A Survey On
Service Matching For Multi-Tenants In Cloud Computing

[1] Ms. Dhara Virani, [2] Ms. Khushbu Virani
[1]P.G. Student, Marwadi Education Foundation Group of Institutes, Rajkot, Gujarat.
[2]P.G. Student, Marwadi Education Foundation Group of Institutes, Rajkot, Gujarat.

Abstract

Cloud computing is a highly Research area in the technical I.T field and economic world, and many of the software industry have entered the development of cloud services. It is the Preeminent on-demand service system along with a "Pay-as-you-go" Policy. Several companies define the profit and probability of incorporating such cloud computing service in big business. However, with the amount of cloud computing services growing rapidly necessitate for a taxonomy framework rises. In Multi-tenant networking, with which multiple customers (tenant) networks are virtualized over a single collective physical infrastructure. It is a way to self-motivated creation of a tenant by integrating existing cloud-based services. Dynamic provisioning in the cloud requires an integrated solution across the technology stack (software, platform and infrastructure) combining functional, non-functional and resource allocation requirements. Research works in the area of web service matching. The taxonomy offers a common terminology and baseline information for easy communication. The Architectural features of multi-tenancy and classify them according to the requirements of end-users, enterprise that use the cloud as a platform, and tenant providers themselves. Service is matched with existing tenants and according to the requirement of end-users. Matching techniques such as string-based, schema based, semantic web service based, constraint-based, linguistic, graph-based and taxonomy-based. Clients spend extreme amounts of time and energy searching through a list of available services.

1. Introduction

The goal of Cloud Computing is to offer on-demand computing services with high reliability, scalability, and availability in distributed environments. Many big players of the software industry, such as Microsoft, as well as other Internet technology including Google and Amazon, are joining the development of cloud services [2].

Cloud Computing is a model for on-demand network access to a mutually cluster of configurable computing resources that can be rapidly provisioned and released with minimal Effort. In its description of essential cloud characteristics, the US National Institute of Standards and Technology (NIST) Captures sound information, what it means to provide IT services [5].

- On-demand self-service.
  Users can maintain services without interaction with the service provider. Just pay for demanded services and use it.
- Ubiquitous network access.
  Cloud services are accessed via the Internet, using standard mechanisms and protocols.
- Resource pooling.
  Computing resources used to provide the cloud service are realized using a uniform infrastructure that's shared between all service users.
- Rapid elasticity
  Resources can be scaled up and down rapidly and elastically.
- Measured service.
  Resource/service usage is constantly metered, supporting optimization of resource usage, usage reporting to the customer, and pay-as-you-go business models.

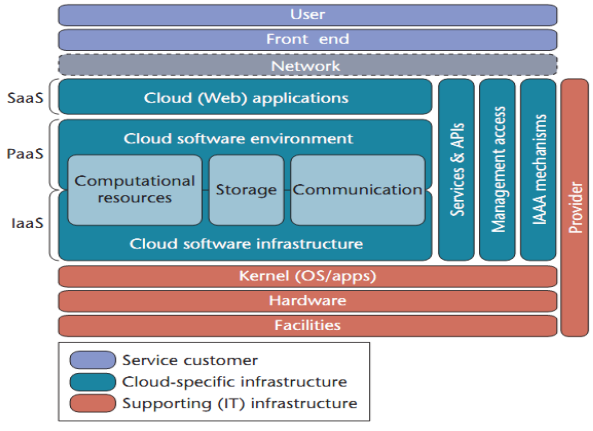

Figure.1. The cloud architecture [5].

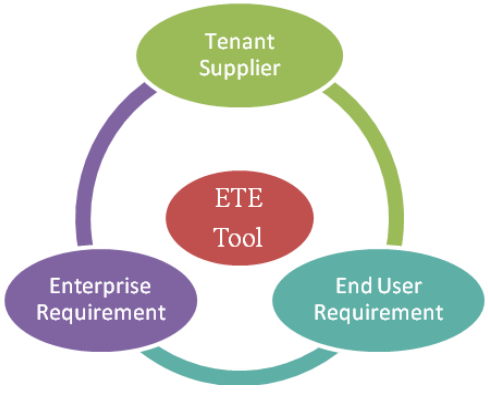

Figure 2. Multi-tenant architect model

Cloud computing constructs on three technologies: Software as a service (SaaS), platform as a service (PaaS) and infrastructure as a service (IaaS). SaaS contributions are typically implemented as Web applications, while PaaS contributions to development and runtime environments for Web applications and services. Infrastructure as a service (IaaS) supports an associated services and APIs, such as the management access for customers, using Web application/service technologies.Fig1. Represent basic architecture of Cloud Computing System. The rest of our paper is organized as follows: Sect. II introduces the approaches and techniques. Section III introduces the taxonomy based service matching. Sect. IV describes analysis of services and complexity tables. In Sect. V concludes the paper.

1.1. Multi-Tenant

Multi-tenant systems allow multiple clients to work on the identical instance of an application; it can be software or database tables or platforms. Different types of job can be run on the same nodes with sufficient resources and without interfering with each other. This encourage virtual tenant to be introduced as service containers for improving resource utilization and time complexity of the service matching system. Multi-tenant system considers functional, non-functional and resource allocation requirements collectively. Multi-tenant system works based on Enterprise Requirement-Tenant supplier-End User Requirement (ETE). This three layered architecture get high performance, provide dynamic services, and support efficient service architecture, secure data management. Figure 2 represent Multi-tenant architect model.

Cloud based systems are designed in a manner that requires a client to select and instantiate service tenants from a catalogue prepared by the provider. As a result clients spend excessive amounts of time and energy searching through a list of available services, resulting in a sub-optimal selection [1]. The tenant selection of our system guarantees fast and efficient tenant selection by matching the client's requirements with the tenant supplier functionalities.

2. Approach And Techniques For Service Matching

2.1. Approach

L. Ramachandran et al. [1] have present Users Requirement is represented by Tenant Requirement model (TRM). It constitutes behaviour, Constraint and state information of a tenant. Tenant state defines like Create, Active or Paused state. Constraint listed by a tenant in its service level agreement (SLA) or managing resources allocated across different tenants. Tenant is provided by Tenant Provider Model (TPM). It considers different Tenant factors value like time to deploy (t), cost(c), performance (p) and extensibility (e), resource requirements (r). Tenant Selection is also considered Degree of Match like that is, exact match > plug-in > subsume > disjoint. And that provides the proper tenant.

Throughout the identification of platform and infrastructural requirements for a new client, the provider uses the maximum of the values from tenants that have provided functionality for the new client. Since the new client is unaware of the memory or

storage requirements. So consider different matching technique and reduce the time complexity [1].

2.2 Techniques

The cloud stack provides a main three types of services. SaaS, PaaS and IaaS. Cloud provides many services related to the stack and many characteristics are provided. So we need to compare user requirement and tenant functionality. Different types of matching techniques are available like string-based, schema based, semantic web service based, constraint-based, linguistic, graph-based and taxonomy-based.

2.2.1 String based service matching

Word based lossless data compression algorithm includes a static dictionary of the most frequent words in texts and this to be considered as tenant functionality. The words are grouped by their number of the letters and stored in the dictionary, starts comparison from the last character of the pattern. If they are same then it compares the first character of the pattern with the character in the same location of the text [7]. Berry Ravindran algorithm has better performance than Word Match about 33%. Raita algorithm has better than Word Match about 27 % and Boyer-Moore algorithm has worse than Word Match about 1 % [7].

2.2.2 Schema based service matching

In the schema based matching techniques, multiple tenants schema is combined into the one database for reduce total cost of ownership. There is to map multiple single-tenant logical schemas in the application to one multi-tenant physical schema in the database. In this Chunk-folding approach is used. Stefan et al.[6] develop algorithms that take the logical schemas of tenants, the distribution of data within those schemas, and the associated application queries, and create a suitable Chunk Table Layout. Because these factors can vary over time, it should be possible to migrate the data from one representation to another.

2.2.3 Semantic web service based service matching

Semantic Web services combine Semantic Web and Web services technology. Service requests and advertisements are interpreted by concepts of associated ontology, and the matchmaking is based on users requested and covered parameters. Index is the service representations to prune the search space, minimizing the number of comparisons required to locate the matching services. Give the ranking for matched services based on ranking function, fetches the top-k matches progressively, and reducing the search engine's response time. In the service descriptions contain preconditions and QoS parameters [8].

2.2.4 Graph based service matching

In this technique Advertisements and search Queries are expressed in terms of OWL-S descriptions. In this technique matching is done by Bipartite graph G= (V, E), one disjoint set is created by Advertisement and other is Query. Where v is matched if two nodes has an edge. This also considered Degree of Matching. Three techniques are used Bipartite Matching algorithm, Greedy matchmaking algorithm, A Brute-Force matching algorithm. This technique, improves the efficiency through reducing the time which is required for construction of bipartite graphs. The search time of bipartite matching is higher than that of the Greedy algorithm but lower than that of the Brute force algorithm.

2.2.5 Taxonomy based service matching

Taxonomy design in tree based structure. Root of the tree is all cloud services. Design of taxonomy includes many services like Security, Interoperability, Data management storage and processing, SLA, Metering and billing, Virtualization management, User-centric Data privacy, migration. Users requirement are considered in that form and service provider provided different services that can match with users' requirement. Numbers of factors to be consider in all types of service. Compare all the parameters and provide a resource. Improved interoperability and clear standards make it easier to develop new cloud services [2].

3. Taxonomy Based Service Matching

There are several open issues with cloud such as security, availability, scalability, interoperability, service level agreement, data migration, data governance, trusty pyramid, user centric privacy, transparency, political and legal issues, business service management etc. In the ETE (Enterprise Requirement-Tenant Supplier- End User Requirement) Architecture, different types of services considered at user and tenant provider side. After consideration match the user requirements with Tenant functionality.

3.1 Enterprise Requirement

According to Robbins [9], an enterprise should always make sure that they know what measurements exist to show you are actually receiving the services and should pay careful attention to the issues like

service levels, privacy matters, compliances, data ownership, and data mobility. Many factors are requiring to deployment of virtual tenant likes security, cloudonomics, data governance over business process management and data migration.

3.2 Tenant Supplier

This Approach describes the requirements of provider service delivery model requirements, service-centric issues, interoperability, data management, storage and processing, fault-tolerance, virtualization management, and load balancing.

3.3 End-User Requirement

Users' requirements are the third key factor to the adoption of any cloud system within an enterprise [3]. This Approach describes the billing and metering requirements, user centric privacy requirements, service level agreements, adaptability and learning requirements, and user experience requirements.

4. Analysis

Table 1
Complexity Analysis

Algorithm	Technique	Time Complexity
Brute Force Algorithm	String Matching	O(mn)
Boyre-Moore Algorithm	String Matching	O(m+n)
Hungarian algorithm-Bipartite graph matching (Dynamic)	Graph Matching	$O(\|v\|^3)$ |V|= no. of vertices in graph
Chunk Folding	Schema Matching	Change Vary over time
Using Taxonomy Tenant selector Matching using dynamic resource allocation	Taxonomy Matching	$O(n^2)$
Tenant selector matching using Degree of match	Taxonomy Matching	$O(n^4)$

Table 2
Analysis of services Related to tenant

Services Requirement	Interoperability	Data Storage	SLA	Virtualization Management	Payment system
Provider Requirement	Supports scaling of application across multiple vendors, Open API	Depend on amount of the queried data and easily extend internal capabilities	Provide encryption and secure identification	Maximum profit using server, client, storage Virtualization	Can build up the trust with End user and provide billing services
Enterprise Requirement	Allowed to be portability between tenant	Elastic storage EC2 service provided	Can claim the charge for service under condition	Can maximum profits	Can build up the trust With tenants

End-user Requirement	User request the reliable and portable data and API	It can be benefited with pay-per- use service	Agreement is important. if Service is fail user can claim refund	It can be cheaper for User	Customer only pays for resources consumed

5. Conclusion

Platform as a service (PaaS) and Software as a service (SaaS) are current software application development and delivery models that an reduce assets expenses, and recover global economy. Internet-based, shared computing platforms managed software assets on demand and altogether avoid the costs and complexity associated with the purchase, installation, configuration, and ongoing maintenance of an on-premises data centre and dedicated hardware, software, and go together with administrative staff.

In this paper, find the crucial issue of dynamic, service-tenant in cloud-based systems, and it contains several key features specifically suited to cloud-based systems like integrated functional and non-functional requirement matching at SaaS, PaaS and IaaS levels; Dynamic resource allocation using state information; and elimination of redundant tenant functionalities in order to prune the search space. And also cost and time are reduced to matching a service and dynamic resource allocation. Match making Algorithm choose an appropriate tenant according to users requirement however, it find based on the degree of match, states, constraints and behaviour of tenant.

6. References

[1] L. Ramachandran, N. C. Narendra, K.Ponnalagu, Dynamic provisioning in Multi-tenant Service clouds", Service Oriented Computing and Applications, Vol. 6, 2012.

[2] C. N. Höfer and G. Karagiannis, "Cloud computing Services: taxonomy and comparison", Journal of Internet Services and application, vol.2, Number 2, 2011, 81-94.

[3] B. Rimal, A. Jukan, D. Katsaros, Y. Goeleven, "Architectural Requirements for Cloud Computing Systems: An Enterprise Cloud Approach", Journal of Grid Computing, Vol. 9, Number 1, 2011, 3-26.

[4] U. Bellur, R. Kulkarni, "Improved Matchmaking Algorithm for Semantic Web Services Based on Bipartite Graph Matching",IEEE International Conference on Web Services, 2007, pp. 86 – 93.

[5] Grobauer, B.walloschek, T.stocker, "Understanding cloud computing Vulnerabilities", Security and privacy IEEE, vol. 9, 2011, pp.50-57.

[6] S.Aulbach, T. Grust, D. Jacobs, A. Kemper, J.Rittinger, "Multi-tenant Databases for Software as a service: schema mapping techniques", SIGMOD, 9–12, 2008.

[7] A CARUS, H. Nusret BULUS, A.MESUT, "WORDMATCH: WORD BASED STRING MATCHING OVER COMPRESSED TEXTS", INTERNATIONAL SCIENTIFIC CONFERENCE, Nov. 2007, vol. I, 357-360.

[8] D. Skoutas, A. Sheth, "Efficient Semantic Web Service Discovery in Centralized and P2P Environments", ISWC 2008, pp. 583–598.

[9] David, R.: Cloud computing explained. Available online at http://tinyurl.com/qexwau (2009). Retrieved on 1 Sept 2009.

[10] Deutsch A, Vianu V (2008) Wave: automatic verification of data-driven web services. IEEE Data Eng Bull 31(3):35–39

[11] K. Nan, J. Yu, H. Su, S. Guo, Hui Zhang and Ke Xu, "Towards structural Web Services matching based on Kernel methods" Frontiers of Computer Science in China, Volume 1, Number 4,2007, 450-458