Real-time video transmission system based on embedded middle-ware TAO

Jiangyan Sun[1*], Xiaoqiang Jia[2]

[1]*Modern Education Technology Center, Xi'an Normal University, Xi'an, 710077, Shaanxi, China*

[2]*School of Mathematics and Information Science, Institute of Applied Mathematics, Weinan Normal University, Weinan 714000, Shaanxi, China*

Received 1 June 2014, www.cmnt.lv

Abstract

Analysis of the application of real-time video transmission system in the cluster operation platform, use thin clients Model, introduce the CORBA middleware, designed a program tailored to ACE + TAO dynamic program, development program as far as possible use smaller memory and smaller storage space, Combined with embedded real-time Middleware technology to construct distributed heterogeneous video transmission system. In order to verify the feasibility of the scheme and real-time, CORBA A/V services and CORBA Naming Service are combined with, and the real-time video transmission system of Embedded Middle-ware Based on TAO is realized. The system is composed of client and server, the client is composed of two parts and divided into sender and receiver according to the role, Use MFC and MiniGUI to develop user interface, run on Windows system and Linux system, the server is in VxWorks system for data forwarding function. With a 100Mbit/s bandwidth LAN environment, video of the subscription, the release, playing, pausing function has been realized by testing. In the aspect of real-time, video data, from collecting to playing with delaying in 150ms. Bandwidth in 352×288 pixel single collection of pictures, in case of 32 frames per second, the bandwidth consumption occupies only 500-540kbps, and realizes the low bandwidth consumption.

Keywords: TAO, heterogeneous platform, video transmission, MFC, the thin client

1 Introduction

With the rapid development of embedded system and network, especially in embedded operating system, the embedded chip is becoming more and more mature, which is moving into the direction of distributed embedded system. Embedded real-time communication network system has been widely used in military industry and other fields. At the same time in the field of military and communication, weapons and equipment are being developed to the direction of intelligent, distributed, and information. A new generation of weapon platform between weapons, each module between weapons equipment and the information system, especially between the command and control center and the weapon terminal high-speed information exchange ability, information integration and integration of high-speed real-time processing ability was put forward higher requirements. However, in the weapon equipment system, not only the need for real-time communication between all kinds of embedded devices, and various devices are often based on heterogeneous hardware and software.

In recent years, the combination of middle-ware technology and distributed object computing technology in military, aviation, communication, industrial control and other fields has been widely used. Using middleware technology in the distributed real-time system can reduce the dependence of hardware, the operating system and application software; The current mainstream of middle-ware products such as OMG CORBA, SunJ2EE, Microsoft DCOM OMG CORBA and so on. Compared with J2EE and DCOM, OMG CORBA not only in terms of the transaction security service is good, but its cross-language and cross-platform ability can't be matched by the latter.

At present, the most active research on the real-time middle-ware is University of Washington and the University of California at Irvine, they developed the TAO following the CORBA [1, 2] standard, it is a real-time CORBA specification and platform for high performance distributed middle-ware supporting for static QoS to application requirements. The research results have been applied successfully to the military, aerospace, industrial control, and achieved good economic and military effect. At the same time, the distributed multimedia applications has increased rapidly in the network bandwidth and CPU processing ability, which need to deliver continuous, real-time data such as audio, video streaming in [7,8] networks. At present, the popular multimedia real-time transmission system including Real Video Player of Real Networks, Vxtree of Microsoft, and more, in the field of open-source software, Open Meetings, a multilingual customizable video conferencing and collaboration system. It supports audio, video, can let you see everyone's desktop, and also contains the whiteboard, whiteboard can import images of various formats and graffiti. OpenH323 provides fully

[*] ***Corresponding author's*** e-mail: 394892738@qq.com

interaction functional, the open source C++ implementation of ITU H.323 video conferencing protocol. Ekiga is a compatible SIP and H.323 video conferencing program, compatible with VoIP, IP phone. Ekiga can make the video and audio dialogue with remote users to use any SIP and H.323 software and hardware.

To solve above problems, if use the traditional distributed systems built and integrate a new generation of weapon platform, which needs to the specific implementation code for different embedded systems and different network environment, the development cost of writing a lot of repetitive code is too high, which reduces the system reliability, maintainability and extensibility. And development of distributed system faced with cross-platform, different operating systems, different language, and cross-protocol. CORBA provides a common framework, it blocked the underlying hardware platform, operating system, as well as the communication protocol between heterogeneity, between the local and remote object using uniform communication interface for distributed and heterogeneous computer environment develop application. In this way, the distributed application developers don't need to care about and repeated processing of low-level details related to platform, and can focus on practical function in the development process of application logic. This will significantly reduce system development cost, shorten the development cycle, and make the system easier to maintain and upgrade.

The main goal of this article is in a distributed heterogeneous platform [12, 13] environment, using CORBA middle-ware [5] technology realize a multi-platform of real-time video data transmission system. Which involves video transmission technology, CORBA middle-ware technology in embedded and real-time operating system, the cutting and transplanting of ACE + TAO out of open-source CORBA products. At the same time, real-time video transmission system are analyzed in more operational platform to realize the main functions of the video real-time transmission requirements, based on the ACE/TAO a distributed real-time video data transmission system was designed and implemented.

2 The requirements analysis

2.1 THE FUNCTIONAL REQUIREMENTS ANALYSIS

Based on the requirements description of heterogeneous platform embedded real-time video data transmission system, the system needs to launch video, subscribe to the video, query videos, begin to receive video, unsubscribe video, access to the basic function. Participants are video acquisition, video subscribe to the end, and the system administrator.

The use case diagram of the system is as shown in Figure 1. System has three participants, respectively is video sender, administrator, and video receiver. The sender first register to the system, after registering successfully start publishing video, first change the global video release subscription data center and then to transmit data through the data transmission system. Video data acquisition device can be acquisition card, USB camera, desktop tools, etc. Video receiver register to the system first, and then by querying the video list subscribe to the corresponding video, after subscribing successful receive the corresponding video data through a video transmission module to get real-time playback, the video receiving end suspend the video and cancel it. System administrators can query the system state, manage registered task of released video and receiving end, and set video receiver.

FIGURE 1 The real-time video transmission system use case diagram

2.2 PERFORMANCE REQUIREMENTS ANALYSIS

This system mainly solves the distributed real-time transmission of video data, so in terms of performance mainly consider the following aspects:

Compression ratio: because of the great amount of raw video data, the video data transmission in the transmission system is compressed by video compression techniques. The higher the compression ratio is, the less the amount of bandwidth is.

Image resolution, Video is composed of sheets of continuous playback image, and the image is made up by one continuous pixels. The higher image resolution is, the greater the amount of storage space is. Under the fixed compression ratio, the greater the image resolution, the higher the bandwidth.

Bandwidth, the network can transfer the amount of data per second. As physical bandwidth is fixed, and cannot be changed. When the bandwidth of the minimum bandwidth requirements is higher than that of physical, systems can't run in the network situation resulting in a decline in the picture quality.

2.3 DATA ANALYSIS

Based on business requirements of embedded real-time video data transmission system, the system of data flow is sent from the video acquisition to the transmission system, transmission system make video data flow distribute and deliver based on video subscribe at the receiving end. Specific data flow diagram is as shown in Figure 2. Video transmitter and video receiver can through the user's id of system register and cancel. The registration and cancellation function is made to change the record of the system. The administrator can through the query command inquiry system running status. And through the access control command control the system permissions to each released video end subscribe end. Video end released the video data released by the input command. The compressed video data, after the group package is stored in the video data buffer. Video distributor control data packets distribution according to the registration, video distribution, video subscription and distribution, the receiver end receiving packets and made video decompressed to real-time displaying. At the same time, the receiver can send the subscription orders for video subscriptions.

2.4 THE ANALYSIS OF SYSTEM STRUCTURE

Almost all cluster operations platform on various systems connect to the local network, in this way, systems can interact between collaborations, and managers can send commands to the various systems through the network conveniently. Similarly, if the development of real-time video transmission system can be implemented by using local network data transmission function, in the same way, which can control all kinds of desktop systems and multimedia information transmission on the embedded system. In the data link of real-time video transmission, which mainly include front-end information acquisition and back-end receiving. If the back-end nodes need to receive the same video data of front-end node and can control real- time video data transmission mode and transmission rate, the front- end nodes will need to maintain connections to multiple nodes, and need to forward a packet for several times. Because the distribution of computer hardware system and software platform in the network is uneven, front-end and back-end node often need to keep the thin client model [11].

Based on the above reasons, this paper proposes a communication framework structure for sending, forwarding and receiving. As is shown in Figure 3: the client is mainly responsible for the video information collection and monitoring, the server is responsible for maintaining connections with multiple client, and complete the management of the node and data store and forward. Based on the framework, the video transmitter and receiver are decoupled, the sender does not need to know the receiver, and the receiver does not need to know who the sender was. Which flexible and can be used on the data transmission channels of different transport protocol for implement pluggable transport protocols. Video network model and programming of the different models can be used in the video transmitter and receiver, and on the forwarding server side can realize the access control, flow control, forwarding control functions and so on. Under the condition of the sending-end and receiving-end system platform unchanged, the server can be chosen to VxWorks [3, 4] systems having strong real-time as the server side, which make the whole system has a higher lateral extension function and real time.

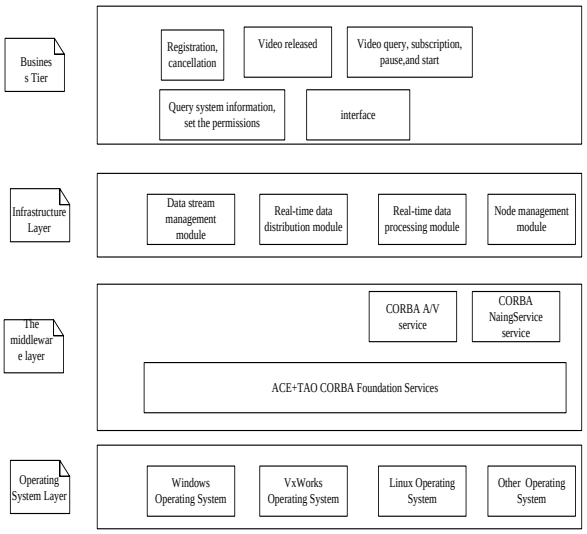

FIGURE 2 The embedded real-time video data transmission system data flow diagram

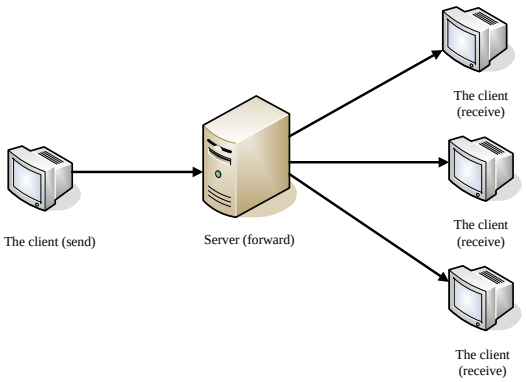

FIGURE 3 Real-time video transmission system communication framework structure

3 The system design

According to the real-time video transmission business requirements analysis, designed a real-time video data transmission system. The overall design of the system is given first and stratified according to the needs of system design; detailed design of specific modules in each layer is given. According to section 1 of the functional requirements and performance requirements, real-time data transmission system should not be able to depend on the specific operating system, network transport protocol. According to the video subscriptions at the receiving end

of video data real time acquisition, distribute, display, etc. The video receiver can query, subscriptions, play and pause video function. The system administrator can check online system operation and set permissions. Based on the above requirements, the system design is as follows.

System adopts hierarchical design, which is divided into the operating system layer, Infrastructure layer, business tier, middle-ware layer, the middle-ware layer use the concrete implementation of the underlying operating system, shield the underlying differences in multiple operating system, and choose open source CORBA middle-ware TAO as the foundation. CORBA service implements A/V streaming data services, CORBA NamingService service make node management. Infrastructure layer include flow management, real-time data distribution, real-time data processing module, node management, etc. Business layer realize video released, the receiving end of query, subscription, start and stop video function, administrators need to query system running condition, and permissions setting function. System overall structure are shown in Figure 4.

The above module according to the function marked out different layers in the first place, made the use of ACE + TAO as the underlying middle-ware layer to provide the basic infrastructure of the upper layer of CORBA services, ACE encapsulate the underlying operating system of unified API, including CORBA services and A/V CORBA Naming Service. Based On the middle-ware layer design infrastructure layer of real-time video data transmission system, which builds on the middle-ware layer in order to ensure the platform independence. On infrastructure achieved the business layer, which mainly solve various business functions in the requirements analysis. The layered design scheme improve the portability of the system, using a middle-ware layer ensure system is able to run under multiple operating systems [7]. The complexity of the system is layered, and achieves complex logic in the infrastructure layer. Business layer mainly focus on the specific business process, which is conducive to the needs of the business change.

4 The system implementation

4.1 THE SYSTEM RUNNING ENVIRONMENT

The targets of Real-time video transmission system operation include Windows, Linux and VxWorks operating system. Windows operating system as the most widely used of the operating system, whose good graphical interface greatly made the user's operation convenient. Windows system as sender will be used for the data collection to reduce driver development for hardware acquisition device. The VxWorks supporting for graphical interface is not friendly enough and the operation is not convenient, but with a real-time, stability and other characteristics. So used in forwarding end as a core server. The receiver uses the windows and Linux operating system.

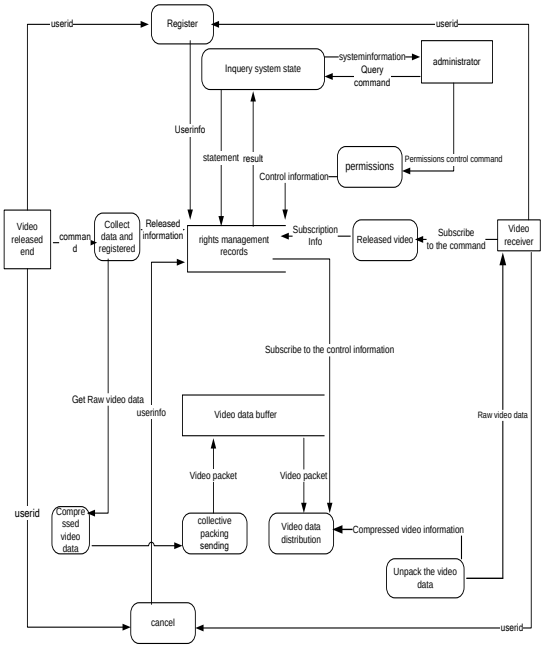

FIGURE 4 System function structure drawing

According to the above operation environment analysis, the system needs to implement distributed collaboration between an embedded system and ordinary desktop system, so the appropriate CORBA middle-ware products are needed to choose. ACE + TAO is a widely used, open source CORBA middle-ware, and with OMG CORBA2.6 specification. This provides support for multiple platforms (including Windows, Linux and VxWorks operating system). Due to the open source ACE + TAO, in order to get more flexible and to reduce the memory occupation in the embedded system, according to the system requirements, the CORBA product can be cut [9] out and optimized.

4.1.1 The ACE + TAO customization tailored to embedded platform

In the embedded systems, the storage resource is very limited. Requirements based on embedded system development program as far as possible use smaller memory and smaller storage space of the program itself. Smaller memory occupation can optimize the process to achieve, but smaller program storage space has been always a difficulty. Based on the above reasons, designed a program tailored to ACE + TAO dynamic program. Using the program can be cut to ACE + TAO, the cutting granularity achieved. CPP level, that is to say, if application uses file functions inside the. CPP, then compiles the. O files Corresponding. CPP file into the library file.

The purpose of this tool is optimized based on ACE + TAO dynamic link library of multiple application development storage space. Existing solution is based on static connection program, program only connect the needed object file module, but static connection more programs can cause the object file redundancy. For

dynamic connection of procedures, the use of multiple programs shares a complete library file, which is needed to install the whole dynamic link library. For a particular application, the application does not use all of the dynamic repository object files. This program only needs a subset of the object file. Specific cutting principle is as shown in Figure 5.

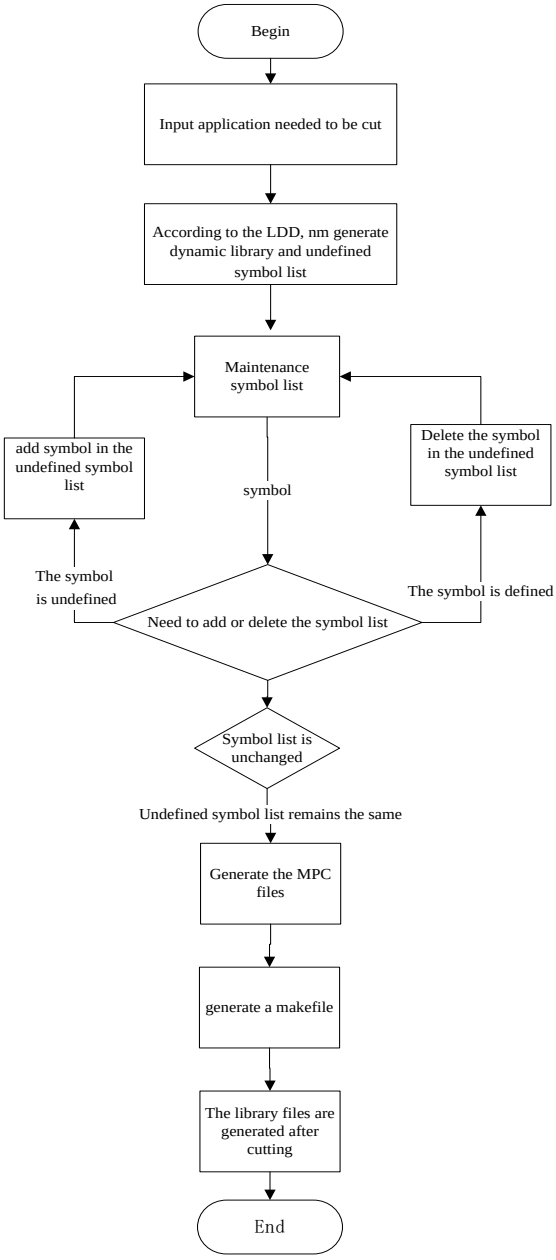

FIGURE 5 ACE + TAO customized cutting program flow chart

4.1.2 Compiled and installed TAO on multi-platform

TAO support multiple platforms such as Windows, UNIX, Linux, real-time operating system VxWorks, Linux OS, etc. Described the system in the article needs to run on multiple platforms, including Windows, Linux, VxWorks operating system. On the operating system compile and install the ACE + TAO, in different ways and configure trival, and according to specific application need to modify the corresponding macro and options.

4.2 THE IDL INTERFACE DEFINITION IN THE SYSTEM

The system use CORBA A/V Stream services as the underlying data transmission and control function, but the A/V services provided cannot satisfy all the control function[12] in section3, part of the function need to use standard CORBA IDL interfaces to define its implementation. The IDL interface is the operating range of description customer sending a request through its object defined. Interface illustrates the support services provided by the interface how to through the set of operations accessed syntax description. Among them, A/V services interface have IDL interfaces with A and B completely different types. Interface definitions lies in the OMG A/V stream specification, there is no need to design later. Only needs to be defined in the application layer a MediaCtrl interface to call streaming service for completing convection control. The specific interface design in the application layer is as shown in Figure 6.

Above the figure, according to the module made division of application layer for IDL interface. Each module in the interface definition of operation corresponds with the operation of a CORBA object requesting call. Client and server communicate with each other through these operations.

4.3 THE NAMING SERVICE IMPLEMENTATION

The server end needs to solve the problem of client for object references, and the client for object reference process includes three steps:

The server end provide object to be announced to the object directory, and some can be in a meaningful way to provide to identify the properties of the object.

The client end submit needed object properties to the object directory, the directory will make the matched object back to the client.

Client end gets object references, server end completed corresponding operation.

In this system mainly using the CORBA naming service implement object orientation. The communication principle is as shown in Figure 7. Server-side regist servo object to the named server, the client program could via the names of the objects provided on the naming service parse it into an object reference, and then the client end sends a request to the server, the server processes the request and returns the result to the client application.

Before the system start up, which need to open the naming service. The naming service programs can be distributed in any computer in the LAN, but its IP address and port number to reference naming service programs are public.

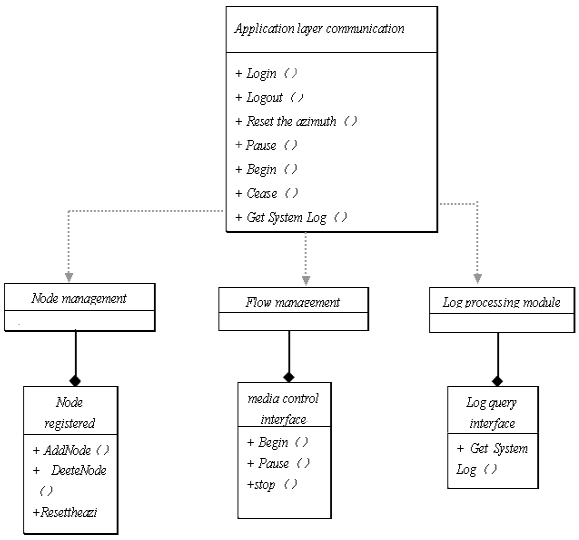

FIGURE 6 Communication interface class diagram

Start naming service mode: open a console window, input NamingService –m 1 ORBEndPoint iiop://192.168.0.100:8000. Parameters:

- m 1 using multicast (multicast) way, so the server program don't have to specify the IP address of the host.

- ORBEndPoint iiop: //192.168.0.100:8000: Use the IIOP communication protocols, and specify the start naming service host's IP address and port number.

4.4 STAR THE SERVER

The server side need to first start, which will bind servo class to the naming service, client via CORBALOC access to the service object references. The Started server main code is as follows:

```
* / / * server startup part main code
/ / initialize the ORB
CORBA::ORB_var orb=CORBA::ORB_init(argc,argv);
// To obtain the root POA object references
CORBA::Object_var
obj=orb->resolve_initial_references("RootPOA");
PortableServer::POA_var poa=PortableServer::POA::_narrow(obj);
// Activate the POA manager
PortableServer::POAManager_var
mgr=poa->the_POAManager();
mgr->activate();
// Create the servo class instance
Regester::Connection_impl servant;
// Get the naming service context reference
CORBA::Object_var naming_context_object =
orb->resolve_initial_reference("NameService");
CosNaming::Naming::Context::_narrow(naming_context_object.in(
));
// Create and initialize a name sequence
CosNaming::Naming name(1);
name.length(1);
name[0].id=CORBA::string_dup("Regester");
// In naming service regist object references,which will bind names
and objects
naming_context->rebind(name_robot.in( ));
// / At last run the ORB, accept the request
orb->run( );
```

According to the design of the node management module, each node of the control class object contains a connector object "Connector", and the "Connector" object contains a server object references. Before registing nodes, there need to be initialized the CORBA connections with the server, which includes the ORB initialization and access server object references. The initialization code lies in Connector Login interface.

First open a background tasks to run naming program, the program monitor the network card. Then open the task server and client respectively and specifies the naming service address.

5 System testing

5.1 TEST ENVIRONMENT

Test environment including 6 PCS, 100 MB card, a camera, a video capture card. Among them, the sender has a camera and video acquisition card, running on Windows XP; Forwarding server runs on VxWorks. Two of Four receiver computers run on Windows XP, two run in the Red Hat Linux 9.0. Client and server via a LAN Ethernet connect each other, which formed a network environment.

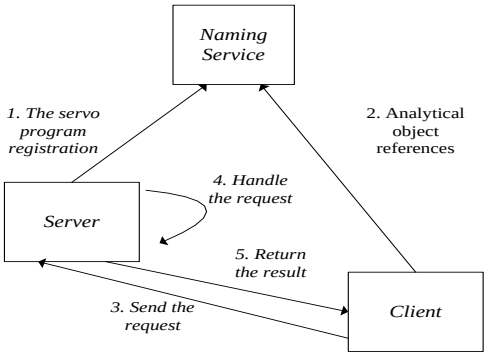

FIGURE 7 The principle diagram of the naming service

5.2 FUNCTIONAL TEST

Because this system involves the communication between the client and server, in the testing process, which needs to start the application program on the server side, then start the client side program, and for each system function by the client call corresponding operation test, not only test in the client the accuracy of the corresponding module and exception handling, also test the function of each module in the server-side action. In the whole testing process, mainly use the method of black box testing, if whose call process is properly according to call the return value in line with the expected value and the operation functions is correct. Node management, flow management was tested respectively; the testing process in line with the expectations, test example is as shown in Tables 1-3.

TABLE 1 The node management operational testing

Testing purpose	Testing process	Testing results
Correct registration testing	Input the correct node attributes	Registration is successful, returns 0
The cancellation of the test	Registered to the server first and then calls the cancellation of the node	Cancellation is successful, returns 0
Reset the bearing test	Registered to the server first, and then call reset bearing operation	Reset successful, returns 0
The node ID error test	Input the ID of wrong node	Registration failed, return 1
The node role error test	Start a client first, send role and properly regist to the server and then start another client end, set sending characters trying to registing to the same server	Registration failed, return 2
Full test to connection number of nodes	To set the maximum number of connections on the server, which in turn start the client, and connect to the server, starting the client number is greater than the maximum number of connections	The connection fails, return 3
Reliability test	Before the call operation, delete node configuration file on the server	Failed to open the configuration file, return -1

TABLE 2 Test flow management operation

Testing purpose	Testing process	Testing results
Flow to begin testing	The customer end nodes are registered to the server at the beginning of the sending end call operation.	Normal display video information at the receiving end, returns 0.
	The customer end nodes are registered to the server at the beginning of the receiving end call operation.	The receiver shows normal video information, returns 0.
Flow to suspend the testing	One sending node and two receiving node are registered to the server, in the process of sending the flow, in the sender end call flow to suspend operations.	Suspend two video receiving data at the receiving end, returns 0.
	One sending node and two receiving node are registered to the server, in the process of sending the flow, in one of the receivers call flow to suspend operations.	The receiver display a normal video information, and the other receiver end suspend, return 0
Flow to stop testing	One sending node and two receiving node are registered to the server, in the process of sending the flow, in one of the senders call flow to suspend operations.	The two receiver video data at the receiving end stop receiving, returns 0
	One sending node and two receiving node are registered to the server, in one of the senders call flow to stop operations.	One receiver display video information normally, and the other stop, return 0.

TABLE 3 The access log operation testing

Testing purpose	Testing process	Testing results
Access the logging test	Registered to the server first, and then can be called in the correct time interval. Access log operation.	To be successful, returns 0, shows the log information
Time zone test	Input the starting time after the termination time.	Tips range is not correct

5.3 PERFORMANCE ANALYSIS

On the system communication link, access to receiver from 1 to 10 respectively. Here are two and four test results listed in Table 4 and 5.

TABLE 4 The computer test results connection of two receiving end

Image resolution	Receiving end	Transmission rate (kbps)	Frame rate (fps)	Delay (ms)
352pixel×288pixel	1	507~545	30~32	97~100
	2	505~540	30~32	98~100
496pixel×384pixel	1	580~640	19~23	110~112
	2	780~820	18~21	117~123
Only transfer rate	1	620~680		
	2	1021~1078		

TABLE 5 Test results connecting the four receiver computer

Image resolution	Receiving end	Transmission rate (kbps)	Frame rate (fps)	Delay (ms)
352pixel×288pixel	1	280~340	19~23	106~120
	2	420~480	25~31	99~105
	3	270~290	16~23	120~127
	4	320~370	19~23	108~120
Only transfer rate	1	320~360		
	2	520~540		
	3	230~280		
	4	460~480		

Based on the research of the transmission rate, select a resolution of 352 pixel x 288 pixel to compare; when the size of a packet is 1500 bytes, transmission rate is 1 MBPS. When a packet size is 3050 bytes, the transmission rate is about 1.5 Mbps. This is caused by increasing cache utilization rate.

The experimental data can be seen, the total transfer rate increase bigger as connection number of the receiver increasing. As the transmission rate between multiple receivers is different, which is related to the client operating system. In real time, when connect four servers in the receiving end, which still can guarantee about 100 ms delay and can meet the demand of the system proposed 120 ms delay.

On image quality, the current video compression standard select mpeg-4 compression algorithm, video frame is composed by frame I, P frame and B frame.

6 Conclusion

Used thin clients Model, introduce the CORBA middleware, made a dynamic program tailored ACE + TAO program, development program as far as possible use smaller memory and smaller storage space, Combined with embedded real-time Middleware technology to construct distributed heterogeneous video transmission system. A Real-time video transmission system has been implemented based on CORBA technology, which compared with the development distributed system in the traditional way, the application objects in the system interoperability are more transparent, effectively complete various transmission and remote service functions required by the system; the convenience of system upgrade and maintain, greatly improve the efficiency of system development. At the same time, because the open source products CORBA TAO has been cut and also ensured that features of embedded system, data distributed server used a combination of VxWorks real-time operating system and real-time CORBA, which made the real-time performance of system greatly improved. Forwarding the serve side of the current implementation of the system cannot inform initiatively each node corresponding management information system. Every time the receiver can get server management node information through calling query node information. Due to the audio and video collection use different devices, and sampling frequency is different, so the problem of synchronization of audio and video data is needed to solve.

Acknowledgements

The work was supported in part by a grant from Military and Civilian Integration Resarch Fund Project of Shaanxi Province (No:12JMR19), the mathematical disciplines fund projects of shaanxi province support disciplines (No:14SXZD010).

References

[1] Calvo I, Almeida L, Noguero A, Pérez F, Marcos M 2014 A flexible time-triggered service for real-time CORBA *Computer Standards & Interfaces* **36**(3) 531-44
[2] Toral S L, Barrero F, Cortés F, Gregor D 2013 Analysis of embedded CORBA middleware performance on urban distributed transportation equipments *Computer Standards & Interfaces* **35**(1) 150-7
[3] Zhang C H, Peng R M 2013 Design of Device Driver in VxWorks. *Applied Mechanics and Materials* **380** 2038-41
[4] Han G N, Li Y F 2011 Analysis and Implementation of TrueFFS Based on VxWorks System *Advanced Materials Research* **41**(1) 584-7
[5] Su Z, Hao J, Qu J, Wu R 2013 A distributed system reliability analysis Based on the middleware *Computer engineering and design* **5** 1669-72
[6] Liu X, Jin Z, Wei J, Zhao X 2013 P2P network multi-view stereo video transmission data block scheduling strategy *Computer engineering and application* (3) 1-6
[7] Wang X Y, Zhang J 2014 A Video Compression Coding Algorithm for Network Content Transmission *Applied Mechanics and Materials* **536** 81-8
[8] Jiang Z L, Zhu J J, Shao Y F 2014 A Media Streaming Data Scheduling Algorithm Based on P2P *Applied Mechanics and Materials* **552** 377-80
[9] Zhang J H, Lei L, Li J F, Cui X Y, Wu Y 2013 Research on Electronic Equipment Fault Diagnosis Expert System Based on Embedded Linux *Advanced Materials Research* **683** 837-40
[10] Kim D-O, Park R-H, Sim D-G 2013 Image and Video Quality Assessment Based on the Similarity of Edge Projections *Signal and Image Processing* **4**(1) 1-16
[11] Fanning K 2014 Thin Client: New Cost Savings? *Journal of Corporate Accounting & Finance* **25**(3) 7-12
[12] Daivandy M, Hünich D, Jäkel R, Metzger S, Müller-Pfefferkorn R, Schuller B 2013 Heterogeneous resource federation with a centralized security model for information extraction *Journal of Internet Services and Applications* **4**(1) 202-5

Authors

Jiangyan Sun, May 1965, China.

Current position, grades: researcher at Xi'an International University of China.
University studies: master's degree in software engineering at Xidian University of China.
Scientific interests: text classification and software engineering.

Xiaoqiang Jia, March 1977, China.

Current position, grades: researcher at Weinan Normal University of China.
University studies: master's degree in software engineering at Xidian University of China.
Scientific interests: text classification and software engineering.