Sensors & Transducers, Vol. 160, Issue 12, December 2013, pp. 607-611

www.sensorsportal.com

Sensors & Transducers

© 2013 by IFSA

http://www.sensorsportal.com

Reliable Power Monitoring System Based on Real-Time and Fault-Tolerant CORBA

Hongjian LIAO, Yawu XU

Guangzhou Higher Education Mega Center, Guangzhou University,
No. 230 Outer Ring Road, Guangzhou, 510006, China
Tel.: +86-15902073512, fax: +86-20-39366406
E-mail: LiaoHongJian@gzhu.edu.cn

Received: 19 October 2013 /Accepted: 22 November 2013 /Published: 30 December 2013

Abstract: To improve the real-time and fault-tolerant performance of data transmission in sensor-based distributed power monitoring system, the principle of event service in common object request broker architecture was analyzed, and the "subscribe/publish" mechanism was realized by improving and extending event service in push model in order to reinforce real-time performance. And a semi-active copy method of fault-tolerant was proposed under push model. On this basis, the model of real-time and fault-tolerance middleware in the power monitoring system was designed and realized with these methods. This model was used in the network communication layers of multiple power monitoring systems, and the results shows that the model is safe and reliable with real-time response. *Copyright © 2013 IFSA.*

Keywords: Power monitoring, CORBA middleware, Event service, Real-time, Fault-tolerance, SCADA.

1. Introduction

In the distributed power monitoring system, it is required to transmit the various equipment data collected on site inerrably to remote monitoring system in real time. This is a typical "critical mission" [1]. For a distributed system used for "critical mission", the user is concerned about its reliability, i.e. the correct success rate of the results obtained in task accomplishment. Real-time and fault-tolerant properties are effective means for ensuring the reliability, and neither of the two can be dispensed. So they shall be studied as a whole. Past five years, many scholars have studied the real-time or fault-tolerant technology in this kind of environment [2-6], but quite a few papers concerned about their application as a whole in the power monitoring system. This paper, with the object-oriented distributed middleware technology, designs and implements system architecture of real-time fault-tolerant middleware, and expounds the specific method for applying it in constructing reliable power monitoring system.

2. Data Transmission Requirements and Model Selection

Because the equipment platforms and data standards vary in the distributed power monitoring system, the following specific conditions must be met to realize real-time and reliable transmission of the data collected:

1) Real-time property, i.e., the basic data obtained shall be transmitted in time to the algorithm module of the monitor end for processing and the processed data shall be transmitted in real time to the display terminal for display;

2) Non-coupling property, i.e., the different SCADA (Supervisory Control and Data Acquisition) data and different data providers/consumers can be connected dynamically, realizing "plug and play";

3) Filtering mechanism, i.e., different data are transmitted to different algorithm modules depending on the demand;

4) Fault tolerance, i.e., the assigned task can be performed correctly when software or hardware failure occurs.

In such environment, the better way is to adopt distributed middleware technology to reduce development cycle and to improve the reliability, availability and maintainability.

Common object request broker architecture (CORBA) [7] is an object-oriented distributed middleware technology launched by Object Management Group (OMG). It defines a distributed structure with open software bus. COABA provides with object services such as name service, safety service and event service to resolve the various problems in distributed system applications.

In distributed calculation, to avoid the idle brought by synchronous communication, it is necessary to choose an asynchronous communication model for the distributed system to realize overlap of calculation and communication. CORBA event service is a relatively ideal selection. It provides with a standard object model, which, through object broker, provides with an event channel between the anonymous event suppliers and consumers, realizing flexible asynchronous or group communication mechanism and allowing the object to register or revoke designated events dynamically [8]. Event service separates the data suppliers from the data consumers. For the multiple types of data that the data suppliers may publish, the data transmission model only picks up and transmits the data in which the consumers are interested instead of all the data. So a "Subscribe/Publish" mechanism is provided [9].

3. Real-Time and Fault-Tolerant Event Service

3.1. Event Service in CORBA

In CORBA standard, event service classifies the objects under 2 types: suppliers and consumers. The suppliers generate events while the consumers dispose events. The event is transmitted by virtue of an object called "event channel". Event channel is a standard CORBA object, which is resident on the object request broker (ORB). As an asynchronous communication medium among multiple "supplier" objects and multiple "consumer" objects, "event channel" is responsible to transmit the event from the "supplier" object to the "consumer" object. After event channel is introduced, the relationship between the event supplier and consumer is loose coupling relationship. They can not see each other, but only make contact with the event channel. An event channel can have any number of suppliers and consumers in contact with it. The structure of CORBA event service is shown in Fig. 1.

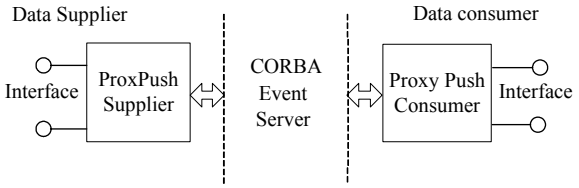

Fig. 1. Structure diagram of event service.

Event channel is the core of event service. Depending on the initialization mode of the event channel, it supports 3 types of data transmission models: Push Model, Pull Model and Push-Pull Model.

3.2. Real-Time Event Service Design

1) Push model: Subscribe/Publish model is used for real-time data transmission in power monitoring system. So Push Model is adopted in this paper. In Push type communication, the data supplier pushes the data actively to the event channel, and the event channel pushes the event data to the consumer. As compared with Pull Model, when an event appears, the active notification mode adopted by the data supplier in Push Model avoids the expenditure of Polling. Therefore, Push Model is most economical in communication resources and has the highest efficiency among the 3 modes. The communication mode of Push Model is shown in Fig. 2.

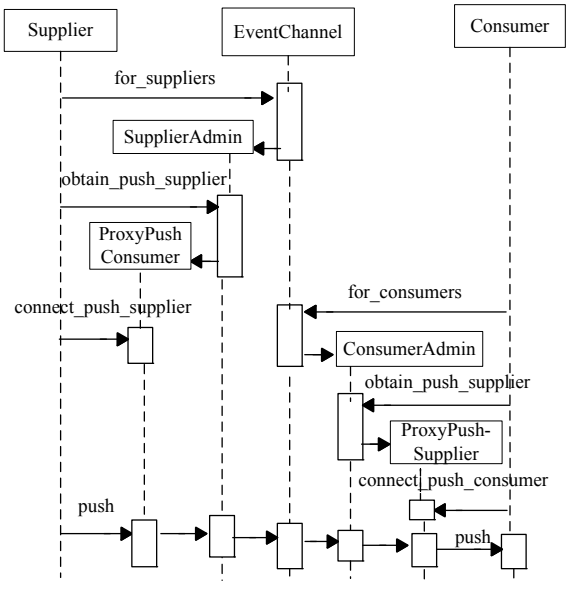

Fig. 2. Transmission process of push type event data.

- At the event Push-Supplier end:

The event Push-Supplier, through the bind operation of denomination service, gives the channel name, obtains the object reference of the Event Channel, and obtains an event Proxy Push Consumer from the event channel created;

The event Push-Supplier is connected through the event Proxy Push Consumer to an event channel instance.

- At the event Push-Consumer end:

The event Push-Consumer gives the same channel name as created by the event Push-Supplier, calls bind operation, and obtains the object reference of the event channel;

The interface for calling the event channel obtains event Proxy Push Supplier. Similar to the Push-Supplier end, the event Push-Consumer is connected through the event Proxy Push Supplier to the event channel instance.

To improve the reliability of the transmission of push type event data, the event data are cached on the Event Channel.

Definition of Push Supplier interface in Push Model:

```
interface PushSupplier{
void disconnect-push-supplier();
};
```

Definition of Push Consumer interface:

```
interface PushConsumer{
void push(in any data) raises(Disconnected);
void disconnect-push-consumer();
};
```

2) Realization of information subscription: In power monitoring system, several remote monitoring systems are often monitoring the project operated by a same SCADA in the same time. The information monitored by different Remotes may vary. To improve the efficiency, variables filtering mechanism is added. The event service maintains a Supplier - Variables to be monitored - Consumer Linked List for recording which variables of which supplier is subscribed by a certain consumer. The structure of the linked list is shown in Fig. 3.

Event Proxy Push Supplier is resident at Remote end. If Remote needs to monitor a certain project are being operated by SCADA, it firstly obtains the object reference of a Proxy Push Supplier and then registers the project variables to be monitored with the event service through the Proxy Push Supplier. If the event service confirms that this project is operating, it will subscribe the information of this project, add the supplier, variables to be monitored and consumer data into the corresponding Supplier - Variables to be Monitored - Consumer Linked List, and return success. If this project is not operated or has been quitted, it will return registration failure information to Remote.

Event Proxy Push Consumer is also resident at SCADA end. If Remote needs to monitor this SCADA, the SCADA firstly obtains the object reference of the event Proxy Push Consumer and then calls continuously the push () interface encapsulated by the event Proxy Push Consumer to push the event information to the event service. The event service will unpack the event information, inquire about the Supplier - Variables to be Monitored - Consumer Linked List, and transmit the event information to the corresponding consumer by calling its interface of push().

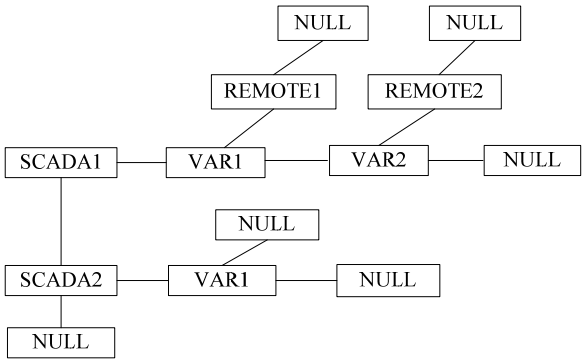

Fig. 3. Structure diagram of information subscription linked list.

The push type event service in CORBA meets the real-time requirements for data transmission. On this basis, the design of fault-tolerant CORBA middleware under the Push model is further discussed.

3.3. Fault-Tolerant Event Service Design

1) Main method: The current fault-tolerant CORBA can be realized through 3 modes: integrated mode, interception mode and service mode. Among these, service mode is to define a group of object and interface as CORBA service to provide the object fault-tolerance with strategy and mechanism. This paper uses event service mode, and realizes with object replication technology.

Object replication technology realizes redundancy through object replication, i.e. one object possesses multiple replicas, which constitute an object group and show to the outside as a whole. Its object reference is called IOGR (Interoperable Object Group Reference) [10]. The consumer uses IOGR to originate request to the object replicas without necessity to know the existence of object group or invalidation of the object group member. The realization of fault tolerance refers to the fact that when an object has replicas on multiple knots (or machines), the object is useable when one of the replicas is useable. The fault-tolerance structure of service mode is shown in Fig. 4.

The replication mode is classified under active replication and passive replication. In active replication, all the replicas of the object are equivalent in identification and respond to the request

of the consumer in the same time, and are transparent to the consumer when the replicas are disabled. The disadvantage is that only one of the responses of all the replicas is accepted in one time and all the others are discarded, wasting large quantity of network resources. As for primary-backup replication, supposing one replica is primary replica and all the others are backup replicas, only the primary replica executes the request of the consumer and transmits the necessary information to the backup replicas. Its disadvantage is that, when the primary replica is disabled, the other replicas need longer time to recover.

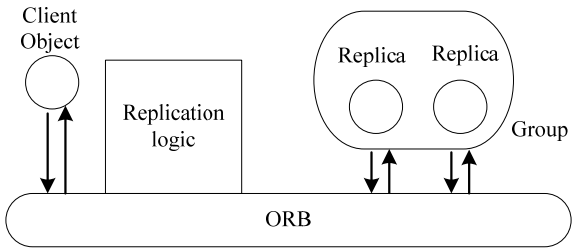

Fig. 4. Fault-tolerance structure of service mode.

Considering the advantages and disadvantages of the two replication types, this system adopts semi-active replication method: In each time, only one event service is allowed to respond to the request of the consumer. That is to say, in each time when SCADA collects new data and needs to publish them, it ties randomly to one of the event services and distributes the data through this event service. The identity of each replica is equal; while the object of event service responding to the request of consumer may be different from time to time, avoiding the disadvantage that the load at a certain knot is excessively high.

2) Maintenance of object consistency: Object consistency means that the status information of all the event services should be kept real-time consistent in the service operation process. In this system, maintaining the event service status information is a 3D linked list that describes the relationship of supplier - variables to be monitored - consumer. The object consistency maintenance is to keep the real-time consistency of the 3D linked lists maintained by the various objects. Therefore, the event service object shall provide with an interface to synchronize the various replicas. The IDL is as follows:

```
Interface EventService{
void push();
bool Synchronize();
};
```

When the event service starts, this replica and the other replicas are synchronized through Synchronize.

4. Realization of Reliable Power Monitoring System

4.1. Realization Method

As for the development platform, the Visibroker for C++ of Borland is used. Visibroker provides with a complete development environment for operation of CORBA. Based on this environment, the user can develop open, flexible and interoperable CORBA applications.

This system divides the whole monitoring environment into 3 layers from bottom to top, as shown in Fig. 5. 1) The first layer is the SCADA that accomplishes on-site data acquisition, including PLC and LCU equipment acquisition data and command transmission. 2) The second layer is ORB flexible bus layer, which is used mainly for resource sharing and system interconnection, forming a uniform distributed service platform. On this basis, RF-TF CORBA middleware is realized. 3) The third layer is the internal realization layer of the various application sub-systems, including modules such as monitoring, alarm and information publication.

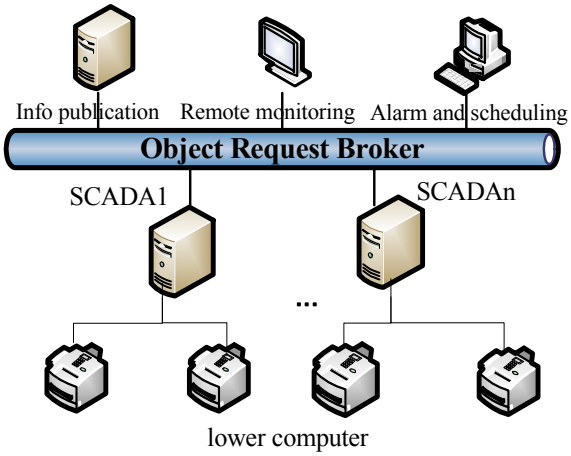

Fig. 5. Topological structure of power monitoring system.

4.2. Structure of Power Monitoring System Model Based on RT-FT CORBA

The flow of the system model is shown in Fig. 6. Once the information acquisition layer obtains data, it will push data to the event channel immediately. Then the event channel pushes data to the various algorithm modules on the basis of the information subscription linked list. The event service, through the fault-tolerance treatment of CORBA middleware, ensures that the data transmission is not affected by the abnormity.

The fault-tolerant field is located in the upper layer of ORB, and is composed of sequence manager, dynamic scheduler, fault detector, replication manager, etc. They jointly provide the user with sequential and reliable data service.

Among them, the replication manager is responsible for creating the object group, managing the object group members, and creating corresponding IOGR for returning to the application system for publication. Dynamic scheduler and sequence manager determine the queuing sequence of the new tasks in the corresponding server scheduling queue. This provides with a basic condition for the replication consistency of the system.

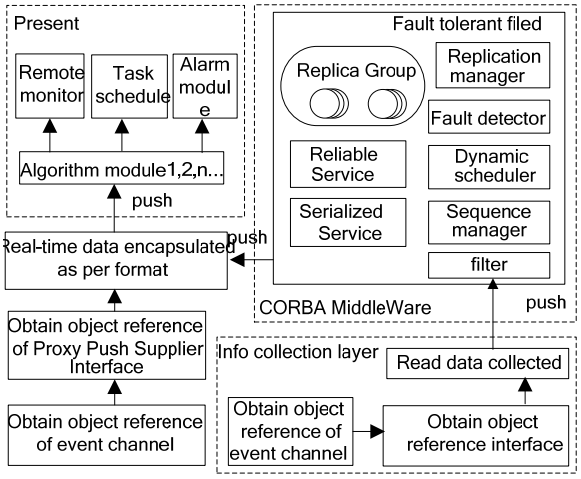

Fig. 6. RT-FT power monitoring data transmission model.

Because Push Model adopts an asynchronous communication mode, the whole data transmission process is non-blocking. The parallel calculation of the consumer end algorithm module reduces the unnecessary waiting time, realizes the overlap of calculation and data communication, and improves the system performance.

5. Conclusion

Application of CORBA in the field of power monitoring system can resolve its complicated isomerism problem and reduce the difficulty of network application development. Adoption of real-time fault-tolerant event service provides with a better approach for realizing reliable distributed power monitoring system. The GZX200 monitoring system software developed on the basis of this technology has been used successfully in several power stations such as Guangzhou Energy Accumulation Hydropower Station and Huangpu Power Plant. The RF-TF CORBA model proposed in this paper is applicable to the various network communications in power system. Its design idea can be used by those in the same trade for reference.

References

[1]. I. Calvo, L. Almeida, A. Noguero, M. Marcos, Supporting a reconfigurable real-time service-oriented middleware with FTT-CORBA, in *Proceedings of the IEEE Conference on Emerging Technologies and Factory Automation (ETFA)*, September 2010, pp. 1-8.

[2]. T. Demirci, A. Kalaycioglu, D. Kucuk, O. Salor, M. Guderl, S. Pakhuylu, Nationwide real-time monitoring system for electrical quantities and power quality of the electricity transmission system, *Generation, Transmission & Distribution, IET*, Vol. 5, No. 5, May 2011, pp. 540-550.

[3]. X. H. Lai, J. Y. Le, Power system simulation platform based on CORBA technology, in *Proceedings of the International Conference on Electronic and Mechanical Engineering and Information Technology (EMEIT)*, Vol. 4, No. 19, August 2011, pp. 1939–1941.

[4]. J. H. Ko, C. W. Park, K. S. Yoon, A review of network technologies in power monitoring systems in Korea, in *Proceedings of the IEEE Conference on Transmission & Distribution & Exposition: Asia and Pacific*, October 2009, pp. 1–4.

[5]. K. Y. Ying, S. Premrudeepreechacharn, A power quality monitoring system for real-time detection of power fluctuations, in *Proceedings of the 40th North American Power Symposium*, September 2008, pp.1-5.

[6]. K. K. Tae, S. K. Jin, H. K. Chan, J. K. Hee, B. K. Lee, Development of a new OS-LESS power monitoring system, in *Proceedings of the 29th International Telecommunications Energy Conference (INTELEC)*, October 2007, pp. 802-806.

[7]. P. Felber, P. Narasimhan, Experiences, strategies, and challenges in building fault-tolerant CORBA systems, *IEEE Transactions on Computers*, Vol. 53, No. 5, May 2004, pp.497-511.

[8]. A. Kumar, A fault-tolerant system for Java/CORBA objects, in *Proceedings of the IEEE International Symposium on Parallel and Distributed Processing*, April 2008, pp. 1-8.

[9]. S. B. Pallickara, H. G. Fox, Fault-tolerant reliable delivery of messages in distributed publish/subscribe systems", in *Proceedings of the 4th International Conference on Autonomic Computing (ICAC)*, June 2007, pp. 19.

[10]. D. Selvakumar, C. Rebeiro, R. Pitchiah, SCADA with fault tolerant CORBA on fault tolerant LANE ATM, in *Proceedings of the 19th IEEE International Conference on Parallel and Distributed Processing Symposium*, April 2005, pp. 8.

2013 Copyright ©, International Frequency Sensor Association (IFSA). All rights reserved.
(http://www.sensorsportal.com)