ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 3, March 2015

Associate Adaptable Transactional Information Store in the Cloud Using Distributed Storage and Meta Data Manager

K.G.S. Venkatesan[1], Dr. Kathir.Viswalingam[2], N.G. Vijitha[3]

Research Scholar, Associate Professor, Dept. of C.S.E., Bharath University, Chennai, India [1]

Dean (R & D), Bharath University, Chennai, India[2].

Department of computer Science & Engg., Bharath University, Chennai, India[3].

ABSTRACT: Over the last few years, "Cloud Computing" or "Elastic Computing" has emerged as a compelling and successful paradigm for net scale computing. One of the key causative factors to the current success is that the elasticity of resources. In spite of the snap provided by the infrastructure and therefore the scalable style of the applications, the elephant (or the underlying database), which drives most of those web-based applications, is not terribly elastic and scalable, and thus limits measurability. In this paper, we have a tendency to propose Adaptable transactional that addresses this issue of measurability and snap of the info store during a cloud computing setting to leverage from the elastic nature of the underlying infrastructure, whereas providing scalable transactional knowledge access. This paper aims at providing the planning of a system current; highlight the major style selections, analysing the various guarantees provided by the system, and distinguishing several vital challenges for the analysis community striving for computing within the cloud.

KEYWORDS : Cloud computing, Adaptable transactional, consistency Infrastructure as a Service (IaaS), ACID.

I. INTRODUCTION

"Utility computing" (popularly acknowledged within the business as "Cloud Computing") has been a particularly victorious model for providing Infrastructure as a Service (IaaS) over the net, and has light-emitting diode to the tremendous success of corporations like Amazon as a technology supplier through Amazon net Services (AWS), Sales force INC. and many additional. It's been wide mentioned to be the "dream come back true" for the IT business, with the potential to transform and revolutionize the IT business by creating software even additional engaging [2]. On one finish of the spectrum square measure these IaaS suppliers that offer work out cycles, storage, network information measure etc., whereas on the other finish of the spectrum square measure suppliers like Microsoft's Azure and Google's App Engine World Health Organization offer Platform as a Service (PaaS), and square measure at a far higher level of abstraction. The term "Cloud Computing" encompasses this entire spectrum of services, however during this paper, we mainly focus on the IaaS models [4].

The major reasons for the widespread quality and success of Cloud Computing are:

• No upfront price and Pay-as-you-go model: permits new applications and merchandise concepts to be tested simply and quickly while not vital initial overhead.

• physical property and illusion of infinite resources obtainable on demand: The elastic nature of the cloud permits resources to be allotted on demand permitting applications to easily proportion and down with load changes.

• Transfer of risk: permits the handling of risk, e.g. failures, to be shifted from the smaller computer code as a Service provider, to the larger entities, i.e. the cloud service providers, who square measure higher equipped to mitigate the risks [9].

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 3, March 2015

Typically, web-based applications have a 3-tier design, the Web Server, Application Server, and the Database Server. In general, completely different instances of application servers and net servers among an equivalent application do not share any state info. Therefore, when the appliance load will increase, the appliance server layer and also the net server layer are often simply scaled up by spawning new machine instances that absorb the increased load. However in commonest cases, the information back-end becomes the quantifiability bottleneck, since the database servers don't simply scale. In such a state of affairs, if the information server conjointly had the elastic property of scaling up and down as per the load characteristics, then the entire computer code stack would scale higher. One would possibly argue in favour of information management services like Amazon's Simple DB which might scale to very large amounts of information and large number of requests, however Simple DB and similar ascendible key-value stores like Bigtable [7] and generator [10], although extremely ascendible, stop wanting providing transactional guarantees even on one row.

During this paper, we propose Adaptable transactional, AN Elastic Transactional knowledge Store, which is elastic on equivalent lines because the elastic cloud, while providing transactional guarantees. Adaptable transactional is designed to be a light-weight knowledge store that supports only a sub-set of the operations supported by ancient database systems, and therefore we tend to decision Adaptable transactional a knowledge store, whereas reserving the term databases for additional ancient database systems. Adaptable transactional is analogous to partition off databases [3] that are common in enterprise systems, while adding options and parts essential towards elasticity of the info store. In our style, we lever age from verified info techniques [21, 12] for dealing with concurrency management, isolation, and recovery, while mistreatment style principles of scalable systems such as Bigtable [7] to beat the constraints of distributed database systems [15, 18]. This paper aims at providing the design of a system current; highlight the major style selections, analysing the various guarantees provided by the system, and distinguishing new challenges for the analysis community.

II. RELATED WORK

Based on the experience gained from building distributed database systems [15, 18], researchers and designers have realized that supporting distributed transactions does not permit scalable and accessible styles. Hence, to satisfy the quantifiability needs of net applications, designers have sacrificed the power to support distributed transactions [13]. This resulted within the style of simpler knowledge stores supported the key-value schema, where tables square measure viewed as a large assortment of key-value entries, and the values might need some structure, or may be viewed as un interpreted strings of bytes [9]. Examples of these systems embody Bigtable [7], generator [10], PNUTS [8], Amazon Simple DB, Face book Cassandra, and many additional. These systems limit access graininess to single key accesses, whereas providing smallest consistency and atomicity guarantees on multi-key accesses. This allows the system to horizontally partition the tables, without worrying regarding the requirement for distributed synchronization. These systems are viewed to be the other finish of the spectrum of information management solutions when compared to transactional databases. Especially, these systems square measure designed for top quantifiability and availability, however to not replace the normal databases for consistent transactional properties, and recovery.

Other systems spanning the center ground embody Sinfonia [1], wherever the mini transaction primitive demonstrates that distributed transactions and distributed commitment, if employed in a prudent manner, will be used for the planning of ascendible distributed systems. Similarly, Chubby [5], that uses the valuable Paxos agreement protocol [14], forms the core of ascendible information management systems like Google classification system [11] and Bigtable [7]. These systems demonstrate that even though the paradigm of distributed databases wasn't successful, plenty of vital ideas that were developed for these systems will be effectively employed in trendy scalable systems.

There have been efforts in designing databases for the cloud. Brantner et al [4] suggest a design for a database server that can use Amazon's S3 as a storage layer for the database engine, while the transaction manager might be located inside or outside the cloud. On the other hand, Lomet et al. [16] suggest a radically different approach where they counsel unbundling the dealings and therefore the data manager. Despite the fact that these techniques open several interesting analysis avenues, the authors don't address the problem of elastic scaling of info systems within the cloud.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 3, March 2015

Adaptable transactional has been designed with the goal of ascendible and elastic dealings management within the cloud.

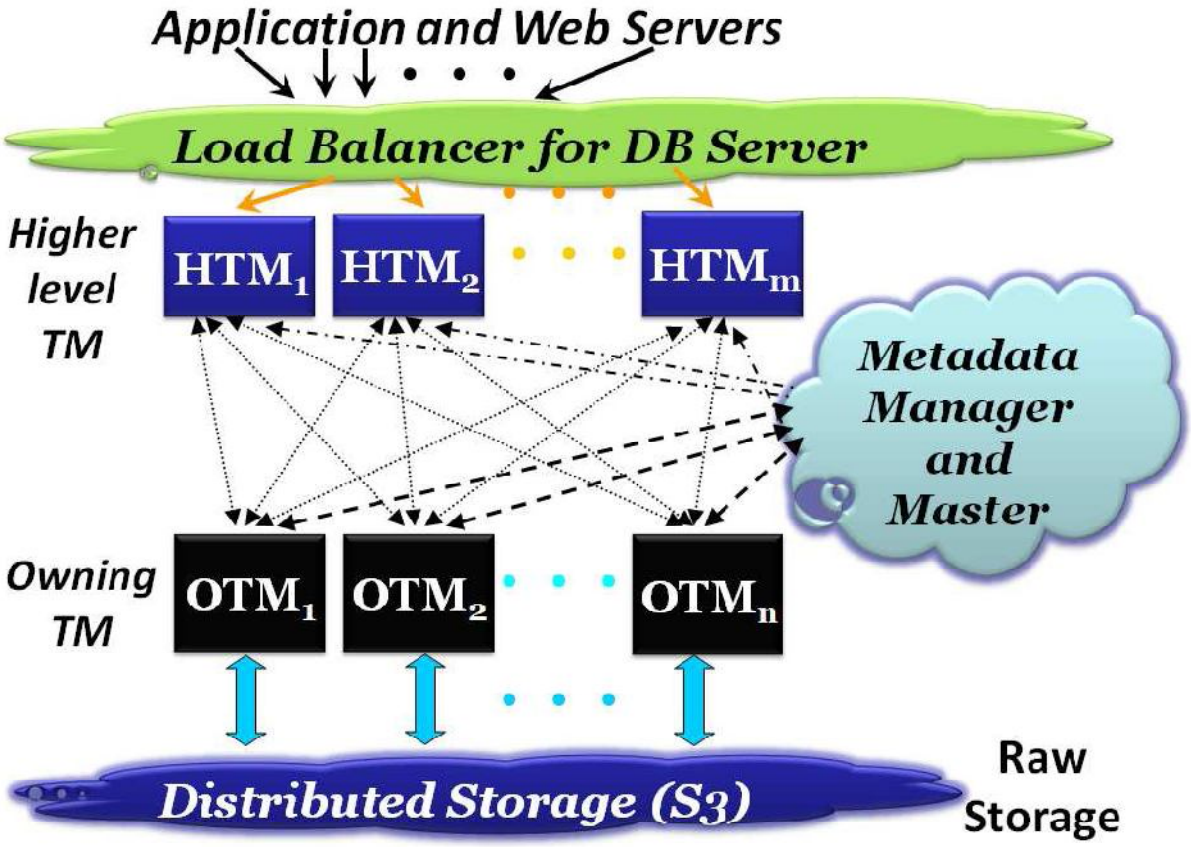

Fig. 1. Overview of the Adaptable transactional system.

III. SYSTEM DESIGN

A. *Data Model*

Practice has shown that the majority application developers do not need a versatile schema as outlined within the SQL commonplace and supported by most information systems. Modern application developers, in most cases, would like systems with simple schema for storing their knowledge, whereas providing quick and economical access to that. The versatile knowledge model supported by standard databases is associate degree overkill in most cases [15, 19]. trendy applications would like a rather increased version of Indexed sequent Access strategies, and this forms the premise for the info model of recent scalable systems like Bigtable [7], generator [10], PNUTS [8], Simple DB etc. For Adaptable transactional, we choose a key-value based mostly style like Bigtable [7] wherever values have application specific structure.

B. *Design Overview*

The Adaptable transactional system has been designed with the intent to provide transactional guarantees in a very climbable manner, rather than retrofitting these options into associate degree existing system. Figure one provides a high-level summary of the look of Adaptable transactional. At the center of the system may be a two level hierarchy

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 3, March 2015

of group action Managers (TM) that square measure responsible for providing transactional guarantees, while providing elastic measurability with increase in demand. At the top of the stack square measure the applying servers and also the web servers that move with the information [21]. Requests to the information square measure handled through the load balancer. When a group action request arrives, the load balancer forwards it to a better Level group action Manager (HTM) based on some load levelling policy. The HTM then decides whether or not it will execute the dealings regionally, or route it to the suitable Owning dealings Manager (OTM) that owns exclusive access rights to the data accessed by the dealings. The particular knowledge for the data store is hold on within the distributed storage layer. All vital state info of the system, i.e. the system state [9] and therefore the information for the tables is managed by the information Manager. All the elements of adaptable transactional area unit situated within the cloud. In Adaptable transactional the information table's area unit divided, and Adaptable transactional may be designed for each static and dynamic partitioning. Static partitioning in Adaptable transactional is analogous to information partitioning [23] – the information designer partitions the information, and Adaptable transactional is chargeable for mapping partitions to specific OTMs, and reassigning partitions with dynamic load characteristics to confirm measurability and snap. In this configuration, the applying is alert to the partitions, and thence may be designed to limit transactions to single partitions. In such a configuration, Adaptable transactional will provide ACID transactional guarantees for transactions limited to a partition. Beneath dynamic partitioning configuration, Adaptable transactional, additionally to managing partition mapping, is additionally chargeable for information partitioning using varies or hash based mostly partitioning schemes [26]. Since the applications don't seem to be alert to the partitions, transactions are not sure to be restricted to one partition. To ensure measurability during a dynamic partitioning setup, and avoid distributed dealings, Adaptable transactional solely supports mini transactions [36], a climbable and primitive with restricted transactional linguistics that ensures recovery but not world synchronization. We have a tendency to currently discuss the different elements of the system, and Section four.1 provides details of varied transactional guarantees provided by Adaptable transactional [26].

C. *Distributed Storage*

The distributed storage layer provides associate abstraction of a fault-tolerant shared disk which may be accessed from anywhere within the network. This abstraction of the storage layer has effectively been employed in the planning of variety of systems like the employment of the Google filing system [11] by Big table and also the use of Amazon S3 in coming up with a info on S3 [37]. The storage layer takes care of replication and fault-tolerance, whereas the appliance accessing the storage ought to make sure that constant object or file isn't being written to at the same time. Considering the fact that synchronous replication is pricey, it can be expected that the storage layer replication are going to be asynchronous and eventually consistent [20]. however if there are a restricted variety of failures, it is simply assumed that the storage layer provides consistent access to single objects, or in alternative words, reads and writes to the storage layer are atomic. Since within the presence of failures, the storage layer would possibly come back stale knowledge, some notion of versions should be related to the info. If the storage layer provides support for versioning, then the system can leverage it, otherwise, versioning ought to be expressly incorporated [29].

D. *Owning dealing Managers*

The Owning dealing Managers (OTM) square measure the entities responsible for the execution of transactions on the partitions of the info's, and have exclusive access rights to the partitions they own. This square measure analogous to the pill servers in Bigtable [27], and own disjoint partitions of the database. Associate in Nursing OTM is to blame for all the concurrency management and recovery practicality for the partitions it owns. Since Associate in Nursing OTM has exclusive access to the set of partitions its own, it will sharply cache the contents of the partition in its native disk, thereby preventing expensive accesses to the distributed storage that actually stores the info. to ensure the sturdiness of committed transactions, all changes created by a dealing should be keep on some medium that may tolerate the failure of the OTM, and permit the system to recover from such failures and guarantee the sturdiness of committed transactions [39].

In order to trot out dynamic partition assignments and the failure of OTMs, and to confirm that only 1 OTM is serving a partition, each OTM acquires a lease for a partition with the data manager, that is revived periodically. If Associate in Nursing OTM has with success non heritable a lease, then the data manager ensures that the OTM has exclusive access

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 3, March 2015

to the partition, and thus the OTM can execute transactions on the partition while not the necessity for distributed synchronization. This mechanism of distributed lease maintenance is analogous to it supported by Chubby [5] and employed in Bigtable [32].

E. *Data Manager and Master*

The data Manager and Master (MMM) is that the brain of the system that stores the system state [9], viz., partition information, mapping of partitions to OTM, leasing information for the OTMs to subsume failures, and monitoring the health of the system. Additionally to providing strong sturdiness and consistency guarantees for the data of the system, this entity conjointly acts as a Master which monitors the health of the system and performs the necessary system maintenance within the presence of failures. The Master monitors the system and ensures that if an OTM fails, then another OTM is instantiated to serve the partition, and conjointly deals with partition duty assignment for load equalization. High consistency of the info hold on in the MMM is bonded through synchronous replication of the contents. For this purpose, we decide a system style similar to that of the ample protection service [34] that uses the Paxos agreement formula [14] for duplicate consistency [6]. Paxos guarantees safety and consistency in the presence of discretional failures; however the supply of the system isn't warranted within the presence of failures. Within the AWS infrastructure, failures in one availability zone square measure isolated from failures in alternative handiness zones. Hence, high handiness will be achieved by the even handed placement of replicas so related to failures do not have an effect on a majority of the replicas. Note that the presence of Paxos [14] within the core makes write accesses to the MMM pricey. As a result, the MMM shouldn't be heavily loaded with a large range of requests. Since the MMM doesn't reside within the knowledge path, and in most cases the purchasers of the system will cache the data, the MMM shouldn't be a performance bottleneck for the system [33].

F. *Higher level group action Managers*

The Higher level group actions Managers (HTM) are designed to absorb all read-only transactions within the employment. HTMs cache subsets of the information for read-only purposes, and answer queries from its cache. In static partitioning, transactions related to one partition are routed to the suitable OTM that executes the transactions. For mini transactions [31], the HTM becomes the organizer, whereas OTMs owning the partitions accessed by the mini transaction are the cohorts. Neither read only transactions nor mini transactions associate any state with the HTMs. In each case, the HTMs do not have any state coupling with OTMs, and also the variety of OTM and HTM instances are often completely different relying on the system configuration and also the load on the system. For routing asking to the suitable OTM, the HTM caches the mapping of partitions to OTMs. Unlike the OTMs, that are accountable for solely partitions of the information, the HTMs span the whole information for read-only queries and mini transactions [35].

IV. SIMULATION

A. *Transaction Management*

In a statically partitioned setup, applications can limit transactions to single partitions. Within a partition, Adaptable transactional provides ACID guarantees similar to transactions in databases. The only subtle difference is in the level of consistency (the C in ACID) guaranteed by Adaptable transactional [38]. To obviate distributed synchronization, and minimize the impact of a single TM failure on the operation of the remaining TMs, Adaptable transactional guarantees consistency only within a partition of the database and there is no notion of consistency across partitions or global serializability [21]. Efficient performance can therefore be achieved since an OTM is guaranteed exclusive access to the partitions it owns, and proven techniques for concurrency control, isolation, and recovery in traditional databases [21, 12] can be used in designing the OTM. Since associate OTM has exclusive access to the partition, it can sharply cache the info contents in its native disk, and therefore the updates are often asynchronously applied to the distributed store during a manner like info check pointing [31]. Since dynamic partitioning doesn't allow applications to limit transactions to one partition, Adaptable transactional solely supports mini transactions [11] during this setup, whereas supporting each sort of transactions for a statically divided setup. In corporal punishment a mini transaction, there is no

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 3, March 2015

want for synchronization among its taking part OTMs, and hence, mini transactions are often with efficiency executed among Adaptable transactional.

B. *Recovery done*

Failure of OTM: to confirm the sturdiness of transactions, OTMs use write-ahead work. As a result, techniques such as ARIES [17] and different similar recovery techniques [21] may be used for economical recovery. In the AWS model, once a machine instance crashes, its native disk is additionally lost. Therefore, to ensure the durability and persistence of transactions on the far side the failure of an OTM, the log entries of committed transactions should be hold on some persistent storage. But since work should be performed throughout traditional dealing execution, the performance of associate OTM would suffer if the log entries area unit forced to a distributed store like S3. AWS provides a stronger answer within the kind of Elastic Block Storage (EBS) that provides persistence beyond instance failures, permitting log entries to be stored on compass point [35]. Our initial experiments with compass point showed that for consecutive reads and writes, the performance of EBS is equivalent to disks related to AWS machine instances. Therefore, victimization compass point for each caching data pages, and storing logs would supply higher performance, though at a better greenback value. Once associate OTM fails, its lease with the information manager expires, and once the master notices that the OTM instance has failing, it will instantiate a replacement OTM, that recovers the state of the failing OTM from the logs. An identical mechanism is used in Bigtable [37] to wear down pill server failures. Failure of HTM: little state is related to the HTMs and therefore, handling HTM failures is simpler. All progress of a read-only dealing is lost once the HTM executing it fails. Purchasers of such transactions outing, and application specific recovery triggered. For mini transactions , no state is related to the organizer, and therefore the recovery principles delineated in Sinfonia [1] may be accustomed recover a mini transaction from HTM or organizer failures.

C. *Snap*

The ability of the system to touch upon dynamic partition reassignments are that the basis for the snap of the information store. Once the load on the system is low, the system can operate with a little variety of HTM and OTM instances [39]. As the load on the system will increase, the partitions can be reassigned to lesser loaded OTMs, or new OTMs may be spawned to soak up the load and take possession of some partitions from the heavily loaded group action managers. Since the HTMs don't share any state information, spawning new HTMs is straightforward, and amounts to spawning new instances which will in real time begin fascinating the load [40]. On the opposite hand, once a brand new OTM instance is made, it's to accumulate a lease from the metadata manager, acquire management for a few of the partitions, update the data to replicate the changed mapping, and perform any partition specific recovery. Once this is done, the new OTM is prepared to execute transactions for the partition, whereas the previous OTMs reject or redirect transactions of the reassigned partitions [47]. Similarly, when the load decreases, the system will notice this and judge to get rid of OTM and/or HTM instances. Removal of HTM instances needs notifying the load balancer regarding the update, then stopping the physical instance. Removing OTM instances needs updates to the data and therefore the transfer of management of the partitions to another OTM. Once management is transferred, the physical OTM instance may be stopped. Since AWS uses a pay-as-you-go model, stopped instances don't incur any cost, and hence, at low masses, the price of in operation the data store conjointly reduces [46].

V. CONCLUSION & FUTURE WORK

The projected style of Adaptable transactional will give ACID guarantees for transactions that square measure restricted to one partition. Apply has shown that almost all internet workloads are restricted to single object accesses [49], and hence, these transactional accesses may be trivially supported by Adaptable transactional. additionally, most enterprise systems square measure designed for statically partitioned off databases, and transactions are restricted to one information partition, and Adaptable transactional will give economical, scalable, and elastic transactional access to the partitioned off information store. In such a scenario, the Adaptable transactional style will still give snap and scalability by dynamic partition duty assignment based on the load on the system, whereas providing sterilisable transactional guarantees among a partition. In addition, for applications wherever the designer doesn't want to statically partition the information, Adaptable transactional will dynamically partition the information [45]. Additionally to

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 3, March 2015

single object transactions, Adaptable transactional may be simply extended to support mini transactions as outlined by the Sinfonia system [48], however versatile transactions don't seem to be supported within the interest of scalability and snap. supported these needs of modern applications, we tend to square measure within the method of formalizing the varied kinds of transactions that may be with efficiency dead by Adaptable transactional, whereas having minimal impact of application style and conserving the 3- tier design of web-servers, application servers, with Adaptable transactional exchange the information servers within the cloud, thus providing a high degree of snap and adaptability throughout the complete design .

VI. ACKNOWLEDGEMENT

The author would like to thank the Vice Chancellor, Dean-Engineering, Director, Secretary, Correspondent, HOD of Computer Science & Engineering, Dr. K.P. Kaliyamurthie, Bharath University, Chennai for their motivation and constant encouragement. The author would like to specially thank **Dr. A. Kumaravel, Dean, School of Computing,** Bharath University for his guidance and for critical review of this manuscript and for his valuable input and fruitful discussions in completing the work and the Faculty Members of Department of Computer Science & Engineering. Also, he takes privilege in extending gratitude to his parents and family members who rendered their support throughout this Research work.

REFERENCES

1. J. B. Rothnie Jr., P. A. Bernstein, S. Fox, N. Goodman, M. Hammer,T. A. Landers, C. L. Reeve, D. W. Shipman, and E. Wong, "Introduction to a System for Distributed Databases", (SDD-1).ACM Trans. Database Syst., 5(1), PP. 1–17, 1980.
2. B. G. Lindsay, L.M. Haas, C.Mohan, P. F.Wilms, and R. A. Yost, "Computation and communication in R*: A distributed database Manager", ACM Trans. Comput. Syst., 2(1):24–38, 1984.
3. F. Chang, J. Dean, S. Ghemawat, W. C. Hsieh, D. A. Wallach,M. Burrows, T. Chandra, A. Fikes, and R. E. Gruber, "Bigtable: A Distributed Storage System for Structured Data. In *OSDI*, PP 205–218, 2006.
4. K.G.S. Venkatesan. Dr. V. Khanna, Dr. A. Chandrasekar, "Autonomous system for mesh network by using packet transmission & failure detection", Inter. Journal of Innovative Research in computer & comm. Engineering, Vol. 2, Issue 12, PP. 7289 – 7296, December - 2014.
5. Teerawat Issariyakul, Ekram Hoss, "Introduction to Network Simulator".
6. K.G.S. Venkatesan and M. Elamurugaselvam, "Design based object oriented Metrics to measure coupling & cohesion", International journal of Advanced & Innovative Research, Vol. 2, Issue 5, pp. 778 – 785, 2013.
7. S. Sathish Raja, K.G.S. Venkatesan, "Email spam zombies scrutinizer in email sending network Infrastructures", International Journal of Scientific & Engineering Research, Vol. 4, Issue 4, PP. 366 – 373, April 2013.
8. G. Bianchi, "Performance analysis of the IEEE 802.11 distributed coordination function," IEEE J. Sel. Areas Commun., Vol. 18, No. 3, pp. 535–547, March - 2000.
9. K.G.S. Venkatesan, "Comparison of CDMA & GSM Mobile Technology", Middle-East Journal of Scientific Research, 13 (12), PP. 1590 – 1594, 2013.
10. P. Indira Priya, K.G.S. Venkatesan, "Finding the K-Edge connectivity in MANET using DLTRT, International Journal of Applied Engineering Research, Vol. 9, Issue 22, PP. 5898 – 5904, 2014.
11. K.G.S. Venkatesan and M. Elamurugaselvam, "Using the conceptual cohesion of classes for fault prediction in object-oriented system", International journal of Advanced & Innovative Research, Vol. 2, Issue 4, PP. 75 – 80, April 2013.
12. Ms. J.Praveena, K.G.S. Venkatesan, "Advanced Auto Adaptive edge-detection algorithm for flame monitoring & fire image processing", International Journal of Applied Engineering Research, Vol. 9, Issue 22, PP. 5797 – 5802, 2014.
13. K.G.S. Venkatesan. Dr. V. Khanna, "Inclusion of flow management for Automatic & dynamic route discovery system by ARS", International Journal of Advanced Research in computer science & software Engg., Vol.2, Issue 12, PP. 1 – 9, December – 2012.
14. Needhu. C, K.G.S. Venkatesan, "A System for Retrieving Information directly from online social network user Link ", International Journal of Applied Engineering Research, Vol. 9, Issue 22, PP. 6023 – 6028, 2014.
15. K.G.S. Venkatesan, R. Resmi, R. Remya, "Anonymizimg Geographic routing for preserving location privacy using unlinkability and unobservability", International Journal of Advanced Research in computer science & software Engg., Vol. 4, Issue 3, PP. 523 – 528, March – 2014.
16. Selvakumari. P, K.G.S. Venkatesan, "Vehicular communication using Fvmr Technique", International Journal of Applied Engineering Research, Vol. 9, Issue 22, PP. 6133 – 6139, 2014.
17. K.G.S. Venkatesan, G. Julin Leeya, G. Dayalin Leena, "Efficient colour image watermarking using factor Entrenching method", International Journal of Advanced Research in computer science & software Engg., Vol. 4, Issue 3, PP. 529 – 538, March – 2014.
18. R.Baeze-yates, C.Hurtado, and M.Mendoza, "Query Recommendation exploitation question Logs in Search Engines", Proc.Int'l Workshop Current Trends in information Technology, PP. 588-596, 2004.
19. K.G.S. Venkatesan. Kausik Mondal, Abhishek Kumar, "Enhancement of social network security by Third party application", International Journal of Advanced Research in computer science & software Engg., Vol. 3, Issue 3, PP. 230 – 237, March – 2013.
20. S. Das, S. Antony, D. Agrawal, and A. El Abbadi, "Clouded Data: Comprehending Scalable Data Management Systems", Technical Report 2008-18, UCSB, 2008.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 3, March 2015

21. K.G.S. Venkatesan, M. Elamurugaselvam, "Design based object oriented Metrics to measure coupling & cohesion", International journal of Advanced & Innovative Research, Vol. 2, Issue 5, PP. 778 – 785, 2013.
22. Annapurna Vemparala, Venkatesan.K.G., "Routing Misbehavior detection in MANET'S using an ACK based scheme", International Journal of Advanced & Innovative Research, Vol. 2, Issue 5, PP. 261 – 268, 2013.
23. W. Vogels. Eventually consistent. *Commun. ACM*, 52(1):40–44,2009.
24. K.G.S. Venkatesan. Kishore, Mukthar Hussain, "SAT : A Security Archicture in wireless mesh networks", International Journal of Advanced Research in computer science & software Engg., Vol. 3, Issue 3, PP. 325 – 331, April – 2013.
25. Annapurna Vemparala, Venkatesan.K.G., "A Reputation based scheme for routing misbehavior detection in MANET"S ", International Journal of computer science & Management Research, Vol. 2, Issue 6, June - 2013.
26. K.G.S. Venkatesan, "Planning in FARS by dynamic multipath reconfiguration system failure recovery in wireless mesh network", International Journal of Innovative Research in computer & comm. Engineering, Vol. 2, Issue 8, August - 2014.
27. T. Joachims, "Optimizing Search Engines exploitation Clickthroygh information", *Proc. ACM SIGKDD*, 2002.
28. K.G.S. Venkatesan and M. Elamurugaselvam, "Using the conceptual cohesion of classes for fault prediction in object-oriented system", International journal of Advanced & Innovative Research, Vol. 2, Issue 4, PP. 75 – 80, April 2013.
29. G. Weikum and G. Vossen, "Transactional information systems: theory, algorithms, and the practice of concurrency control and recovery", Morgan Kaufmann Publishers Inc., 2001.
30. K.G.S. Venkatesan, "Comparison of CDMA & GSM Mobile Technology", Middle-East Journal of Scientific Research, 13 (12), PP. 1590 – 1594, 2013.
31. J. Gray and A. Reuter, "Transaction Processing: Concepts and Techniques", Morgan Kaufmann Publishers Inc., 1992.
32. K.G.S Venkatesan, D. Priya, "Secret-Key generation of Mosaic Image", Indian Journal of Applied Research, Vol. 3, Issue 6, PP. 164-166, June - 2013.
33. R. Karthikeyan, K.G.S. Venkatesan, M.L. Ambikha, S. Asha, "Assist Autism spectrum, Data Acquisition method using Spatio-temporal Model", International Journal of Innovative Research in computer & comm. Engineering, Vol. 3, Issue 2, PP. 871 – 877, February - 2015.
34. K.G.S. Venkatesan, AR. Arunachalam, S. Vijayalakshmi, V. Vinotha, "Implementation of optimized cost, Load & service monitoring for grid computing", International Journal of Innovative Research in computer & comm. Engineering, Vol. 3, Issue 2, PP. 864 – 870, February – 2015.
35. M. K. Aguilera, A. Merchant, M. Shah, A. Veitch, and C. Karamanolis, Sinfonia, "A new paradigm for building scalable distributed systems" In *SOSP*, pages 159–174, 2007.
36. K.G.S. Venkatesan, B. Sundar Raj, V. Keerthiga, M. Aishwarya, "Transmission of data between sensors by devolved Recognition", International Journal of Innovative Research in computer & comm. Engineering, Vol. 3, Issue 2, PP. 878 – 886, February - 2015.
37. G. DeCandia, D. Hastorun, M. Jampani, G. Kakulapati, A. Lakshman,A. Pilchin, S. Sivasubramanian, P. Vosshall, and W. Vogels, "Dynamo: Amazon's highly available key-value store", . In*SOSP*, PP. s 205–220, 2007.
38. K.G.S. Venkatesan, N.G. Vijitha, R. Karthikeyan, "Secure data transaction in Multi cloud using Two-phase validation", International Journal of Innovative Research in computer & comm. Engineering, Vol. 3, Issue 2, PP. 845 – 853, February - 2015.
39. H. Baer. Whitepaper: Partitioning in Oracle Database 11g. Technical report, Oracle, June 2007.
40. M. Brantner, D. Florescu, D. Graf, D. Kossmann, and T. Kraska, Building a database on S3. In *SIGMOD*, pages 251–264, 2008.
41. M. Burrows. The Chubby Lock Service for Loosely-Coupled Distributed Systems. In *OSDI*, pages 335–350, 2006.
42. T. D. Chandra, R. Griesemer, and J. Redstone, "Paxos made live: An engineering perspective", In *PODC*, pages 398–407, 2007.
43. K.G.S. Venkatesan and M. Elamurugaselvam, "Using the conceptual cohesion of classes for fault prediction in object-oriented system", International journal of Advanced & Innovative Research, Vol. 2, Issue 4, PP. 75 – 80, April 2013.
44. S. Ghemawat, H. Gobioff, and S.-T. Leung, "The Google file system" .In *SOSP*, PP. 29–43, 2003.
45. P. Helland, "Life beyond distributed transactions: an apostate's opinion", In *CIDR*, PP. 132–141, 2007.
46. L. Lamport, "The part-time parliament", ACM Trans. Comput. Syst., 16(2), PP. 133–169, 1998.
47. D. B. Lomet, A. Fekete, G. Weikum, and M. J. Zwilling, "Unbundling transaction services in the cloud", In CIDR Perspectives, 2009.
48. C. Mohan and H. Pirahesh, "ARIES-RRH: Restricted Repeating of History in the ARIES Transaction Recovery Method", In *ICDE*, PP. 718–727, 1991.
49. W. Vogel, "Data access patterns in the amazon.com technology platform", In *VLDB*, PP. 1–1. VLDB Endowment, 2007.