Design and Applied Research of the Distributed Real-time Database in Smart Grid

Chunfeng Liu, Yan Jiang, Feng Zhao, Qiao Sun, Yang Zhang, Zhiqi Li, Chao Li

Beijing Guodiantong Network Technology Company Ltd.,

No.1 Hangfeng Road, Fengtai District, Beijing, 100070, China

{liuchunfeng, jiangyan, feng_zhao, sunqiao, zhangyang2, lizhiqi, lichao3}@sgepri.sgcc.com.cn

Abstract— **Real-time database systems combined with distributed architecture possess the capability to satisfy the timing constraints and preserve the data consistency while storing and processing the massive real-time data in the smart grid. In this thesis, we first review the current situation of the research on distributed real-time database. Then according to the characteristics of the electricity data in the smart grid, we design and implement a distributed real-time database. At present, this new distributed real-time database has been already applied into many business systems in State Grid Corporation of China. It presents good stability and accuracy, and provides a reliable real-time data support to the construction of smart grid in China.**

Keywords— **Distributed, real-time database, smart grid, typical applications, electric energy data acquire system**

I. INTRODUCTION

As the lifeblood of the national economy, electric power has a direct influence on the development of the country. With the characteristic of high volume of transactions, Modern electronic commerce applications and their related services cannot remain alive without the on-line support of computer systems, especially the database technology [1]. Currently, the power grid in China is transforming towards smart grid and the requirements of intellectualization and automation would bring about deeper integrations of various information and communication techniques with smart grid. Internet of Things, considered as the third revolution in the digital technology after the computer and Internet, is playing a significant and positive role in the construction of smart grid[2].

As we know, the numerous sensing devices of IoT are generating massive electricity information data every minute, which could be used in the accurate management, real-time control and scientific decision-making of the production, transmission and distribution of the electricity power. Obviously, in order to timely monitor and control the modern electric manufacturing facilities and process the sensed information, the massive data has a high real-time requirement. Usually, conventional database systems would not be applicable for the time-critical applications because of the lack of predictability and the poor real-time performance. Database systems, which have the distinct feature in satisfying time constraints associated with transactions are called real-time database. Both timing constraints and data consistency should be taken into consider in real-time database systems when scheduling the transactions so that they can be accomplished by their corresponding deadlines[3]. For instance, both the query and update on the acquisition data of smart electricity meters, mainly working for the calculation of the multistep electricity price and electricity increment, must satisfy not only the database consistency constraints but also the time constrain, processed within the given deadlines. Apart from the time constraints that are out of the demand of continuously data tracking, timing correctness requirements are also proposed due to the need to keep data available for the decision-making activities of the controlling systems[4]. Based on these situations, real-time databases are used in a wide range of applications in the power industry.

At the same time, the electricity data also has astounding quantities. In China, the scale of the data points in smart grid has now reached about thirty million and it will keep increasing as more and more electric terminal equipment and new energy technologies being involved in the smart grid. However, the traditional real-time database systems are usually centralized, running on a single computer system, which limits their capability of data storage and processing so that they can hardly meet the demands of the massive real-time data in capacity and efficiency. A distributed database system comprises a collection of nodes connected to each other via the communication network, in which every node contains a database system and work with each other based on some sort of agreements[5]. Compared with the centralized ones, the distributed database systems perform much better when dealing with the massive data. Thus, real-time database systems combined with a distributed architecture can make it a reality that allows us to satisfy the timing constraints and preserve the data consistency while storing and processing the massive real-time data in the smart grid.

In this thesis, we first review the current situation of the research on distributed real-time database. Then according to the characteristics of the electricity data in the smart grid, we design and implement a distributed real-time database. Finally, we analyse its typical applications in the State Grid Corporation of China.

II. Related Works

Distributed real-time database systems (DRTDBS), are generally defined as "a collection of multiple, logically interrelated databases distributed over a computer network where transactions have explicit timing constraints" [6]. As we see, there are some difficulties in satisfying the timing constrains that come from the multifarious real time activities in the distributed systems due to the logical consistency requirements of the database and also the distributed nature of the transactions.

There are some surveys, researches and other papers in this field, including the topics about the real-time data compression, commit protocols, priority assignment, concurrency control, memory optimization for data processing, dynamic replication for periodic transactions and so on. Bristol E. H. first proposed the data compression algorithm called swinging door algorithm, in which the recording limit was based on the slope of the line between the current measured values and the previously recorded values[7]. Watson M. J. and Liakopoulos A. et al. described three ways of data compression, including piecewise linear functional approximation, application of a data transform and the discarding of the vector quantization and the insignificant transform coefficients, each of the method were used to compress large sets of the real process data[8]. Haritsa et al. mainly focused on the problems of creating a priority ordering for the transactions distinguished by both value and deadlines[9]. Dogdu Erdogan and Ozsoyoglu Gultekin presented the new priority assignment and load control policies for the repeating real-time transactions[10]. Ramesh Gupta et al. did an in-depth research in which they evaluated the relative performance of various commit protocols, such as PA, PC, 2PC and 3PC[11]. Lam et al. proposed a protocol integrating transaction commitment and concurrency control for the real-time transaction, which is called the deadline-driven conflict resolution (DDCR) [12]. Ramamritham K. and Sen R. applied a novel storage model, ID based storage, which conspicuously reduces the storage costs[13]. Udai S. and Manoj M. et al. introduced a memory efficient fast distributed real time commit protocol (MEFCP) based on a new locking scheme and write operation which was divided into update and blind write[14]. Yuan Wei and Andrew A. Aslinger et al. presented two dynamic replication control algorithms designed for medium and large scale distributed real-time database systems, which dynamically determine where and how often the replicas are replicated[15].

III. Design of DRTDBS

A. System Requirements

In this thesis, our primary objective is to design and implement a distributed real-time system towards the smart grid, which combines the real-time database system with the technologies of cloud computing. Generally, the electricity data generated during the production and management process in the smart grid has the following key characteristics including the large amount, high real-time capability, weak relationship, fixed structure and vigoroso randomicity. According to the above data characteristics and some specific application systems in the smart grid, our new system should meet the following requirements:

1) Basic Functions: The basic functions of a real-time database system should be included, such as the fast accessing, processing and management of the data, multi-task and multi-channel parallel, effective recovery and reloading.

2) Capacity: Satisfies the demand of large capacity (20 million measuring points).

3) Transparency: Provides the transparent user interface so that users needn't care about the logical partitioning and physical location of the data.

4) Scalability: Possesses high extensibility and on-demand scalability.

5) Load balance: Automatically balance the workload of each real-time database site.

6) Reliability: Possesses high reliability that allows the system to run properly despite the corruption of the hardware in some extent.

7) Standard interface: Provides the standard interfaces for data accessing.

8) Peripheral support: Provides abundant peripheral tools, such as the statement and the publication on the Web.

B. System Architecture

1) Physical Architecture

The system consists of the following physical parts, namely global scheduling severs, proxy severs, database servers and cloud storage devices.

Figure 1 shows the physical architecture of the distributed real-time system:

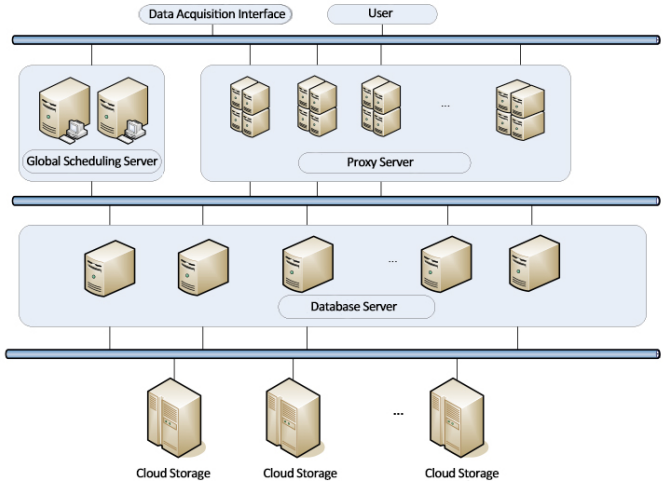

Figure 1. Physical architecture of the distributed real-time database

- ***Global Scheduling Servers*:** they deploy MYSQL and keep records of the global information based on the hot standby technique.
- ***Proxy Servers:*** they are the real access points of all the client connections and always remain connected to the database servers.

- ***Database Servers:*** as the logical database nodes, they deploy both real-time and historical services, regarded as the core of data processing in the system.
- ***Cloud Storage Devices:*** they save historical data files, providing efficient access interfaces to import, read and modify the shared historical data files and they also implement the security mechanism like redundancy backup to improve the reliability of the system.
- Users and the data acquisition ports are connected to the real-time database systems via the distributed network.

2) Logical Architecture

The internal core components of the system mainly consist of three kernel layers, including the scheduling layer, the real-time database layer and the cloud storage layer.

Figure 2 shows the logical architecture of the distributed real-time database:

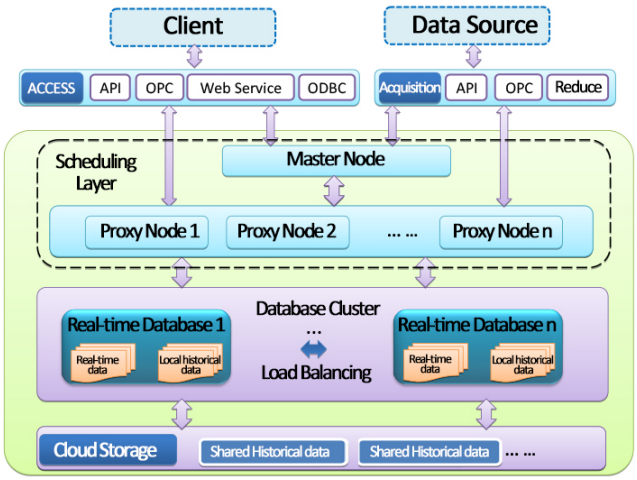

Figure 2. Logical architecture of the distributed real-time database

- ***Scheduling Layer:*** This Layer is comprised of master node and several proxy nodes. All the proxy nodes are deployed on a distributed architecture, possessing the capability to process the task requests raised from clients in parallel. Specifically, when the master node receives the connection request from the client, it will assign the proper proxy node to establish the connection according to a certain strategy. Then, the proxy node will split or merge the task and select the appropriate database node to handle this task. The quantity of the proxy nodes is flexible according to the scale of the system, ensuring the efficiency of the scheduling layer. Meanwhile, load balancing is achieved, mainly for the network, memory, hard disk, CPU and other hardware. To guarantee the high reliability of the system, all the proxy nodes are based on a peer-to-peer model so that they could be the redundance of each other. Hot standby technique is applied in the security mechanism of the master node.
- ***Real-time Database Layer:*** This layer is comprised of several database nodes deployed on a distributed architecture. Based on a nonequivalent model, all the database nodes work together to undertake the whole real-time and historical task, providing the basic features of real-time database and communicating with the cloud storage devices. Load balancing of all the database nodes is achieved, mainly for the network, memory, hard disk, CPU and other hardware. In addition, online redundancy is also required to achieve the high reliability, allowing the real-time database system to operate properly despite the corruption of the hardware in some extent.
- ***Cloud Storage Layer:*** This layer provides the function to store and access data for the real-time database layer. The right to read and write files, Multi-replica online redundancy, management of load balance and efficient access interfaces are all required in this layer.

C. System Services

There are seven primary services in the distributed real-time database system, including node service, scheduling service, proxy service, real-time service, historical service, log service and security service.

Figure 3 shows the service structure based on the different kinds of nodes.

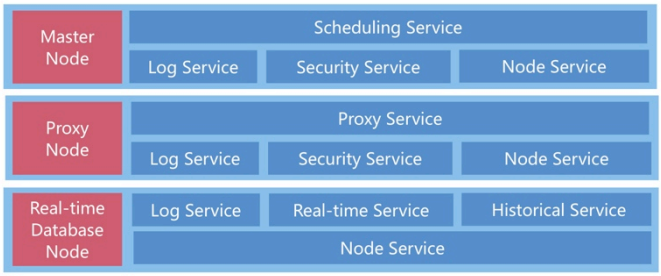

Figure 3. Service structure of the distributed real-time system

Figure 4 shows the calling relationship of each service. Here each dotted arrow stands for a logical call and the arrow points to the interface provider.

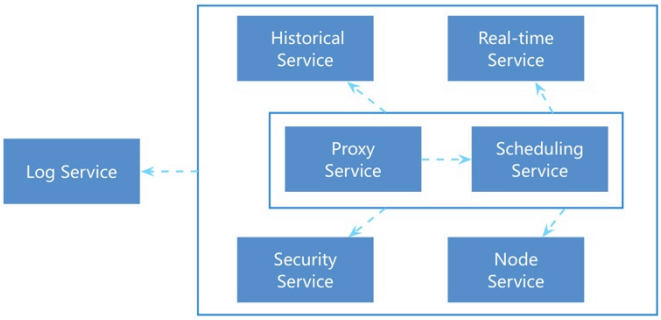

Figure 4. Calling relationship of each service

1) Node Service: When the node service start on servers including scheduling severs, proxy servers and real-time database servers, every node in the system will be active. The master node monitors the statuses of all the nodes and maintains a table of node information to ensure the normal operation of the functional modules on every node. Node Service consists of four parts, namely node connection, node management, node service management, and node information table maintenance.

Figure 5 and Figure 6 show the key workflow in node service:

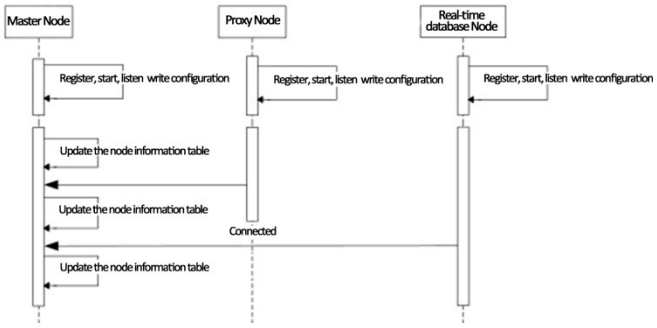

Figure 5. Initialization of the node service

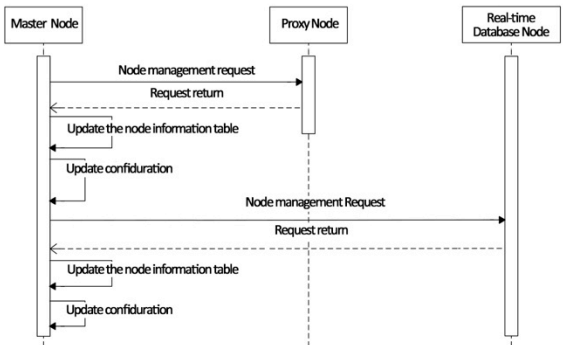

Figure 6. Management of the node service

2) Scheduling Service： Scheduling service is deployed on the master node, which undertakes the task for connection and scheduling between client and proxy nodes. It consists of five parts, namely balance scheduling of proxy node, balance scheduling of the measuring points in real-time database, maintenance of the connection information management table, verification of users' logon rights as well as the system authorization information.

3) Proxy Service： Proxy service is mainly responsible for the connection management of the real-time database, the merge and split of the tasks for data acquisition and client accessing. Master node will select proper proxy node to connect to the client and the proxy node will process the client request according to the strategy of connection scheduling and task allocation. This service mainly includes three functional modules, namely connection management module, task allocation module and measuring point service module.

Figure 7 and Figure 8 show the key workflow of connection management module in proxy service:

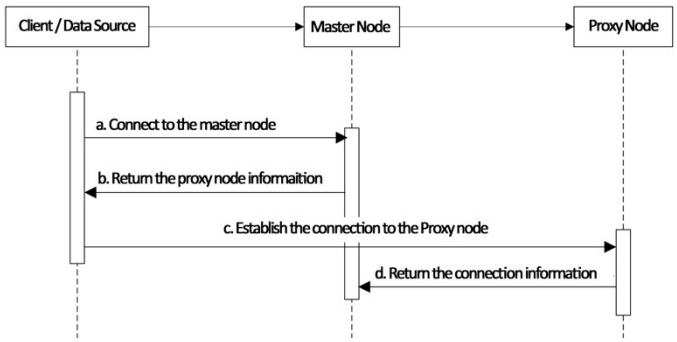

Figure 7. Establish the connection between the client and proxy node

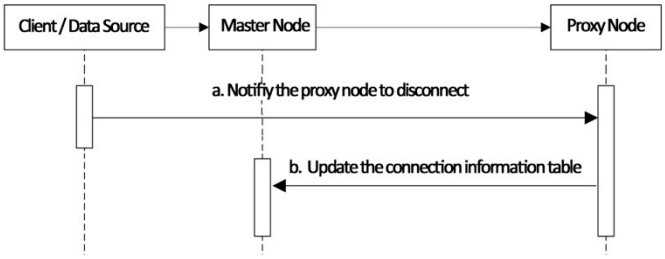

Figure 8. Client is disconnected from the proxy node

4) Real-time Service： Real-time service is deployed on the real-time database servers. It provides the read/write interface of the real-time data for the proxy service and completes the specify data manipulation request assigned by the proxy service, including writing the real-time data to the database as the proxy service appoints and submitting the real-time data from the database to the proxy service. There can be several real-time services on a single server, each of which is related to the read/write operations of real-time data in a measuring points set.

5) Historical Service： Historical service provides the read/write interface for the historical data, enabling the read and write operations towards historical data that is stored in the local disk or shared storage. It is deployed on every real-time database server, transferring the snapshot of all the measuring points in the current database server to the historical archive files. At present, the historical services establish their own queue of the historical archive files in the cloud storage so that the real-time data files will be directly stored in the cloud storage, which effectively improves the data usability and security.

Specifically, every historical archive file is made up of massive data blocks. The system allocates a data block with fixed size (such as 1k, 8k, 256k) in the memory for each measuring points where the real-time data will be written. When the block is full, it will be transferred to the historical archive file in cloud storage. Limited by the size of the memory, the block cannot be too large, especially when the amount of measuring points is huge. For example, given a system with 10 million measuring points, if each block takes up the space of 4k, the size of the memory should be 10,000,000 * 4k = 40G, which would not be accomplished at present. However, the smaller the block is, the more I/O operation is needed and the lower efficiency the system will have due to the poor performance of cloud storage in processing the small blocks. To solve this problem, the system aggregate the small blocks to a bigger one in the memory before they were written to the cloud storage. The size of the aggregated block depends on the feature of different cloud storage. Thus, we observe the system performance in different condition to find the best size of the block.

Table 1 shows part of the testing result (due to the protection of trade secret) based on the different sizes of the aggregated data blocks. In this test, 50 blocks are sequentially written into the cloud storage in parallel.

Figure 9 shows a line chart for the testing result from three aspects.

TABLE 1. SYSTEM PERFORMANCE BASED ON DIFFERENT SIZES OF BLOCKS

Size	256k	128k	64K	32K	16k	8k
Iops	221.9737	471.8816	966.2155	1906.6997	3880.8744	7704.0671
MBps(MB)	55.4934	58.9852	60.3885	60.5844	60.6387	60.1880
CPU(%)	3.0824	3.3858	3.9872	4.1145	9.9065	17.9335

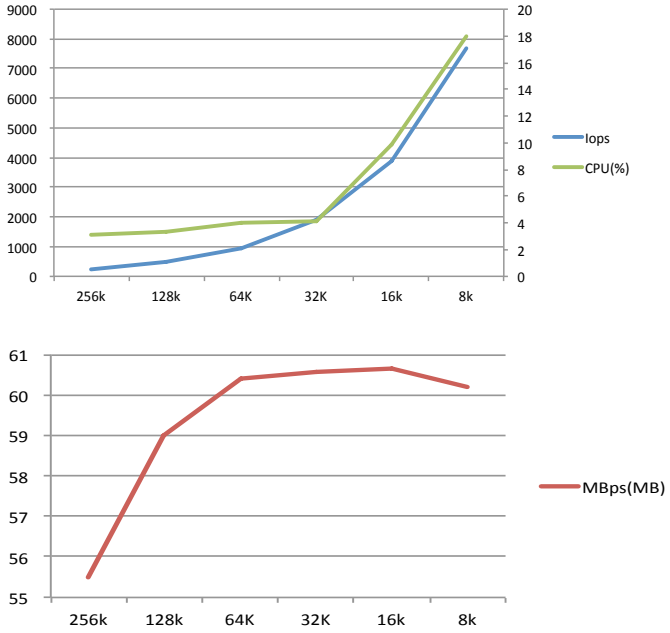

Figure 9. System performance based on different sizes of blocks

6) Log Service: Log service is able to keep track of the various events that generated in the operation of the system. When the system malfunctions, the administrator could find where the problems reside knowing roughly about the running conditions of the system.

7) Security Service: Security service is provided to ensure a secure and stable operation of the system. It also provided the management of user permissions, which guarantee that all the users can only operate within the purview of authorization.

IV. TYPICAL APPLICATIONS OF DRTDBS

As the development of the smart grid, the distributed real-time database systems are being widely used due to their high capability to store and process the massive real-time data. In this section we will introduce some typical applications of our distributed real-time database system on the smart grid in China.

A. Production real-time supervisory system of Power Generation Group

Production real-time supervisory system is a real-time management system for the Power Generation Group that can supervise the real-time production data of the plants under its jurisdiction, in order to improve the operation and supervision efficiency for the group headquarters. The system uses our distributed real-time database to collect the various data that generated from the production process of the plants and put them into the central database, realizing the control and management for the distributed production process and helping the administrators to master the operation status of all the plants. This system has not only improved the operation efficiency and economic benefits but also enhanced the security, reliability and facility utilization of the Power Generation Group.

B. Power Forecast System of Wind Power

Due to the volatility, intermittence and low-energy-density of the wind, wind power is also volatile. When a large-scale wind power station connected to the power grid, the volatility of wind power will have a side effect on the power balance and frequency adjustment on the power grid. Forecasting for the wind power will help the power-dispatching department to arrangement the dispatching plan in advance, which ensures the security of the power grid. This forecast system consists of our distributed real-time database, which is used to collect and analyse the high-precision data from the wind power stations. Based on the real-time data, the system can forecast the wind power in the coming 0~72 hours and update the result every 15 minutes. The power forecast system also provides a friendly user interface and it is very helpful for the management of wind power station.

C. Electric Energy Data Acquire System

Electric energy data acquire system collects, processes and monitors the electricity information to realize the following functions, such as power quality monitoring, online analysis and management of electricity, calculation of the multistep electricity price and so on. At present, the amount of smart electricity meters in smart grid has reached 30 million and the sampling frequency has increased to every 15 minutes once. Given that there are 5 values in each meter, the amount of the data can be 30,000,000*5*4*24 = 14,400,000,000 in one day. The conventional database systems have particularly poor performance due to the I/O bottleneck. Based on this situation, our distributed real-time database is involved to store and process the massive real-time data. The system supports 1000 connections at the same time and is able to complete the collection, storage and computing of the data from 30 million smart electricity meters every 15 minutes. It provides safe, reliable and efficient data acquisition and processing capability to the electric energy data acquire system.

V. CONCLUSIONS

Today, this new distributed real-time database has been already applied into many business systems in State Grid Corporation of China (SGCC). The distributed real-time database system possesses the ability to store and process the massive real-time data. It presents good stability and accuracy, and provides a reliable real-time data support. However, extensive research still can be done in many related field, such as the mixed storage strategy between RAM, SSD and HDD, multi-core parallel processing, in-memory computing and so on.

Technology is changing every day, with new problems continually arising and new requirements constantly emerging.

SGCC will keep promoting the research and industrial development in smart grid and make further contributions to building up the world-class power grid and world-class utility.

REFERENCES

[1] Aldarmi S A. Real-time database systems: concepts and design[J]. REPORT-UNIVERSITY OF YORK DEPARTMENT OF COMPUTER SCIENCE YCS, 1998.

[2] Liu J, Li X, Chen X, et al. Applications of internet of things on smart grid in china[C]. Advanced Communication Technology (ICACT), 2011 13th International Conference on. IEEE, 2011: 13-17.

[3] Son S H, Poris M S, Iannacone C C. Implementing a Distributed Real-Time Database Manager[C]. DASFAA. 1991: 51-60.

[4] Ramamritham K, Son S H, Dipippo L C. Real-time databases and data services[J]. Real-Time Systems, 2004, 28(2-3): 179-215.

[5] Garcia-Molina H, Lindsay B. Research directions for distributed databases[J]. ACM SIGMOD Record, 1990, 19(4): 98-103.

[6] Shanker U, Misra M, Sarje A K. Distributed real time database systems: background and literature review[J]. Distributed and Parallel Databases, 2008, 23(2): 127-149.

[7] Bristol E H. Swinging door trending: Adaptive trend recording[C]. ISA National Conference Proceedings. 1990, 45: 749-753.

[8] Bristol E H. Swinging door trending: Adaptive trend recording[C]. ISA National Conference Proceedings. 1990, 45: 749-753.

[9] Haritsa J R, Livny M, Carey M J. Earliest deadline scheduling for real-time database systems[C]. Real-Time Systems Symposium, 1991. Proceedings., Twelfth. IEEE, 1991: 232-242.

[10] Doğdu E, Özsoyoğlu G. Real-time transactions with execution histories: priority assignment and load control[C]. Proceedings of the sixth international conference on Information and knowledge management. ACM, 1997: 301-308.

[11] Gupta R, Haritsa J, Ramamritham K, et al. Commit processing in distributed real-time database systems[C]. Real-Time Systems Symposium, 1996., 17th IEEE. IEEE, 1996: 220-229.

[12] Lam K, Cao J, Pang C, et al. Resolving conflicts with committing transactions in distributed real-time databases[C]. Engineering of Complex Computer Systems, 1997. Proceedings., Third IEEE International Conference on. IEEE, 1997: 49-58.

[13] Ramamritham K, Sen R. DELite: database support for embedded lightweight devices[C]. Proceedings of the 4th ACM international conference on Embedded software. ACM, 2004: 3-4.

[14] Shanker U, Misra M, Sarje A K. A memory efficient fast distributed real time commit protocol[M]. Distributed Computing–IWDC 2005. Springer Berlin Heidelberg, 2005: 500-505.

[15] Wei Y, Aslinger A, Son S H, et al. ORDER: a dynamic replication algorithm for periodic transactions in distributed real-time databases[C]. 10th International Conference on Real-Time and Embedded Computing Systems and Applications (RTCSA 2004). 2004.

Chunfeng Liu was born in Liao Ning Province, China, on July 31th, 1981. He was graduated from Beijing University of Posts and Telecommunications and got a master's degree of electronic and communication engineering. In the past 10 years, He was mainly engaged in the research of electric power informationization.

Yan Jiang was born in Shan Dong Province, China, on March 4th, 1989. She was graduated from Renmin University of China and got a masters' degree of software engineering in 2013. Now she is engaged with the research of cloud computing in smart grid.

Feng Zhao was born in He Bei province, China, on October 14th, 1980. He was graduated from Beijing University of Posts and Telecommunications and got a master's degree of automation. Now he is a Ph.D. student in China Electric Power Research Institute, majoring in Electrical Engineering and Automation.