Virtualization Model of a Large Logical Database for Diffused Data by Peer-to-Peer Cloud Computing Technology

Takeshi TSUCHIYA, Tadashi MIYOSAWA, Hiroo HIROSE
Faculty of Business Administration and Information
Tokyo University of Science, Suwa
Chino City, Nagano, Japan
Email: {tsuchiya.takeshi, miyosawa, hirose }@rs.tus.ac.jp

Keiichi KOYANAGI
Faculty of Science and Engineering
Waseda University
Kitakyushu City, Fukuoka, Japan
Email: keiichi.koyanagi@waseda.jp

***Abstract*—In this paper, we propose the use of peer-to-peer cloud computing technology to integrate databases that are distributed (diffused) throughout the Internet. We also propose the use of a large, virtualized, logical database that is managed using Structured Query Language (SQL). Combining databases into larger collections of data-"big-data" can make possible wonderful new services. Our proposed model has two main characteristics: First, SQL controls and manages data relationships using relational database management systems and key value stores, but discards location information. Second, service is made scalable by collaborations among distributed nodes. From the results of our evaluation, our model has sufficient service scalability for collaboration between distributed nodes, but not sufficient performance for big-data platforms.**

***Keywords*-Distributed Databases, Peer-to-Peer, Cloud Computing, Service Virtualization**

I. INTRODUCTION

Recent increases in data traffic from many type of web services indicate that the Internet services are being generalized and diversified. Almost all databases for web services use the relational database management system (RDBMS) model, which manages several types of contents concentrated on a specific nodes, but does nott allow for easy scaling of services. In particular, this model provides atomicity, consistency, isolation, and durability (ACID) characteristics for data and services. However, the scalability of node distribution is limited to two or three nodes in master/slave relationships. The master node manages all data exclusively. Therefore, the master node cannot usually process amounts of data, such as more than 10,000 request per second or operations more than a million data. However, this model includes some useful functions based on ACID characteristics for processing large amounts of data.

In recent years, several types of databases have come into use, which use the "No SQL" model. One type is a key value store (KVS) database, which focuses on service scalability and continuity [2][3]. No SQL databases provide service scalability, load balancing, and high availability that can scale out manner. However, they do not have some basic but useful functions that are provided by RDBMS databases, such as transactions, complicated retrieval by conditions, and aggregate processing. These new database services continuously involve the above functions in the view of service and require to construct same functions as middleware of application. These database services distributed throughout the Internet and are distinguished into the mentioned two types. These perform the same service functions as data management services, but have a different attitude. Therefore, it is difficult to unify all databases on the Internet according to either model. There can also be several types of models of the Internet services.

Currently, several services such as social networking services (SNSs) correct and manage No SQL databases with large logical spaces. The databases are used for recommendation services and Web-marketing information. The types and quality of information that can be retrieved from each service are limited and depend on each service. Sharing of contents data among services have possible to generate the new points of view from these. However, generating new points of view requires processing and analyzing data that have been combined from many databases in each service, which is difficult for No SQL databases to do adaptively for the above-mentioned reason. During processing, complex data must be moved and merged into a new No SQL database for control and analysis.

This paper proposes the use of peer-to-peer (P2P) cloud-computing technologies to create a logical data management space for integrating databases that are distributed throughout the Internet. Every user can acquire and control all data connected to the proposed logical space and generate schema information and transactions among the distributed databases. In other words, the proposed model virtualizes these database and shares integration data management in same way. Therefore, following the process out lined in this this paper should improve the quality of database service by integrating distributed data throughout the Internet, scaling up RDBMS functions, and clarifying the use of big data.

This paper is devided into five parts; Section 2 discusses the proposed model based on requirements from the analysis of current databases. The results are evaluated in Section 3. Section 4 discusses possibility and usage as system. Section

 ISBN: 978-1-61208-318-6

5 concludes this paper.

II. Logical Integration Model for Disributed Databases

This section discusses and propose our overlay network model for distributed databases over the Internet.

A. Requirements for Model

Several types of databases are used for network services, which differ in functions and characteristics. Our proposal method enables access to all data on these databases without minds of their belonging and allows them to be managed independently. Nodes must be flexible, adapting to service conditions such as the No SQL model, for big-data service, and control functions, such as RDBMS model. The interface protocol of the proposed model also affects usability for developers and current resources. Therefore, its interface is is expected to use similar the current SQL. Each database service on the Internet is independently are managed under each definition of policy for security in advance. The method of management in the proposed model must overcome and manage these differences. Therefore, our model can flexibly adapt policies from the logical space.

B. Creating Logical Spaces Via Node Collaboration

This section discusses and clarifies the method of constructing a logical space using distributed computing technology.

The logical overlay networks among databases are not provided by static servers but are constructed by collaboration among nodes in regards to service scalability. We have discussed about P2P distributed platform technology in [1] and [8]. All nodes and objects on this overlay network are identified by 128-bit IDs. This platform provides flexible and dynamic node management for general normal nodes to demand for features such as node movement with continuous services and scaling up using P2P cloud computing technologies. This platform also manages data or pointer information for data on the databases of component nodes.

Nodes ware selected for the overlay network based on the author's presidential research in [8]. Under this algorithm, nodes are sequentially selected on the basis of their conditions and performance: they are divided into some roll on overlay network demand of service situation. This election algorithm is defined as a combination of complex calculations based on the features of services such as stability of service, network quality, and processing power. These selected nodes are forming the best-effort overlay network.

C. Method of Managing Distributed Data

This section discusses the mapping of distributed data on the overlay network and clarifies how to manage data on distributed nodes.

All distributed data and tables on databases over the Internet are mapped to the overlay network in the following way, and they are virtualized as a single large database. Selected component nodes behave as service managers of the logical overlay network and adjust the range of the overlay network. The area of the network is 64bit by 64 bit –two-dimensional logical space. Mapped data and tables are allocated in this range, and the range is identified by a unique 128bit object ID. Each piece of data can be easily accessed using this range ID and sequential coordinate information. This ID is generated in the form "URI + the original name of the database (table). " The hashed value of the key is the object ID, and it points to an area on the overlay network. This point, called the starting point, is shown in Fig. 1 at the top of mapping, and mapped data and tables are allocated after it. Therefore, all starting points can be acquired above mentioned two types of information on the large overlay network. Applications and users are easy to derive. The range of mapped data is adjusted and allocated by demand of their quantity. This method is described in section II-D.

Each starting point manages the mapping range of data and information from the original database. The information from all starting points is shared as cashed information among component nodes of the overlay network. The range of mapped data can over lap in another range on the overlay network, such as the starting point, and mapped data using the ensured range for each service. When the expectant range for mapping data has already been ensured by other mapped data or tables, the derivation of the expected starting point uses "object ID + table name + 1" as the key of a hashed function. As a result, all starting points are displaced the range ensured others. New expected range need to place without ensuring as other range. The derivation of the starting point is incremented iteratively: "object ID + table name + 2" until the range for data mapping is ensured.

The size of the mapping range is not confined to applications and users. Allocated size for mapping is assumed to determine the amount of mapped data on current databases and estimated additional data in services. When data is unexpectedly incremented, the new range is added to the current range, and the platform updates range information managed at the starting point. Although frequent addition to the range makes data distribution unbalanced, the proposed overlay network is adapted to the function for the re-allocation of area or for autonomous scaling bases on demand and situation.

Data and tables having in data relationships such as an RDBMS are mapped to ensured ranges with their data relations. Data without RDBMS relationships are mapped to there by sequence of each object ID.

D. Security Method for Data Mapping

Updating between mapped data and original data in each distributed database is restricted in the following two ways to ensure data consistency.

 ISBN: 978-1-61208-318-6

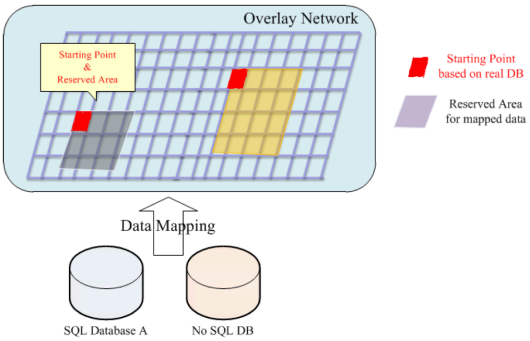

Figure 1. Data Mapping on an Overlay Network

(i) Data management is allowed only for mapped data by overlay network functions with interdicting original data on distributed databases.
(ii) The direction of data control flow is limited: only original data can be update and added, and only overlay network function can adjust the read and controls for mapped data.

Although data management in our proposed model has these restrictions, it satisfies the eventual consistency model discussed in P2P distributed services, and ensures data consistency in some range.

The restriction on data management, restriction (i) assumes that all management of mapped data is shared among applications and users such as in current RDBMS service. This restriction is in regards to the use of the overlay network as a large database.

For restriction (ii), an instance of the process monitoring change of the original data is allocated on the neighbor of the original databases. Although this manner sensitively monitors the differences, the delay of data consistency is less than with the previous method. This method also allows the adaptation of current important service data called big data to the overlay network easily. In particular, the distribution of current data among No SQL model databases enables the generation and management of relationships among data.

E. Management of Coordinate Spaces as One-Dimensional Information

Mapped data on an overlay network is managed on a 64bit×64bit range of a two dimensional logical coordinate space which is similar to an RDBMS model database. Managing this two-dimensional space to the distributed algorithm is necessary for management by collaborations among selected nodes. It uses Z-Ordering [9] for reducing dimensional information. This manner makes 64bit ×64bit two-dimensional logical information converting to 128 bit one-dimensional information. This 128 bit value corresponds to the point of two-dimensional coordinates. All nodes with the same platform can derive this reduced value based on

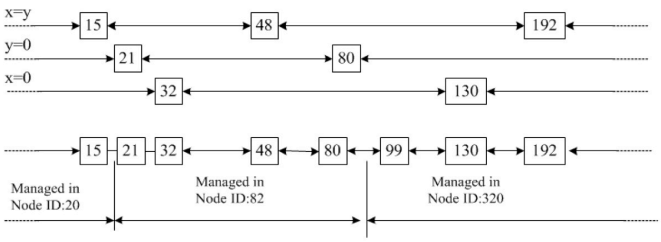

Figure 2. Management of Reduced Coordinate Information

that cordial information for the allocation of data without inquiry to other nodes. The use of this algorithm reduces load without requiring derivation protocol.

The reduced coordinate information is managed by the structure of a layered list such as a Skip List [10] among nodes shown in Fig. 2. Each component node is described sequentially by node ID in Fig. 2. They only manages node ID information of predecessor and successor node.

The three layers from the top layer shown in Fig. 2 provide the list information of the horizontal axis, vertical axis, and diagonal axis on based on two-dimensional information. Managing the information from typical coordinate points affects the efficiency of the acquisition of data among distributed nodes that manage the above mentioned list information.

The bottom layer in Fig. 2 is the list structure of all coordinate points. When the data is allocated to this coordinate points, this list adds information about mapped data or pointer information to the data. Component nodes divide the list information of bottom layer in the order of their node ID. Each node only knows and manages some objects with smaller object IDs than themselves as same as ring management manner of Chord [11]. Component nodes enable the inference of who manages objects sweepingly by object ID without inquiry into other nodes. This protocol reduces the ability to acquire coordinate information efficiently. At the same time, the distribution of an object ID would not concentrate in a typical range of space using a hash function

F. Interface for Applications

The SQL commands generated from applications are received and analyzed at the interface layer shown in Fig. 3. The data management method from users and applications discussed in section II-C enables the attainment of mapped data using start point information which hashed object ID and table name. This overlay network corrects information from mapped databases using received commands among distributed databases and enables the generation of new tables mapped to themselves. Therefore, our proposed model enables the provision of data management service such as RDBMS without the distribution of databases.

 ISBN: 978-1-61208-318-6

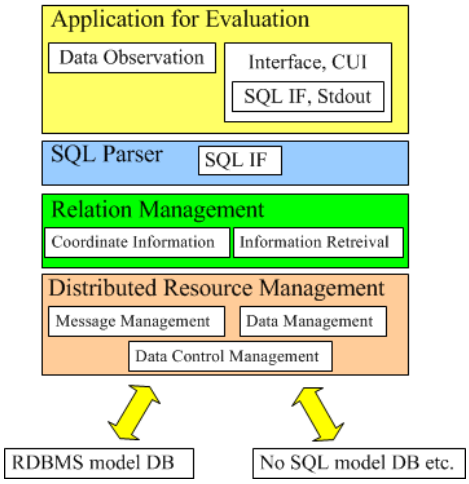

Figure 3. Current Implementation for Evaluations

Table I
IMPLEMENTATION ENVIRONMENT

OS	Windows 7 Professional
CPU	Xeon 2.4GHz×2
RAM	4 GB
DB	My SQL 5.1.44
DevLang	Java SDK 1.6

G. Implementation

The above mentioned fundamental functions are implemented as shown in Fig. 3. This model works as middleware and provides the mentioned data management functions for several types of applications. This implementation model is depicted in a Table I environment and evaluated below. The logical overlay network is composed of three layers, and the outline of the functions of each layer is described in Fig. 3. In this evaluation, the application is set for a specialized e-commerce service.

III. EVALUATION

The proposed logical database model is evaluated on the basis of the serviceability and availability of e-commerce that is implemented using the model.

A. Emulation Scenario

Evaluation using real nodes and environment is difficult because of their distribution scalability, and a simulation can indicate the abilities of the proposed model in an applicable distributed environment. We conduct simulations of implemented nodes.

The simulation environment is constructed by two nodes, each of which manages a hundred other nodes. This environment enables us to dynamically construct the network at the application layer such as cloud computing with hundreds of order of nodes. Generally speaking, the common RDBMS model can be composed of a few distributed nodes at most. On the other hand, the No SQL model is composed of lots of distributed nodes without a limitation – what is called a ”scale free” model. Therefore, this simulation environment is necessary because of the hundreds of nodes. Our evaluation in this paper is constructed by two hundred distributed nodes.

Nodes are executed as independent instances on the Internet, and they behave independently based on their data. All interactions in this environment use the same protocols and methods as the Internet protocol, and all instances are allocated to a server by a unique node ID: Instances with odd node IDs are distributed to one node, and instances with even node IDs to the other. Therefore, communications among the nodes (instances) configured in the skip list shown in Fig. 2 generate communication traffic between these servers and enable the simulation of realistic situations. Data management on these nodes is connected to six independent databases of the RDBMS model,as shown in Fig. 3. All the including data and tables are mapped to the proposed logical overlay network and then evaluated.

B. Simulation Environment

This simulation uses the TPC-W [12] standard for e-commerce services, which was created by the Transaction Processing Performance Council (TPC).

Our proposed model integrates six independent databases for e-commerce services using cloud technologies and provides access to all data on these databases without caring about location of its belongings. All of these integrated databases use the RDBMS model, and they include management information for each service entity, such as a customer or commodity.

Simulating user accesses to integrated overlay network in the same time, more than a hundred processes are executed on this simulation environment. In this case, a process corresponds to a user. Each user (process) starts to access web page starting with the top one and retrieves some types of commodities using queries randomly generated on the overlay networks. Each page that is retrieved results includes one-hundred commodities per page. If the results of retrieval are a thousand commodities, 10 pages are generated. This evaluation clarifies the elapsed time from the generation of the process to the generation of the pages, including the results of the retrieval query as a response. This evaluation also discusses the serviceability and availability of the proposed overlay network approach for distributed databases.

C. Evaluation Results

The following sections are discussed and include the results of evaluations for technology points.

1) Availability and Responsiveness of database service: Regarding the serviceability of node distribution as a database service, Fig. 4 compares two types of average

 ISBN: 978-1-61208-318-6

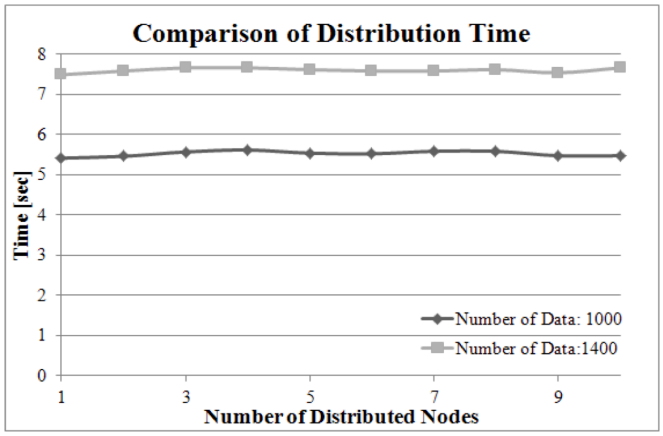

Figure 4. Relation of data quantity and node distributions

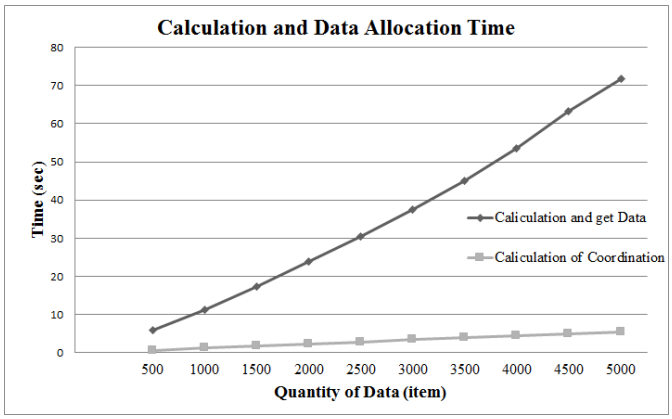

Figure 5. Serviceability of proposed model

elapsed time, which means the difference between pages generated in response to a query (10 and 14 pages) including results (1,000 and 1,400). The x axis shows the number of distribution nodes, and the y axis shows the time until generating response pages from the first access. These curves in the graph indicate similar trends and stable lines until 10 nodes unrelated to node distributions. These trends have continued over 200 of node distributions, although the lines are not described in this graph. The graph shows that node distribution does not affect the serviceability of the proposed model caused by the control and management of data. The graph also shows that the elapsed time is relative to the increase in the number of results (data items) from $1,000$ to $1,400$ data. The graph indicates that time can be incremented by about 85% in both cases caused by node distributions. From the result, service scalability caused by node distribution data does not affect the protocol and serviceability in the logical area.

2) Serviceability:: The data management method for overlaying the network discussed in sec. II-C used two types of method due to limitation of data replication. Fig.5 indicates the comparison of these manners and each characteristic. The x axis shows the number of retrieved results, and the y axis shows the elapsed time for processing. In Fig. 5, the line (a) indicates the case for managing the mapped data on an overlay network, and the line (b) indicates in the case of managing pointer information of data on it without themselves.

Fig. 5 shows that the elapsed time for processing is increased by the increment of the target data in both cases. In specific, the increment rate of line (b) is significantly larger that of line (a). This difference between these lines is caused by the amount of processing elapsed time from sending the query to generating the results. In the case of line (b), the query sent to the overlay network for retrieval is independently distributed to each databases, and each database manages the data entity mapped to pointer information. In comparison with line (a), the divide between a query to a database and the collection of results from a database takes elapsed time for generating retrieval results.

IV. Considerations

This section discusses the characteristic and usage situations of the proposed model as a as a realistic system based on the evaluation results and model design.

A. Database Service

What follows is some discussion of functionally and serviceability.

Relationship management among pieces of data: Users map and manage data without reference to where data is located. Data is retrieved and generated across overlay networks. In other words, the proposed model allows relationships to be generated for pieces of data on distributed databases and then allows applications to use those relationships without moving data to a single database. In particular, some implementations of RDBMS database manage big data using a No SQL model. Service Data is often divided into some shards to retain read-write capability and improve response times. For data managed using No SQL, relationships between pieces of data and tables cannot be generated and managed unless the change of type of database. However, mapping all data to an overlay network enables the relationships to be generated and managed without unifying these distributed databases. Thus, our proposed model provides the integration of big data services effectively without database re-construction.

Availability as a database service: The proposed overlay network is composed of distributed databases that use a P2P platform. The amount of data and the number of component databases can be changed flexibility and dynamically: data and databases can be added, changed and removed. The evaluation results show that the quality of the proposed

 ISBN: 978-1-61208-318-6

model does not affect the number of distributed node. Thus, the overlay network provides a continuous and stable service by demand of its situations autonomous adjustment among component nodes for some event: load of node and outage from the overlay network caused by the increment of traffic on the overlay network. This method provides more availability than the current RDBMS model, making it more flexible and autonomous. Similar with No SQL, the proposed model adopts a P2P distributed model and it can be expected to provide similar service availability.

Usage in big-data situations : The proposed model enables the generation and management of relations among pieces of data without an ordinal model and allocations for big-data generated from SNSs and web services. Developer do not need to divide into new shard and they apply conventional relations of data to new services when service providers expand or improve service. Therefore, the proposed model is effective and useful for current big-data services.

B. Future Work

The above section clarified the characteristics of our proposed database service in comparison with the current RDBMS model in regards to serviceability, scalability, and availability. The model also provides important function for services. However, evaluation results indicate that the delay (lag) for transactions and response to service are urgent issues. The improvements provided by the implementation are limited by the distributed system. Therefore, future work will apply a management algorithm to the overlay network spaces for the replication or caching of distributed well-accessed data described in section IV-A to improve the implementation. Our proposed model should be feasible for use on real services.

V. Conclusion

We proposed and discussed a means for integrating databases that are diffused (distributed) throughout the Internet. The databases are integrated using peer-to-peer cloud computing technology to construct a large logical database space. All data in the distributed databases is integrally controlled by SQL regardless of the types of component databases (such as RDBMS, or KVS) and the size of the databases. These improvements make possible integrated platform technology for web services and new applications such as big-data services. However, the current implementation cannot be immediately applied to such services due to low performance during situations. The simulation suggests that the model can be used for big-data services and can be improved by adding function such as data caching, and replications. The model must be improved before next steps can be taken .

Acknowledgment

This research was partially supported by Strategic Information and Communication R&D Promotion Program (SCOPE) of the Ministry of Internal Affairs and Communications, Japan, Grant number:122304003 .

References

[1] T. Tsuchiya, H. Sawano, M. Lihan, H. Yoshinaga, and K. Koyanagi, “A Distributed Information Retrieval Manner Based on the Statistic Information for Ubiquitous Services”, Progress in Informatics No. 6, pp. 63–78, April 2009

[2] F. Chang et al., “Bigtable: A Distributed Storage System for Structured Data”, ACM Transactions on Computer Systems, Vol. 26 Issue 2, June 2008

[3] W. Vogels, “Eventually consistent”, Communications of the ACM Rural engineering development, Vol. 52 Issue 1, January 2009

[4] A. Lakshman, and P. Malik , “Cassandra: a Decentralized Structured Storage System”, ACM SIGOPS Operating Systems Review, Vol. 44 Issue 2, April 2010

[5] ”MongoDB”, 31 January 2014, <http://www.mongodb.org/>

[6] E. Meijer, and G. Bierman, “A co-Relational Model of Data for Large Shared Data Banks”, ACM Queue, Vol.9, Issue 3, pp. 1–19

[7] U. F. Minhas et al., “Elastic Scale-out for Partition-Based Database Systems”, Proc. of Int. Self-managing Database Systems, Washington, DC, April 2012

[8] H. Yoshinaga, T. Tsuchiya, and K. Koyanagi, “Coordinator Election Using the Object Model in P2P Networks”, 3rd International Workshop on Agents and Peer-to-Peer Computing, Springer Berlin/Heidelberg, Vol. 3601, pp. 161–172, New York, 2004

[9] G. M. Morton, “A Computer Oriented Geodetic Data Base and a New Technique in File Sequencing”, Technical Report, IBM Ltd., Ottawa, Canada, 1966

[10] W. Pugh, “Skip lists: a Probabilistic Alternative to Balanced trees”, Communications of ACM 33, pp. 668–676, June 1990

[11] I. Stoica, R. Morris, D. Karger, M. Kaashoek, and H. Balakrishnan, “Chord: A Scalable Peer-to-Peer Lookup Service for Internet Applications”, ACM SIGCOMM, pp. 149–160, San Diego, CA, 2001

[12] TPC-W, ”TPC Transaction Processing Performance Council”, 31 January 2014, <http://www.tpc.org/tpcw/>

 ISBN: 978-1-61208-318-6