


Figure 3: Architecture of Main Repository

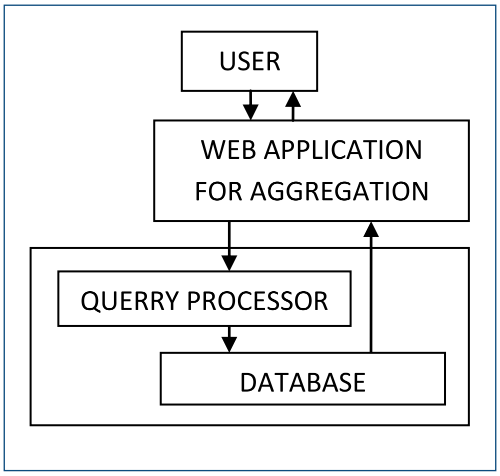


repository can be depicted as in Figure 3.

Web Interface

The Web interface forms the frontend of the system. It is a website which is hosted using Internet Information Services (IIS) on the local network for demonstration purposes. When the customer / user searches for different cloud services based on certain criteria, he is actually not searching the Web for the cloud services, instead he/she is searching the aggregator's index of the cloud services found on the Web. The webpage consists of various options to parse the cloud services based on criteria like computing category and subscription type. Also the user can search the services by writing a query in natural language e.g., 1CPU, 1GB RAM etc. The web interface is as shown in the Figure 4.

Figure 4: Web Interface



Test Cases

The implemented system was tested with different combinations of the various options provided to filter the cloud services based on criteria such as computing category and subscription type. The system was also tested where the user searches for services by writing a query in natural language. Following section provides the detailed representation of few of the many test cases that were used to test the system. The implemented system was tested on quad core system running on Windows 8 64-bit operating system, x64-based processor with 8.00 GB Internal Memory (RAM).

Test case 1: The combination of the options provided to search the cloud services are Computing category – All and Subscription type – All. The results are as shown in Figure 5 (a) and (b).

Test case 2: The combination of the options provided to search the cloud services are Computing category – All and Subscription type – Pay as you go. The results are as shown in Figure 6 (a) and (b).

Test case 3: The combination of the options provided to search the cloud services are Computing category – Infrastructure as a Service and Subscription type – Subscription plan.

Figure 5 (a): Results of Computing Category – All, Subscription type – All

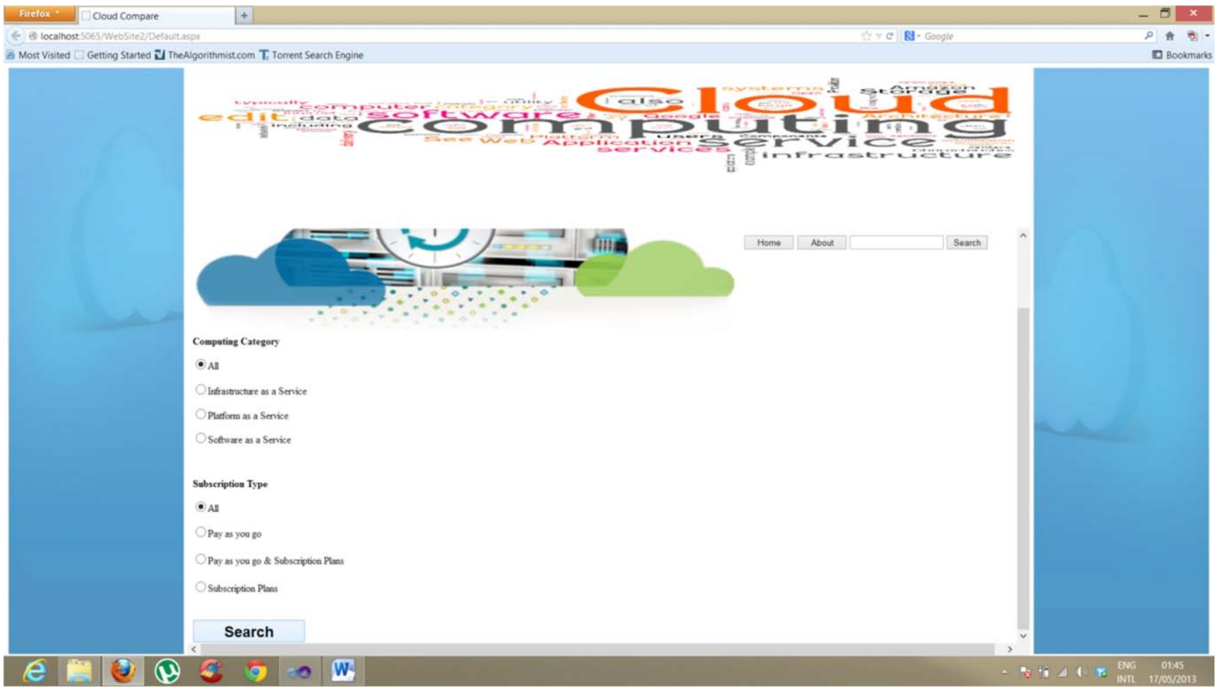
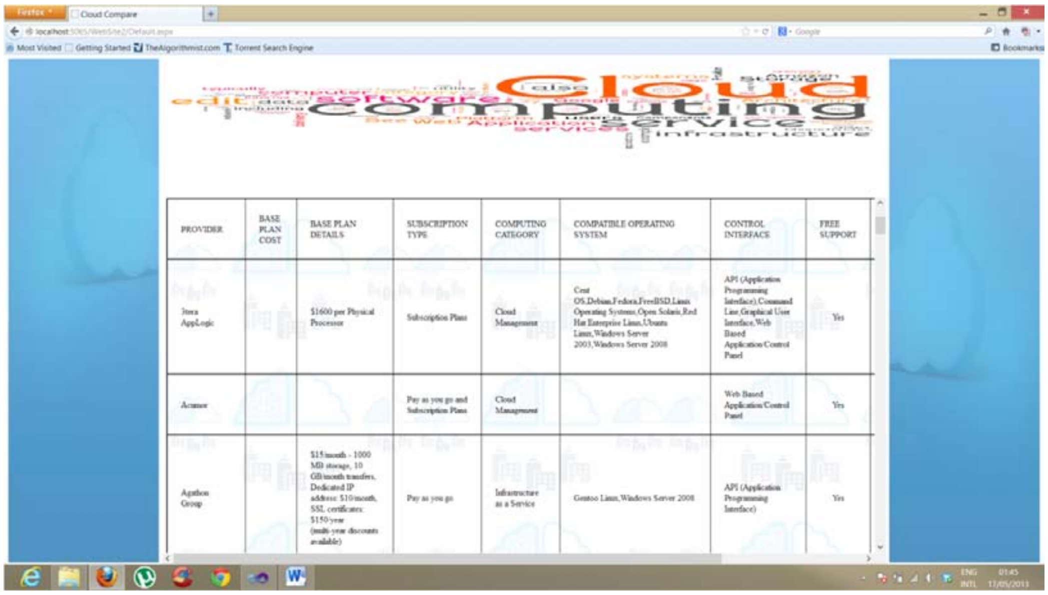


Figure 5 (b): Results of Computing Category – All, Subscription Type – All



Test case 4: The system was tested to search for services by entering a query in natural language such as Query – storage.

The other test cases are for different computing categories such as Software as a Service, Platform as a Service, Infrastructure as a Service etc. and subscription types were Pay as you go. Subscription plan for all combinations were tested. All the test cases reveal that the availability was 100%. That depicts the reliability and efficiency of the proposed system. However some filtering and customisation are required for individual requirements and priorities.

Performance Analysis

The graph in Figure 7 depicts the different resource request provided in natural language by the user, based on which

Figure 6 (a): Results of Computing Category – All, Subscription Type – Pay as you go

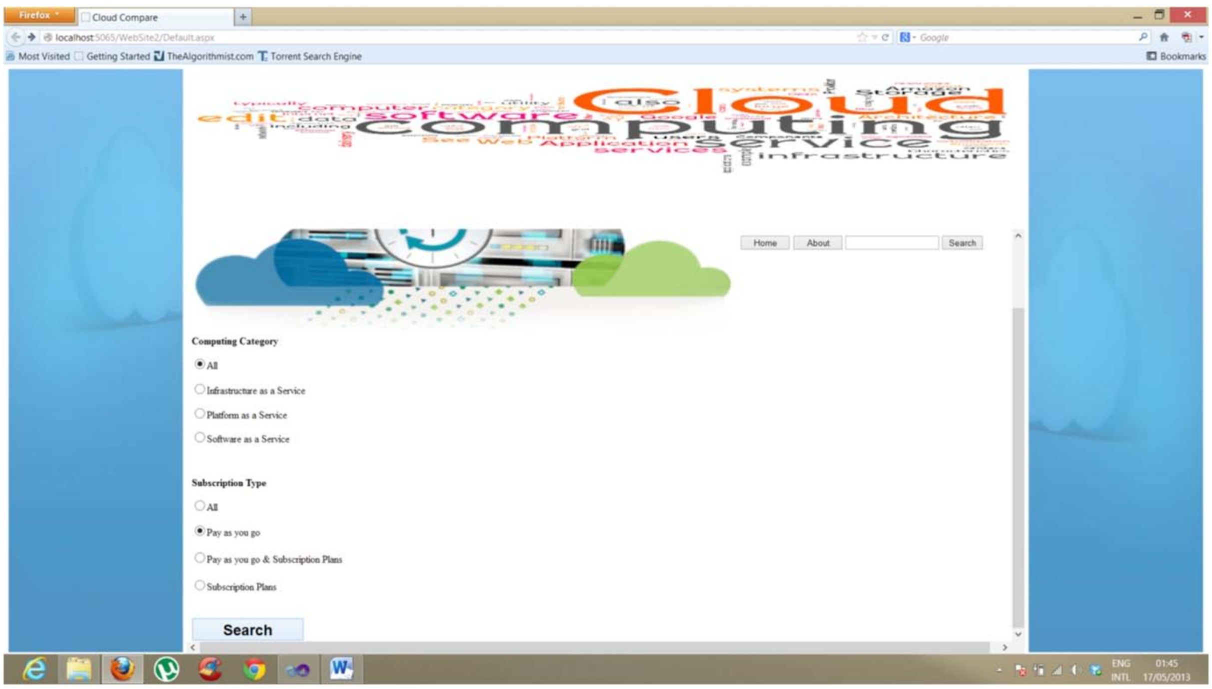
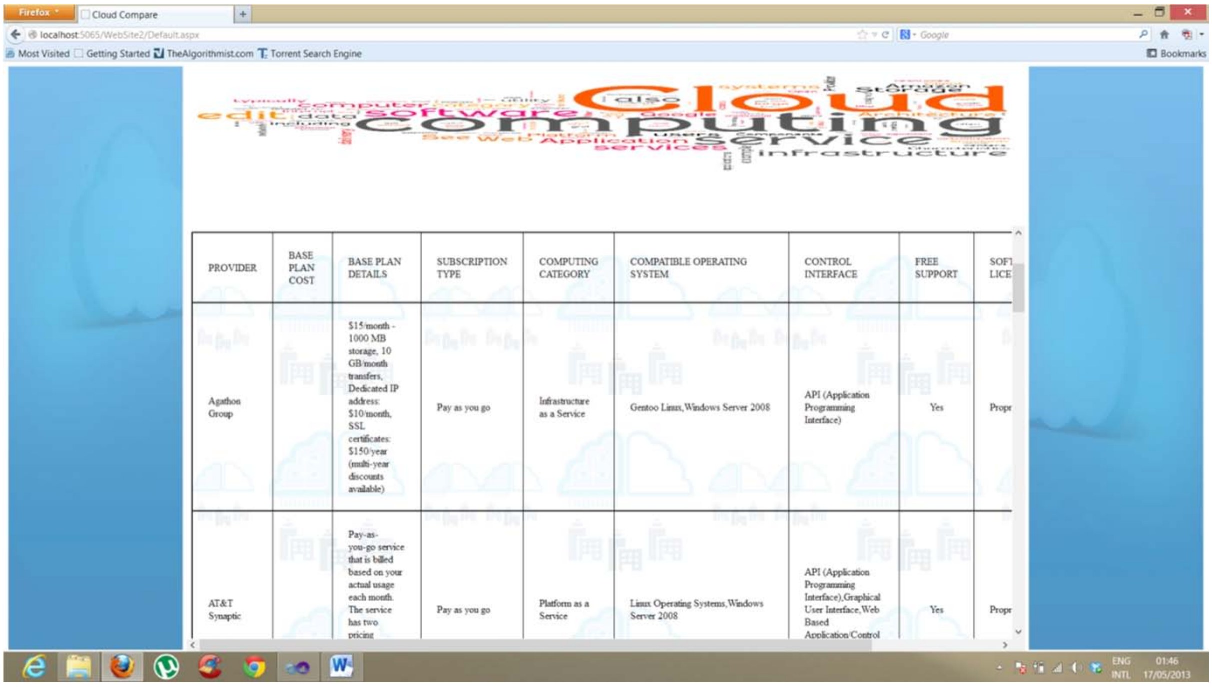


Figure 6 (b): **Results of Computing Category – All, Subscription Type – Pay as you go**



PROVIDER	BASE PLAN COST	BASE PLAN DETAILS	SUBSCRIPTION TYPE	COMPUTING CATEGORY	COMPATIBLE OPERATING SYSTEM	CONTROL INTERFACE	FREE SUPPORT	SOFT LICE
Agathon Group		$15/month - 1000 MB storage, 10 GB/month transfers, Dedicated IP address: $10/month, SSL certificates: $150/year (multi-year discounts available)	Pay as you go	Infrastructure as a Service	Gentoo Linux, Windows Server 2008	API (Application Programming Interface)	Yes	Propr
AT&T Synaptic		Pay-as-you-go service that is billed based on your actual usage each month. The service has two pricing	Pay as you go	Platform as a Service	Linux Operating Systems, Windows Server 2008	API (Application Programming Interface), Graphical User Interface, Web Based Application/Control	Yes	Propr

Figure 7: **Requests v/s Time Taken**

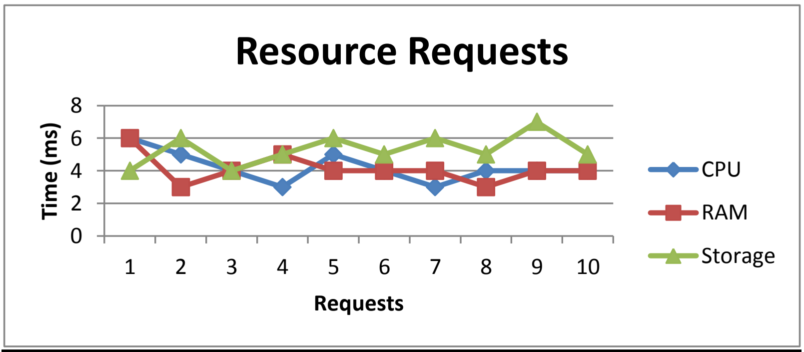
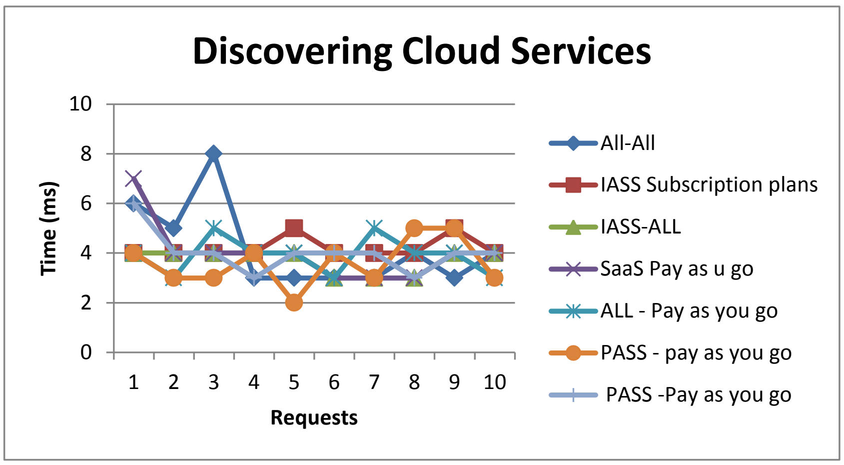


Figure 8: **Time to Discover Services**



the services are searched. Each line in the graph is plotted with results obtained based on homogeneous queries. The system was observed for the time taken to provide the services by the repository for different combinations of services and for the different resource requests in natural language. It was observed that the variations in time to receive the service are minimal, indicating that the user is not spending much time in searching for the services. It is available just at one point.

The graph in Figure 8 depicts the different criteria based on which the services are searched. Each line in the graph is plotted with results obtained based on homogeneous combination of search criteria. From the graph it can be inferred that the variation in the time taken to discover the cloud services is minimal.

Conclusions

The recent developments in the area of Cloud Computing have led to a good increase in the number of CSPs and thus the number and type of cloud services. Thus from the perspective of the end user it is a tedious and laborious task to discover for specific cloud service that is appropriate for the user's needs. The system developed helps the end user to choose a cloud service with ease, from a pool of various different CSPs. The user can make a choice by comparing between different providers based on the required QoS (cost, the features provided with the service, support for the service, etc). As the system is updated in an automated fashion the user is provided with real-time information about the available services. Thus provides a single stop for the user to query the required services.

Scope for Future Work

The developed system only provides a repository for the available services from different CSPs. Caching may be included at the main repository to improve the response time and hence the efficiency of the system. A provision can be provided where in a new CSP can register voluntarily and advertise the services provided by them. It should also be possible for a customer to register themselves and get updates on the newly discovered cloud services based on the services used in the past or of interest. Metering the usage of resources can be added to the system for accounting purpose. An efficient scheduling algorithm has to be implemented to breakup ties in the event of multiple requests for the same service.

References

Altmann, J., & Rana, O. F., (2010). GECON2010, LNCS 6296, pp. 16-33, 2010. Springer-Verlag Berlin Heidelberg.

Buyya, R., Ranjan, R., Calheiros, R. N. (2010). *Intercloud: Utility oriented federation of cloud computing environments for scaling of application services***.** Proceedings of the 10th International Conference on Algorithms and Architectures for Parallel Processing.

Cavalcante, E. (2012). *Optimizing services selection in a cloud multiplatform scenario.* Retrieved form http://www.ppgsc.ufrn.br/~evertonrsc/publications/2012-LatinCloud.pdf

Di Modica, G., Petralia, G., & Tomarchio, O. (2013). *An SLA ontology to support service discovery in future cloud markets*. 27th International Conference on Advanced Information Networking and Applications Workshops.

Garg, S. K., Versteeg, S., & Buyya, R. (2011). *SMICloud: A Framework for Comparing and Ranking Cloud Services***.** 2011 Fourth IEEE Conference on Utility and Cloud Computing, (pp. 210-218).

Gartner. (2015). Gartner Research. Retrieved from

http://www.gartner.com/technology/research/cloud-computing/report/cloud-services-bokerage.jsp

Huang, C. J. (2013). An adaptive resource management scheme in cloud computing. *Engineering Applications of Artificial Intelligence,* 26(1), 382-389.

Jain, P., Rane, D., & Patidar, S. (2012). *A novel cloud bursting brokerage and aggregation (CBBA) algorithm for multi cloud environment***.** 2nd International Conference on Advanced Computing & Communication Technologies (ACCT).

Kousiouris, G., Kyriazis, D. P., Varvarigou, T., Oliveros, E., & Mandic, P. (2011). Taxonomy and state of the art of service discovery mechanisms and their relation to the cloud computing stack cloud computing stack.

Leyli, M. & Jahani, A. (2014). Ranking approaches for cloud computing services based on QoS: A review. *ARPN Journal of Systems and Software,* March, 4(2), 50-58.

Li, B., Cao, B., Wen, K., & Li, R. (2011). Trustworthy assurance of service interoperation in cloud environment. *International Journal of Automation and Computing*, 8(3), 297- 308.

Manvi, S. S., & Shyam, G. K. (2014). Resource management for infrastructure as a Service (IAAS) in cloud computing: A survey. *Journal of Network and Computer Applications*, 41, 424-440.

Mattess, M. (2011). *Cloud bursting: Managing peak loads by leasing public cloud services*. Cloud Computing: Methodology, Systems, and Applications, CRC Press, USA.

Maurer, M., Brandic, I., & Sakellariou, R. (2013). Adaptive resource configuration for cloudinfrastructure management. *Future Generation Computer Systems*, 29(2), 472-487.

Mındruta, C., & Fortis, T. F. (2013). *A semantic registry for cloud services*. 27th International Conference on Advanced Information Networking and Applications Workshops.

Miranda, Z. (2012). *An ontology based system for cloud infrastructure services discovery*. Retrieved from http://arxiv.org/ftp/arxiv/papers/1212/1212.0156.pdf

Modica, G. D., Petralia, G., & Tomarchio, O. (2012). *A business ontology to enable semantic matchmaking in open cloud markets.* International Conference on Semantics, Knowledge and Grid.

Nair, S. K. (2010). Towards secure cloud bursting, brokerage and aggregation. *IEEE 8th European Conference on Web Services (ECOWS).*

Qi, Z., Cheng, L., & Boutaba, R. (2010). Cloud computing: State-of-the-art and research challenges, *The Brazilian Computer Society***.**

Qu, L., Wang, Y., & Orgun, M. A. (2013). *Cloud service selection based on aggregation of user feedback and qualitative performance assessment*. Proceedings of 2013 IEEE International Conference on Services and Computing, pp. 52-159.

Ranjan, R., Buyya, R., & Benatallah, B. (2012). Special section: software architectures and application development environments for Cloud computing. *Software- Practice and Experience*, 42, 391-394.

Taekgyeong, H., & Kwang, M. S. (2010). An Ontology enhanced Cloud Service Discovery. Retrieved from www.iaeng.org/publications/IMECS2010_pp644-649.pdf

Toosi, A. N., Calheiros, R. N., & Buyya, R. (2014). Interconnected cloud computing environments: Challenges, taxonomy and survey. *Journal ACM Computing Surveys (CSUR)*, July, 47(1).

Tserpes, K., Aisopos, F., Kyriazis, D., & Varvarigou, T. (2010). Service selection decision support in the internet of services. *Economics of Grids Clouds, Systems and Services*, 6296, 16-33. Springer Berlin Heidelberg.

Vouk, M. A. (2008). Cloud computing- Issues, research and implementations. *Journal of Computing and Information Technology*, 16(4), 235-246.

Zhang, J., He, L., Feiyi, H. & Bin, L. (2013). Service discovery architecture applied in cloud computing environments. *Applied Mechanics and Materials*, 241-244, 3177-3183.

Zhang, M., Ranjan, R., Haller, A., Georgakopoulos, D., Menzel, M., & Nepal, S. (2012). An ontology-based system for Cloud infrastructure services' discovery. In *Collaborative Computing: Networking, Applications and Worksharing (CollaborateCom), 2012 8th International Conference on* (pp. 524-530).