Critical Analysis of Distributed File Systems on Mobile Platforms

Madhavi Vaidya Dr. Shrinivas Deshpande

Abstract **-The distributed file system is the central component for storing data infrastructure. Applications that process large volumes of data require a backend infrastructure for storing data. In this paper, we develop a comparative classification for describing distributed file system architectures and use this categorization to survey existing distributed file system implementations on mobile platforms. We used the classification and the survey results to identify architectural approaches on the basis of the ongoing research and some DFS which are fully researched and stable in their execution.**

Key Words **– Ceph, DFS, iRODS, HDFS**

I. INTRODUCTION

The Distributed File Systems (DFS) introduces more complexity because the users and devices are physically distributed across locations. A Distributed File System makes it convenient for the users to use files in the distributed environment. It is used to build a hierarchical view of multiple file servers and shares on the network.

More and more data-intensive applications today generate and handle large volumes of data on a regular basis. With the emergence of the recent infrastructure which is running on cloud computing platforms achieves highly scalable data management which is a critical challenge, as the overall application performance is highly dependent on such a highly available data.

In this paper, we have elaborated a presentation and a comparison on various DFSs based on following fundamental issues: Remote Information Sharing, User Mobility, Availability and fault tolerance, Reliability and Caching.

II. BACKGROUND WORK

DFS is a file system that supports the sharing of files in the form of persistent storage over a set of network connected nodes[1]. Apart from persistent storage and information sharing, DFS should provide additional features, such as remote information sharing, user mobility, availability.

Remote Information Sharing – it implies that any file should be transparently accessed from any node, irrespective where the file is located.

User mobility – the system should be flexible enough in such a way that one can work from any node at any instant of time without the need to relocate any storage device.

Reliability and Fault Tolerance– the file should always be available in spite of temporary failure. The DFS must maintain multiple copies on different nodes, which are called as replicas if one of the nodes is unavailable then a client automatically switches over to one of the replicated nodes. [2,3,4,5]

Cache Consistency and coherence – In a file system, the cache is located on client’s node, multiple users may access the same data or file at a particular instant of time. Caches are said to be consistent if they all are consist with the same most-recent data. DFS needs to maintain the consistency among its data copies.

Load Balancing – It is the ability to auto balance the system by adding or removing servers. This feature allows balancing the system load among the resources nearly equally within the grid. A load balanced resource attempts to balance storage and retrieval across nodes to avoid taxing the servers. Here, the automatic deployment plays an important role.

DFS should provide location transparency and redundancy to improve data availability in the face of failure or heavy load. This is possible by allowing shares in multiple different locations to be logically grouped under one folder, or root. AS compared to the local file systems many DFS performances are very low because they perform synchronous I/O operations for cache coherence and data safety [6].

Some of the few challenges which are faced by Distributed File systems which create the major bottleneck in the performance of DFS and they are caches which can be provided at either the file server or client. Another major issue of DFS is the failure of a machine which can be distinguished by the failure of the communication link, slow responses due to heavy overloading. So, when a site doesn’t act in response there are no chances to find out whether the site has failed or stopped processing or whether the communication link has been broken but the site is still operational.

One important issue is a support for storing and processing data on virtual storage devices. Such computing resources are used as per the need basis in clouds[7]. The support of storing and processing data on externalized, virtual storage resources required the very important aspects related to performance, scalability and what type of standard will be given. Some of such devices can also be mobile in nature. These computational resources and data are largely underutilized in today’s mobile applications. Some mobile applications extract and aggregate information from multiple phones. Video and photo publishing applications like Instagram and Twitter or Faceboook allow users to see/ upload or download pictures which are posted by other users. So such have to be shared online. In the same manner, sensor data is uploaded from time to time which is gathered in various applications and is transferred over network lines can be an example of mobile platforms. For such implementation and evaluation of mobile and cloud computing platforms , the obstacles, challenge and solutions have tried to address using the survey of various file systems on the basis of some parameters like fault detection, cache consistency and coherence, replication and load balancing.

III. TAXONOMY OF DISTRIBUTED FILE SYSTEMS

***Ceph DFS* –**Ceph is an open source distributed file system developed by Sage Weil. Ceph addresses three critical challenges of storage systems—scalability, performance, and reliability. It is totally a distributed file system. Ceph delegates responsibility for data migration, replication, failure detection, and failure recovery to the cluster of OSDs(Object Storage Devices) that store the data, while at a high level, OSDs collectively provide a single logical object store to clients and metadata servers. [8,9,10]. CRUSH (Controlled Replication Under Scalable Hashing) has been developed as a pseudorandom data distribution algorithm that efficiently and robustly distributes object replicas across a heterogeneous, structured storage cluster.[11]. Ceph holds both small and big data. Ceph provides algorithms to distribute the metadata workload and allows to dynamically add newservers, the single metadata server of the centralized system is a bottleneck. Ceph succeeds in placing data according to free disk space and moving data from an overloaded server to another one. In Ceph DFS, the load balancing is done almost automatically.

***iRODS DFS* :** iRODS, developed by the Data Intensive Cyber Environment(DICE) group, organizes distributed data up to a rangeof petabytes. Indirectly, it has the ability of processing large data. It is released with an Open-Source-Licence. iRODS stores data on heterogeneous storage systems. iRODS is only beneficial for big files, since small files cannot besplit into blocks. It must be configured to avoid the overload. In iRODS, the commands are manually run to perform a load balancing [12].

***Google File System(GFS)* –**The need of establishing this file system was fulfilled as it met the storage needs and for processing terabytes of data across thousands of disks. A GFS cluster consists of a single master and multiple chunk-servers and is accessed by multiple clients handled in centralized fashion. Files are divided into chunks, each identified by a unique 64-bit handle, and are stored on the local systems as Linux files. Each chunk is replicated at least once on another server, and the default is three copies of every chunk. The master only tells clients (in multibyte messages) which chunkservers have needed chunks. Clients then interact directly with chunkservers for the subsequent operations. In this manner the high availability of nodes and indirectly the data is maintained. A large chunk size decreases network and burden created by metadata. A drawback of a using large chunk size is that “hot-spots” may develop for the chunks of files requested by many clients. Each chunk is replicated on multiple chunkservers. Master node’s process is also restarted if the master or server fails. Neither the client nor the server caches file data[13].

***CODA*(Constant Data Availability)** is a distributed file system developed as a research project at Carnegie Mellon University since 1987 under the direction of Mahadev Satyanarayanan. Codais a distributed file system developed as a research project. It descended directly from an older version of Andrew File System (AFS-2) and offers many similar features. It is slightly slower than AFS and server replication degrades performance by a few percent. Coda uses a local cache to provide access to server data when the network connection is lost. It has some more features like good scalability, Server replication. A decentralized distributed file system to be accessed from autonomous workstations. It allows all servers to receive updates, allowing for a greater availability of server data in the event of network partitions[14]. *C*ODA extends AFS and it provides constant availability through replication. CODA has three fundamental objectives

- Scalability: to build a system that could grow without major problems
- Fault-Tolerance: system should remain usable in the presence of server failures, communication failures and voluntary disconnection[15].

Hadoop Distributed File System(HDFS)–

It is a distributed file system. But the nodes are managed by the centralized single server called as which is termed as the Name Node. HDFS implements a Write Only Read Many model; when a file is created, the data is written onto it. Later the file can be read but modifications are nor possible again. HDFS splits the data into blocks which are replicated and distributed across several datanodes according to a default placement policy. HDFS defines the utilization of server as the ratio of space used to the total capacity of the server; in this manner the load balancing is handled.

Other way out for the balancing the load across the nodes is, replicas are moved from one node to another according to the placement policy. Hadoop is designed to tolerate node failures.

Servers in HDFS are fully connected and communicate with each other to detect some faults such as network or server failures to keep the system available. Every three seconds, datanodes send heartbeats to the namenode saying they are alive and confirm their availability. The namenode considers the datanodes out of service or not in a working condition if hearbeats are not sent. Accordingly the decisions are taken by namenode for the load balancing[16].

IV. ANALYSIS AND COMPARISON OF DFS FOR MOBILE PLATFORM

The mobile platforms can be sensor devices and the data gathered from them can be studied along with the mobile devices and the cloud platforms. The classification has been made one by one for the above listed distributed file systems. The distributed file system, Ceph’s design is made in such a way that it is autonomous, self-healing, and intelligent, enables modern cloud storage infrastructures to place data, rebalance the cluster and recover from faults dynamically but not applicable for mobile platforms[17]. MapReduce-based sensor data processing and access platform for intelligent cities has been implemented on HDFS and the focus of[18] is on using the MapReduce framework to process the raw data uploaded from the sensors, and then using HBase, for distributed, scalable, big data store, to save the sensor data.

Network traffic recording or archiving is always applied in network forensics, network troubleshooting, and user behavior analysis. Since October 2009, China Unicom has been developing a home-brewed big data storage and analysis platform based on the open source Hadoop Distributed File System (HDFS) as it has a long-term strategy to make full use of this Big Data using a distributed HBase which is used for distributed storage[19]. The Distributed File system for Cloud plays an important role to many clients who have access to the same data/file for providing important operations. Though, Scalability and security are fundamental goals in Coda, it plays an important role on Mobile platforms but it says Weak connectivity is a fact of life in mobile computing. Connectivity can be exploited to benefit mobile users of a distributed file system[17].

In sensor networks, raw data is processed locally, not on other nodes and this is ideal situation for preserving power by avoiding network transfers and MapReduce framework which is the part of HDFS prefers to process data locally and it has the capacity to do so by offloading computations to other nodes when necessary[20].

Coda supports disconnected operations allowing clients to access and modify files even disconnected from network which also saves the power. But, Hyrax[21] does not allow disconnected operations because of the data nodes depending on the name nodes for mapping file paths to data blocks. Even unlike Coda, Hyrax does not depend on the central set of servers to host data. Instead, it proves the mobile devices can be used to improve the reliability and performance as it

has implemented Hadoop Distributed File System.

The comparative study has been given below in the tabular format.

DFS Type/ Parameters	**Architecture**	**Reliability**	**Cache consistency& coherence**	**Load Balancing & Fault Tolerance**	**For Mobile Clients**
Ceph	Distributed	Fully connected	Lock	Manual	Not Applied
iRODS	Centralized	P2P	Lock	Manual	Not Applied
GFS	Distributed Cluster Based	TCP	Append Once Read Many model Client/Server doesn't cache data	Load Balancing done by Master	Not Applied
Coda	Distributed	Disconnected operation Fully fault tolerant[22]	Client side whole file caching[23][24]	Manual	Applied
HDFS	Distributed but handled by Centralized Server Cluster Based	Fully connected	WORM Distributed cache	Auto	Applied

CONCLUSION

The DFS is one of the most important and widely used forms of shared permanent storage. The continuing interest in DFS bears the evidences to the robustness of this model of data sharing. Some of the key issues have been addressed here. The features of each DFS have been elaborated here. Last but not the least their role on cloud and mobile platforms have also been compared. From the above classification, it has been found that the HDFS is suitable and provides most of the essential features for a mobile and cloud computing infrastructure. Even there are many more solutions provided by Hadoop which can be directly applied to challenges in a mobile and cloud computing environment. In future, we would like to study the feature of managing and processing data using small and large files and their security on mobile platforms.

REFERENCES

[1] Elizer levy and Abraham Silberschatz “Distributed File Systems : Concepts and Examples” , ACM Computing Surveys, Vol.22, No.4, December 1990.

[2] Chandramohan A. Thekkath, et al, "Frangipani: A scalable Distributed File System", System Research Center, Digital Equipment Corporation, Palo Alto, CA, 1997.

[3] Barbara Liskov, et al, "Replication in the Harp File System", Laboratory of Computer Science, MIT, Cambridge, CA, 1991.

[4] John Douceur and Roger Wattenhofer, "Optimizing file availability in a server-less distributed file system" In Proceedings of the 20th Symposium on Reliable Distributed Systems, 2001.

[5]Yasushi Saito and Marc Shapiro, "Optimistic Replication", ACM Computing Surveys, Vol. 37, No. 1, pp. 42-81, March 2005.

[6] Edmund B. Nightingale, Peter M. Chen, and Jason Flinn, “Speculative Execution in a Distributed File System”, ACM SOSP’05, Brighton, United Kingdom, October 23–26, 2005.

[7] Luis M. Vaquero, Luis Rodero-Merino, Juan Caceres, and Maik Lindner, “A Break in the Clouds: Towards a Cloud Definition” *SIGCOMM Comput.Commun. Rev.*, 39(1), pp 50- 55, 2009.

[8]Weil, S.A., Brandt, S.A., Miller, E.L., Long, D.D.E., Maltzahn, “Ceph: A scalable, High-performance distributed Fil system”, In Proceedings of the 7th Symp. On Operating Systems Design and Implementation (OSDI). Pp 307-320,2006.

[9] Weil, S.A.: Ceph: reliable, scalable, and high-performance distributed storage. PhD Thesis, Santa Cruz, CA, USA, 2007.

[10] Weil, S., Brandt, S.A., Miller, E.L., Maltzahn, C.: Crush: Controlled, scalable,decentralized placement of replicated data. In: Proceedings of SC '06, Nov 2006.

[11] Thesis - Weil, S., Ceph, “Reliable, Scalable, and High-Performance Distributed Storage”, Dec 2007.

[12] Denis Hunich, Ralph M¨uller-Pfefferkorn , “Managing Large Datasets with iRODS - A Performance Analysis”, in proceedings of the International Multiconference on Computer Sc and Information Technology, Published by IEEE, pp 647-654, 2010.

[13] Sanjay Ghemawat, Howard Gobioff, and Shun-Tak Leung, “The Google File System”, SOSP’03, ACM Publications, 2003.

[14] Chieh-Yih Wan, Shane B. Eisenman, Andrew T. Campbell, “CODA: Congestion Detection and Avoidance in Sensor Networks”, proceedings of SenSys’03, ACM

[15]M. Satyanarayanan, J. J. Kistler, P. Kumar, M. E. Okasaki, E. H. Siegel, D. C. Steere , “Coda: A Highly Available File System for a Distributed Workstation Environment”, IEEE TRANSACTIONS ON COMPUTERS, VOL. 39, NO. 4, APRIL 1990.

[16] Apache Hadoop : hadoop.apache.org.

[17] Ceph Architecture and Documentation: http://docs.ceph.com/docs.master/architecture.

[18] Chi-Yi Lin,Chia-Chen Li, Wei-Che Liao ,"A Sensor Data Processing and Access Platform Based on Hadoop for Smart Environments", Network-Based Information Systems (NBiS), Published by IEEE, pp 455-460,2014.

[19] Wenliang Huang, Zhen Chen, Wenyu Dong, Hang Li, Bin Cao, and Junwei Cao ,"Mobile Internet Big Data Platform in China Unicom", Volume 19, Number 1, Published by IEEE, pp 95-101.

[20]M. Satyanarayanan, Mobile Information , IEEE,1996.

[21] Thesis - Eugene E. Marinelli, “Hyrax: Cloud Computing on Mobile Devices using MapReduce”, 2002.

[22]Lily B.Mummert, Maria R. Ebling, M. Satyanarayanan, "Exploiting Weak Connectivity For Mobile File Access", SIGPOS, pp143-155, Dec 199.5

[23]James J. Kistler and M. Satyanarayanan, “Disconnected Operation in the Coda File System”, The Kluwer International Series in Engineering and Computer Science Volume 353, pp 507-535, 1996.

[24]Book : Andrew Tenenbaum, Maarten Steen, “Distributed System Principles and Paradigm”.

AUTHOR’S PROFILE

Madhavi Vaidya

Madhavi Vaidya is currentlly working as Assistant Professor at Vivekanand Education Society’s Arts, Science and Commerce College, Mumbai and has completed MCA and MPhil in Computer Science. Her areas of interest are Databases, Distributed Systems/Databases,Software Engineering, Parallel Architecture,Linux OS, Operating Systems & HadoopArchitecture. She has published 10 research papers in various conferences and a poster at **IRISS- ACM** Goa Conference along with her supervisor and 4 articles including **ACM-Wnewsletter** published in May 2014 named “Leveraging Thermal Power in Data Processing Frameworks”. She has published a book for Vipul Prakashan, Mumbai on “Database Management Systems” She is the member of IEEE and ACM association

Dr. S. P. Deshpande

Dr. Shrinivas P. Deshpande is currently working as Associate Professor for MCA at P. G. Department of Computer Sc. And Technology, DCPE, HVPM, Amravati, India. He did MCA and M.Sc.(Phy) from S. G. B. Amravati University, Amravati and received Ph.D in Computer Science and Engineering from S. G. B. Amravati University, Amravati. He has published number of research papers in National and International Conference. His area of interest includes Software Engineering, Data base, Data Warehousing and Data Mining.