Distributed Ontology Cloud Storage System

Mohamed Helmy Khafagy
Computer Science Department
Fayoum University
Egypt
mhk00@Fayoum.edu.eg

Haytham Tawfeek Al Feel
Information System Department
Fayoum University
Egypt
htf00@Fayoum.edu.eg

***Abstract*—There are dramatically increasing interests from both academic and industry in the trend of cloud computing. Cloud computing depends on the idea of computing on demand that provide, support and delivery of computing services with stable and large data space. Our research concerns with improving the searching process in the cloud storage via avoiding the bottleneck in central ontology cloud storage system since all data chunks in the cloud must be indexed by a master ontology server.**

The contribution of this paper proposes new cloud storage architecture based distributed ontology as one of the main semantic technologies. This architecture provides better scalability, fault tolerance and enhanced performance for searching in the cloud storage avoiding the central bottleneck

Keywords— Cloud- Ontology -P2P-Performance-Storage File System-Semantic Web

I. INTRODUCTION

Cloud computing is a new business paradigm that is considered a hotspot topic during the last few years, till now. Cloud computing has many definitions, but the clearest definition is the one described by the National Institute of standards and Technology (NIST) which defines it as " A model for enabling ubiquitous, convenient, on demand network access to a shared pool of configurable computing resources including (networks, servers, storage, applications and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction" [1].

User who will use the cloud computing will access different services via web portals and will able to use huge storage utilizes by minimum cost compared to the buying of machines to do this work. There are different cloud systems available such as Elastic Cloud of Amazon [2], Google File system (GFS) [3] and Blue Cloud of IBM [4], but all these systems do not have the ability of searching inside the file content which was provided by the OCSS [5]. OCSS provides searching inside the file content depending on ontologies instead of metadata, unfortunately, OCSS still have a problem of bottleneck due to the number of requests to the central server such as the GFS. Semantic Web is introduced by Tim Berners Lee as architecture for interconnected communities and vocabularies [6] [7]. From this point of view ontologies will be used as interconnected vocabularies between different chunks and servers.

This paper proposes a new Cloud Storage Architecture that based on Distributed ontology and depends on semantically enriched. Ontologies and linked data which are the main components of the Semantic Web technologies that enable machine and people work in collaboration, facilitating machines to communicate with each other.

The rest of this paper is organized as follows: Section 2 contains an overview of related work Google File System and Ontology Cloud Storage System OCSS while Section 3 presents Distributed Ontology Cloud Storage System while Section 3 describes OCSS. On the other hand, we found in Section 4 show the benchmark used for evaluate our technique. Section 5 presents the test of this work and the experimental results showing a comparison between GFS, OCSS and the DOCSS Distributed Ontology Cloud Storage System. Finally, Section 6 concludes this paper and suggests the future work.

II. . RELATED WORKS

In this section, we shall introduce some related work regarding cloud storage system.

A. Google File System

The first Cloud computing term Presented by Google's in [8]. As shown in figure 1 Google cloud File system has four systems, which independent of and linked to each other. They are Google File System for distributed file storage, Map Reduce program model for parallel applications [9], Chubby for distributed lock mechanism [10] and Bigtable for Google large-scale distributed database [11]. Google File System [3] is considered a distributed file system which enables the computer to access different files allocated on other servers in the same network. Files in Google File System are divided into multiple chunks with fixed size. Each chunk is replicated in different chunk servers. There is a master server which contains metadata work as an index which contains the location of each chunk and its copies. When a client needs to access to a chunk, this client will request the file from the master server. The master server will retrieve the file name associated with the location of that chunk. The client will go to its location at

the chunk server. In the updating process, permissions for any modification to any chunk are the main responsibilities of the master server. The client will request a modification to a specific file from the master server. The master server will guide the client to the primary chunk server that have this file and will prevent any changes to the chunks associated to this file in another chunk server. When the modification transaction updated the versions of the file and chunks then it will be replicated in the replica servers. The master server is playing the coordinator role of all processes inside this architecture which can cause a bottleneck to the master server which can cause a single point of failure [12]

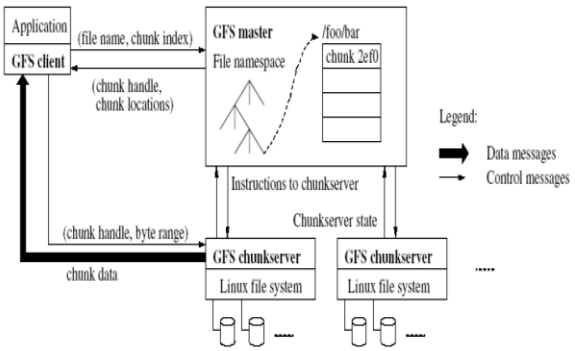

Figure 1. Architecture of Google File System [3]

B. Ontology Cloud Storage System OCSS

Figure 2 shows OCSS architecture that replaces the master metadata index in the cloud by an ontology which facilitates the searching inside the file content and increases the speed of search.

When user performs a Read Operation Client sends a request to the Ontology. Containing the logic identifier or the keywords related to the file content then the location of the file, which contains the matching keyword, will be determined with its Replicas. The OCSS will select the chunk server which contains the latest version number, if there is more than one candidate, the OCSS will select the nearest node by comparing the IP address of the client and the data server, then returns the best address to the client. When the client gets the best address, it will then send its request to the chunk server which contains the data block. Now the chunk server acts as a data provider as the traditional cloud storage platform does.

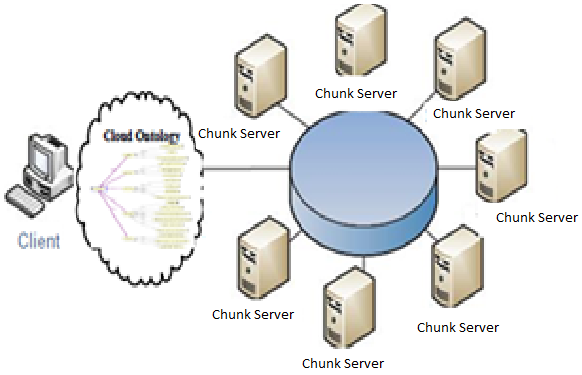

Figure 2. Cloud Storage Based on ontology

When user performs a Write Operation Client sends a request for a data block with logic identifier and new Keywords to the Ontology. Then the location of the file, which contains the matching keyword, will be determined with its Replicas. The Ontology updated with new information and new Keywords if there are existed. The OCSS will select the chunk server which contains the latest version number, if there is more than one candidate, the OCSS will select the nearest node by comparing the IP address of the client and the data server, then return the best address to the client after that The OCSS will lock the selected chunk server and its replicas. Then the write process will begin on the chunk server and its replicas. After the updating of each replica, the version number also will be updated and the locked are released.

III. DISTRBUTED ONTOLOGY CLOUD STORAGE SYSTEM

A. Ontology

Ontology is considered one of the main semantic technologies that are defined as "a formal explicit specification of a shared conceptualization" [13] [14] [15].

Between lines, we can understand that the ontology is a representation for knowledge by a way that the machine can read and discover meaning and relations between different resources inside the ontology. One of the defects of cloud computing is; It's weakness in understanding of data and searching within the file contents. This weakness could be fulfilled by the using of ontologies, because ontologies are used as common, sharable, usable and understandable structured information in different domains. [16]

There are three approaches for construction of ontologies they are, single, multiple and hybrid ontology approaches. The single ontology approach depends on the idea of a global ontology for all resources, but this approach face the lack of integration while the multiple approaches depends on the idea of private or local ontology for each resource, but also this approach alone has drawbacks especially in the construction process [16]. In our work, we will use the hybrid

approach to take advantage of the global vocabulary and the localization of ontologies due to each resource; which means that the ontology could be integrated between different resources and in the same time it is localized. Ontologies are encoded here using the Web Ontology Language (OWL) as the representation language because of its expressiveness, in addition to its capability to be extended in the future [17]. Protégé is used as an ontology editor that facilitates the work with classes, properties and instances. On the other hand, ontologies are evaluated by generic criteria such as clarity, simplicity, interoperability, expressivity, compatibility, versioning, consistency and reusability. In addition to the evaluation by competency questions that discover the ability of ontologies in providing user requirements.

The methodology used for building ontologies in this work depends on the work of Be **KACTUS** [18], which use the Top-Down strategy for identifying concepts and is application dependent. In addition to that ontology can be developed by reusing others and can be integrated with other ontologies in other applications [19] which are needed in our distributed Architecture Ontology Cloud Storage System.

B. Linked data

Linked data are a way of representing the relationship between interrelated datasets on the web to facilitate access management and update of that data [20].

Wikipedia defined linked data as "a method of publishing structured data so that it can be interlinked and become more useful. It builds upon standard Web technologies such as HTTP and URIs [21]. Inside each server, there is ontology containing two parts; the first part is a private part that contains all information related to the files in the server; while the public part contains all information about other files that are places in other servers. The ontology is updated regularly depends on the linked data and software designed for that purpose depends on a similar idea of extraction of data that are used in DBpedia [22] [23] [24].

C. Arctecture

As we show in figure 3 DOCSS build two ontologies in each chunk, Local ontology and Global ontology, Local ontology contain information about the files and its content found in this chunk but global ontology have information about all files and its content in the whole cloud.

D. Read Operation

Client sends a request to the nearest Ontology Containing the name and the keywords related to the file content.

The location of the file which contains the matching keyword would be determined with its Replicas if this file found in this local ontology

If local ontology can't find the file then send the search request to global ontology and determine the chunk server which contains the latest version number, if there is more than one copy, the DOCSS will select the nearest node by comparing the IP address of the client and the data server, and then returns the best address to the client.

When the client gets the best address, it will then send its request to the address of the chunk server which contains the data block. Now the chunk server acts as a data provider as the traditional cloud storage platform does.

E. Write Operation

Client sends a request to the nearest Ontology Containing the name and the keywords related to the file content.

The location of the file which contains the matching keyword would be determined with its Replicas if this file found in this local ontology then make the change in the file and update local ontology and global ontology and send update to all global ontology in the cloud.

If local ontology can't find the file then send the search request to global ontology and determine the chunk server which contains the latest version number, if there is more than one candidate, the DOCSS will select the nearest node by comparing the IP address of the client and the data server, and then returns the best address to the client.

Client sends update request to the chunk server contain the latest version of the file. The DOCSS will lock the selected chunk server and its replicas. Then make the change in the file and update local ontology and global ontology and send update to all global ontology in the cloud

After the updating of each replica, the version number also will be updated and the locked are released.

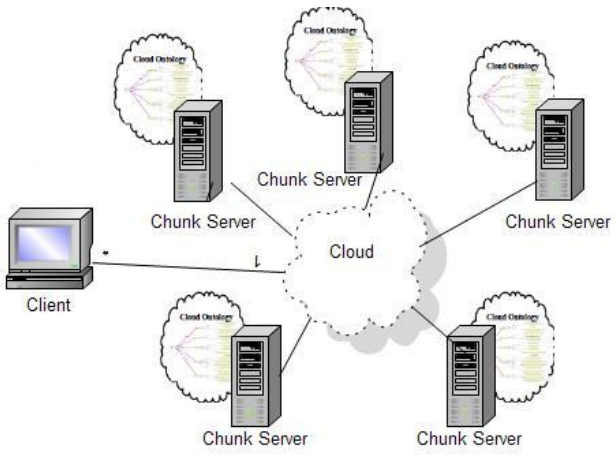

Figure 3. Distributed Cloud Storage Based on ontology

F. Replication

The main problem facing cloud computing is the increasing of the availability of storage system, which will be solved here in the DOCSS by using the agent-scheduling routine replication technique [25].

IV. BENCHMARK

A benchmark developed here using C#.NET to simulate and test the GFS, OCSS,P2P and DOCSS including the number of chunk servers, clients, operation types (read/write) and number of operations are entered as parameters to the system. Also, we used OWL, RDF, and XML for building Ontology.

V. EXPERIMENTS RESULTS

We implemented all three architectures using our model. Our simulation was fed with varying number of clients in order to test both response time and throughput.

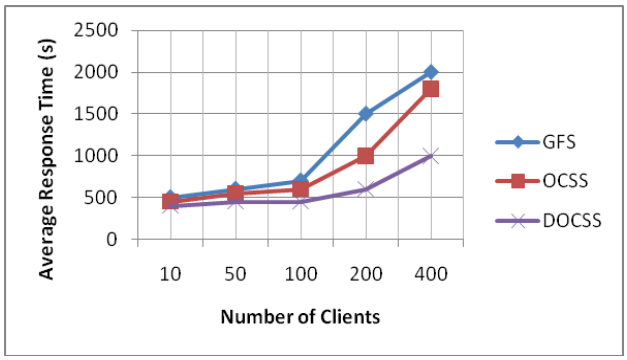

Figure 4. Figure 4: Effect of average response time during write operation

In figure 4, we demonstrate that with increasing number of clients, DOCSS shows better response time during write mode. Google's result showed highest response time due to the bottle-neck resulted from the centralized architecture, compared with OCSS and DOCSS.

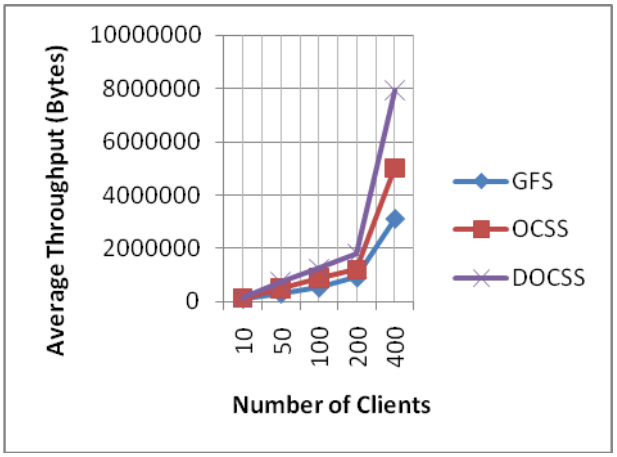

Figure 5. Figure 5: Effect of average throughput write operation

Figure 5 DOCSS and OCSS show higher number of bytes written due to the agent scheduled replication procedure used to maintain consistencies among replicas, compared with Google.

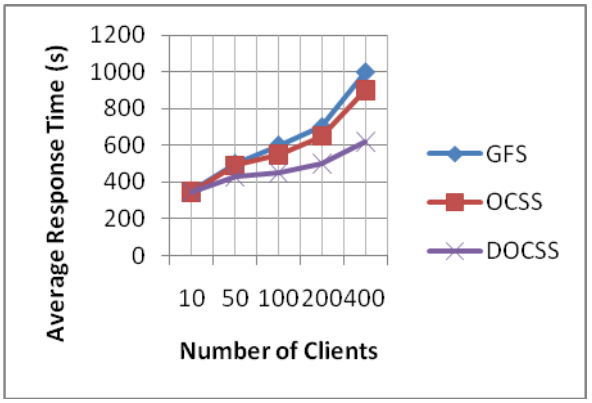

Figure 6. Figure 6: Effect of average response time during read operation

In figure 6, we demonstrate that with increasing number of clients, DOCSS shows lowest response time during read mode compared with OCSS and Google as DOCSS was free from the bottleneck from which Google and OCSS suffered.

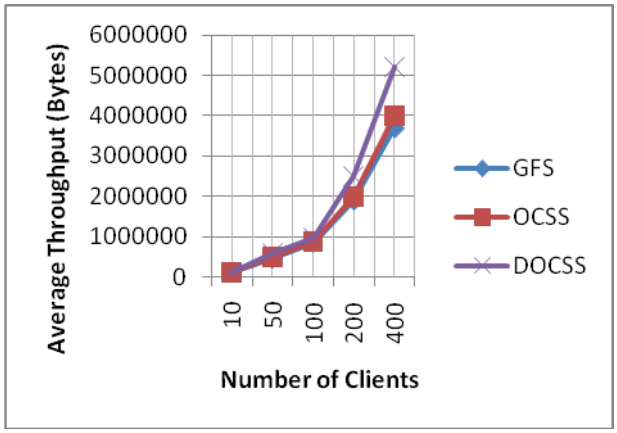

Figure 7. Figure 7: Effect of average throughput during read operation

Figure 7 illustrates that all three architectures show similar results for the number of bytes read.

VI. CONCLUSION

The experimental results confirm that DOCSS shows better results compared to both Google & OCSS architectures in terms of response time, due to the fact that DOCSS was a defect from the bottle neck resulting from the centralized architecture of Google and OCSS. Our test environment was composed of 5 servers accommodating 50 files distributed randomly and the numbers of clients were entered as a parameter ranging as 10, 50, 100, 200 & 400 where all clients each accessed 10 files applying both read/write operations. We show that as the number of clients increase, the response time increases for both Google and OCSS compared to DOCSS, whereas Google and DOCSS shows better throughput for the increasing number of clients compared to OCSS.

REFERENCES

[1] Final Version of NIST Cloud Computing Definition Published. 25 October 2011. http://www.nist.gov/itl/csd/colud-102511.cfm.

[2] Amazon Elastic compute cloud (URL) :http: //aws.amazon.com/ec2/, Access on Jan 2011.

[3] S.G hemawat, H.Gobioff, and S'Leung. The Google file system, In proceedings of the 19th ACM podium on operating systems principles, pages 29-43,2003

[4] Boss G, Malladi P, Quan D, Legregni L, Hall H. Cloud computing. IBM White Paper, 2007.

[5] Haytham Tawfeek al Feel, Mohamed Helmy Khafagy "OCSS: Ontology Cloud Storage System ", IEEE First International Symposium on Network Cloud Computing and Applications Toulouse, France November 2011.

[6] T. Berners-Lee, J. Hendler, and O. Lassila, "The Semantic Web," in Scientific American, May 2001, pp. 29-37.

[7] T. Berners-Lee, "Artificial Intelligence & the Semantic Web, " World Wide Web Consortium, 2006. [Online]. Available: http://www.w3.org/2006/Talks/0718-aaai-tbl/overview.html#(1). [Accessed: March. 12, 2012].

[8] Barroso LA, Dean J, Hölzle U. Web search for a planet: The Google cluster architecture. IEEE Micro, 2003,23(2): PP 22-28.

[9] Dean J, Ghemawat S. MapReduce: Simplified data processing on large clusters. In: Proc. of the 6th Symp. On Operating System Design and Implementation. Berkeley: USENIX Association, 2004. PP 137-150.

[10] Francesco Maria Aymerich, Gianni Fenu, Simone Surcis. "An Approach to a Cloud Computing Network", ICADIWT, 2008 PP 113-118

[11] Chang F, Dean J, Ghemawat S, Hsieh WC, Wallach DA, Burrows M, Chandra T, Fikes A, Gruber RE. Bigtable: A distributed storage system for structured data. In: Proc. of the 7th USENIX Symp. On Operating Systems Design and Implementation. Berkeley: USENIX Association, 2006. PP 205-218.

[12] Fesehaye, Debessay, Malik, Rahul, Nahrstedt and Klara. A Scalable Distributed File System for cloud computing ScienceCloud2010

[13] T.R Gruber" A Translation Approach To A Portable Ontology Specification, " Knowledge Sharing", "International Journal of Human–Computer Studies", vol43, pp907-928,1995

[14] T.R Gruber " Towards Principles for the design of ontologies used for knowledge sharing","International Journal of Human-Computer Studies", vol 43,pp 907-928, 1995.

[15] W.N. Borst, "Construction of Engineering Ontology" Ph.D. Dissertation, Center for Telematica and Information Technology, University Of Tweenty, 1997.

[16] R. Subhashini and J.Akilandeswari, "A Survey On Ontology Construction Methodologies"," International Journal of Enterprise Computing and Business Systems"

http:// www.ijecbs.com Vol.1, Issue 1, January 2011.

[17] http://www.w3.org/TR/owl-features/ ,2012.

[18] The KACTUS Booklet Version 1.0. Esprit Project 8145. September 1996. http://www.swi.psy.ura.nl/prjects/newKACtUES/Report.html.

[19] Fernandes Lopez, Overview Of Methodologies for Building Ontologies. Proceedings of the IJCAI99 Workshop on Ontologes and Problem Solving Methods (KRR5) Stockholm, Sweden/ august 2.1999.

[20] http://www.w3.org/standards/semanticweb/data,2012.

[21] http://en.wikipedia.org/wiki/Linked_data,2012.

[22] Mohamed morsy, Jens lehmann. LOD2. Creating knowledge on the Interlinked Data, Report, 1/9/2010

[23] Dimitris Kontokostas, Charalampos Bratsas, S[illegible]ren Auer, Sebastian Hellmann, Ioannis Antoniou, George Metakides: Internationalization of Linked Data. The case of the Greek DBpedia edition. Journal of Web Semantics: Science, Services and Agents on the World Wide Web (JWS), Elsevier,2012,

[24] http://www.w3.org/TR/rdf-sparql-query/ ,2012.

[25] E. Sarhan, A. Ghalwash, M.Khafagy, "Agent Based Replication for Scaling Back-End Databases of Dynamic Content Web Sites", 12th WSEAS International Conference on OMPUTERS, Heraklion, Greece, 2008. PP 867-862