http://www.zju.edu.cn/jzus; E-mail: jzus@zju.edu.cn
ISSN 1009-3095 *Journal of Zhejiang University SCIENCE* V.5, No.1, P.22-30, Jan., 2004

A hierarchical architecture of centralized monitoring and controlling system and its high-performance and interoperability protocol*

HUANG Li-can(黄理灿)†, WU Zhao-hui(吴朝晖), PAN Yun-he(潘云鹤)

(*College of Computer Science, Zhejiang University, Hangzhou 310027, China*)

†E-mail: lchuang@cs.zju.edu.cn

Received Sept.11, 2002; revision accepted Feb.10, 2003

Abstract: This paper describes a hierarchical architecture and a high-performance and interoperability protocol for centralized monitoring and controlling systems (CMCS). The protocol we proposed can interoperate different monitoring and controlling systems constructed by different companies, each with different functions and communication protocols. The protocol reduces the amount of traffic and has real-time and high-performance advantages. The protocol was implemented in CMCS for telecommunication power supply and air-conditioner used by the Telecommunication Bureau of Zhejiang Province. This paper deals with the hierarchical architecture and function of CMCS and packet format, command ID, and SDL description of its protocol. We also discuss the properties of the interoperability and performance of the protocol in this paper.

Key words: Monitoring and controlling system, Protocol, Architecture
Document code: A **CLC number:** TP393.04

INTRODUCTION

Centralized monitoring and controlling system (CMCS) is important in many fields such as telecommunication network management (Telecommunication General Bureau of China, 1997a; 1997b; Ministry of Posts and Telecommunications of China, 1997; Huang, 1999), traffic network management, electric power network management, environment monitoring system, virtual scientific experiments, and so on. CMCS is a computer network system that senses signals, controls distributed equipment remotely, displays monitored data and alarms in real-time, and records monitored data and alarms. It also includes information management, configuration file management, etc.

Hierarchical CMCS is a CMCS with hierarchical structure. It has two main advantages. Firstly, its hierarchical control makes management of distributed equipment easy; secondly, its remote and centralized monitoring enables understanding of the entire system's outline.

Hierarchical CMCS has two difficult problems to be solved. (1) Because there are a lot of heterogeneous and distributed sub-layer systems, each with different functions and communication protocols, constructed by different companies, how to interconnect these heterogeneous sub-layer systems into an upper-layer system is a very important problem. (2) The centralized architecture of the CMCS results in an increased traffic overhead at the top sub-layer systems.

The protocol we proposed solves these problems. It reduces the amount of traffic and has real-time, high performance and scalable properties. The protocol gives a unified specification for the interoperability of heterogeneous and distributed sub-layer systems. The protocol was implemented in CMCS for telecommunication power supply and air-conditioner used by the Telecommunication Bureaus of Zhejiang Province (Huang, 1999; Huang and Wu, 2001). In this paper, we describe the architecture of the centralized monitoring and controlling systems and its protocol, discuss the advantages of the protocol, and analyze the properties of the interoperability and performance of the protocol.

* Project supported by virtual cooperation research granted by the Chinese Ministry of Education

HIERARCHICAL ARCHITECTURE OF CENTRALIZED MONITORING AND CONTROLLING SYSTEM

The architecture of the centralized monitoring and controlling system presented here has a tree-like hierarchical structure as shown in Fig. 1. The different layer subsystems have different functions and structures. The upper-layer subsystem interconnects with the lower-layer subsystems. There are one root subsystem and several leaf subsystems in the system. Each subsystem can be used independently without upper-layer subsystem. Hierarchical CMCS is the union of SCADA, network management, and MIS. Every layer of subsystems includes at least the following modules:

Equipment control and data acquisition
Alarm
Historical data management
System maintenance
Protocol Communication

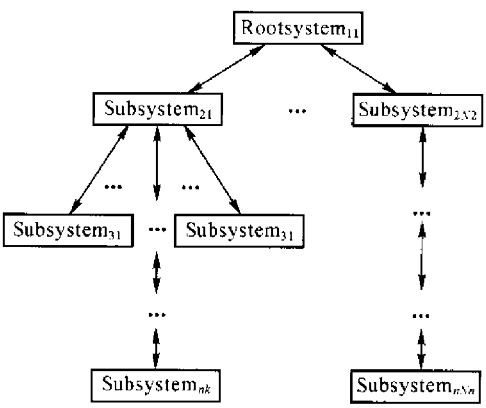

Fig.1 Architecture of hierarchical centralized monitoring and controlling system

Upper-layer subsystem functions through lower-layer subsystems, can ignore the control point of the lower-layer subsystems by configure database; and generally takes care for the more important control points. The lowest-layer subsystem (leaf subsystem) can actually control the equipment and acquire data and alarms. Fig. 2 shows the scenario of command flow. In Fig. 2, Subsystem$_{31}$ is a leaf node; subsystem$_{21}$ is an upper-layer subsystem; and subsystem$_{11}$ is root subsystem. The command flow may be as follows: (1) subsystem$_{11}$ produces the command; (2) subsystem$_{11}$ forwards this command to subsystem$_{21}$; (3) subsystem$_{21}$ forwards the above command to subsystem$_{31}$; (4) Subsystem$_{31}$ executes the command; (5) Subsystem$_{31}$ forwards the results to subsystem$_{21}$; (6) subsystem$_{21}$ forwards the results to subsystem$_{11}$.

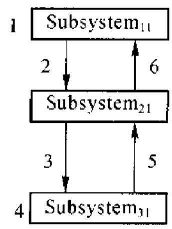

Fig.2 Flow of commands and results

PROTOCOL DESCRIPTION

The protocol communication is full duplex point-to-point connection, which uses bi-directional ARQ (automatic repeat request) mode. Fig.3 shows a scenario of the flow of commands and data between different layer subsystems. Because of limited paper length, we only describe

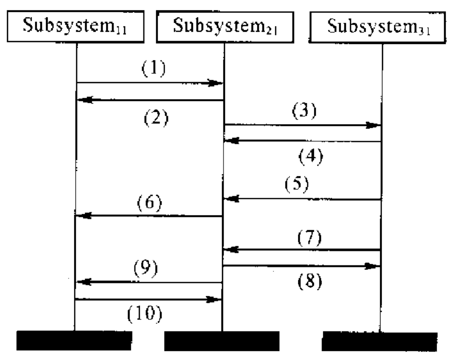

Fig.3 Scenario of flows of commands and data described by Message Sequence Chart (ITU-T, 1992)

(1) Send Command1;
(2) Acknowledge (1);
(3) Forward Command1;
(4) Acknowledge (3);
(5) Send Real-time-data1;
(6) Forward Real-time-data1;
(7) Send results of Command1;
(8) Acknowledge (7);
(9) Forward results of Command1;
(10) Acknowledge (9)

the three main parts of the protocol (packet format, command ID and description of SDL (Specification and Description Language) (Belina *et al*., 1991; ITU-T, 1995; 2000; Khendek and Vincent, 2000)

Packet format

There are two types of packets, Command packet that transmits the command code and data, and ACK packet giving notification of the correctness of the received command packet. The packet formats are listed in Tables 1 and 2.

Table 1 Command packet format

Series No.	Bytes	Name
1	1	SOI(packet start)
2	1	VER (version)
3	2	FRAMENO (packet ID)
4	1	MESSAGE_TYPE
5	1	SOURCE_ID
6	1	DESTINATION_ID
7	1	CID (command ID)
8	1	RESENTCOUNT
9	2	LENGTH
10	X	DATAFIELD (PDU)
11	2	CHECKSUM
12	1	EOI (packet end)

Table 2 ACK packet format

Series No.	Bytes	Name
1	1	SOI(packet start)
2	1	VER (version)
3	2	FRAMENO (packet ID)
4	1	MESSAGE_TYPE
5	1	SOURCE_ID
6	1	DESTINATION_ID
7	1	CID (command ID)
8	1	RTN (notify error types)
9	2	CHECKSUM
10	1	EOI (packet end)

The data types used in Packet Data Unit (PDU) of the packets are classified into standard data types and structure types, which are defined by ASN.1 (ITU-T, 1998a; Olivier, 2002). The encoding rule of ASN.1 data types is PER (ITU-T, 1998b). Some data types are as listed below.

```
SubsystemID: = OBJECT IDENTIFIER
Control_point_identification: = SEQUENCE {
  equipment_identification
        OCTET STRING (SIZE (2)),
  equipment_trademark_identification
        OCTET STRING (SIZE (1)),
  equipment_serial_No
        OCTET STRING (SIZE (1)),
  module_serial_No
        OCTET STRING (SIZE (1)),
  monitor_and_control_point CHOICE {
    remote_control
        OCTET STRING (SIZE (3)),
  telemeter
        OCTET STRING (SIZE (3)),
  remote_regulate
        OCTET STRING (SIZE (3))}}
Control_point_group: = SEQUENCE OF
        Control_point_identification
Query_data : = SEQUENCE OF {
        subsystem SubsystemID,
        control_point_group Control_point_
  group }
```

Command ID

Command IDs, their meanings and PDU names defined with ASN.1 are shown in Table 3. Commands from upper-layer subsystem can be classified into two types (general commands, bracket commands). General commands require the lower-layer subsystem to execute only once. Bracket commands require the lower-layer subsystem to execute repeatedly until the upper-layer subsystem sends another bracket command to end this command. These commands are "Begin to Query data" and "End to Query data".

SDL description of the protocol

The subsystem has three arrays to store queues: CommandArray stores the new generated commands to be sent; AnswerArray stores the ACKs to be sent; and ReceiveCommandArray stores the received commands to be executed later. The subsystem has three processes: CommandProcess accepts new commands and executes the received commands; SendPacketProcess sends command packets and ACK packets; and ReceivePacketProcess receives command packets and ACK packets. We use Telelogic TAU SDL suite (Telelogic Company, 2002) to model and describe the protocol. In the descriptive SDL diagrams, we only focus on the control sections of the protocol. We omit the dealing of PDU and operations on the above queues. Fig.4 shows the two subsystem blocks interoperating with each other and the details of these subsystem blocks. Subsystem block CMCSBlock_n is the same as CMCSBlock_m except that the input route in commuChannel Channel is the output route of CMCSBlock_m, and vice versa.

Table 3 Command ID

CID	Description	PDU parameter
11H	Notify system's reset	None
12H	Query line status	None
13H	Begin to Query data	Query_data
14H	End to Query data	Query_data
15H	Time checking	Time_stamp
16H	Begin to take over subsystem's control	Take_over
17H	End to take over subsystem's control	Take_over
18H	Remote control	Remote_control
19H	Set parameters	Set_parameter
1AH	Get parameters	Get_parameter
1BH	Get report data	Get_report
1CH	Get statistical data	Get_statistics
1DH	Get historical data	Get_history
31H	Send real time data	Real_time_data
32H	Send results of command execution	Command_result
33H	Send alarm	Alarm
34H	Send parameters	Send_parameter
35H	Send report data	Send_report
36H	Send statistical data	Send_statistics
37H	Send historical data	Send_history
38H	Enforce to end subsystem's control	End_take_over
39H	Send subsystem's configuration	Send_config

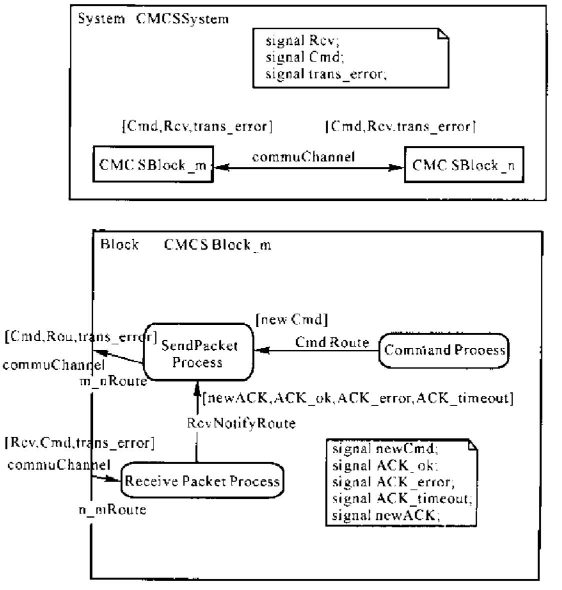

Fig.4 SDL diagrams describe CMCS system and Blocks

THE INTEROPERABILITY ANALYSIS OF THE PROTOCOL

This paper proposes a protocol on how to interconnect heterogeneous and distributed subsystems into a centralized and hierarchical system. The protocol is based on ASN.1 standard which has the benefit of interoperability of all heterogeneous systems as Fig.5 shows. Furthermore, conversion between the local IDs (such as

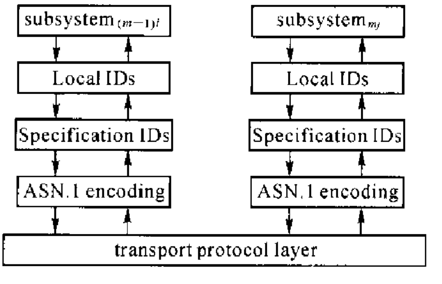

Fig.5 interoperability of the protocol

Note: Subsystem$_{mj}$ is the father node of Subsystem$_{(m-1)i}$ in hierarchical CMCS tree

control-point ID, equipment ID, and so on) and the specification IDs makes possible the interoperability of the heterogeneous subsystems developed by different companies. Moreover, our protocol can retain all the functions of the subsystems and satisfy all the requirements of CMCS. To implement the protocol does not change previous system structure and only needs a little code changing, especially if the protocol and ID conversion are implemented and wrapped by using software component technologies such as COM, JavaBean or CORBA.

PERFORMANCE ANALYSIS OF THE PROTOCOL

In CMCS, showing data in real-time is important. In general there are three kinds of protocols to get real-time data. The simple protocol (which belongs to the kind of all data transferred protocol) to get real-time data is to transfer all data up to the upper-layer subsystem. But this protocol will cause a large amount of traffic and a significant delay. Another protocol to get real-time data is SNMP (Sidnie, 1995) protocol (which belongs to the kind of query/response protocol), which queries real-time data repeatedly. If there are many real-time data to be obtained at the same time, the need of many queries and the mode of repeated query will cause significant delay. The protocol presented here (which is the kind of selected data transferred protocol), only transfers the needed data up to the upper-layer subsystem, and has high performance.

We know that the data representations are limited in the software system, including table, graph, curve, sound, and so on, and the data people pay attention to are also limited. For example, if we focus some time on the real-time data of UPS, then the real-time data of a diesel generator cannot be represented and people do not have interest in these data at the same time. So, at this time, the real-time data of the diesel generator are "unuseful". Otherwise, if we focus some time on the real-time data of a diesel generator, which may be represented as a trend curve, then the real-time data of UPS are unuseful. Our protocol is to ignore these unuseful data. To do it, we use the following method: when people focus on some data, the system sends "Begin to Query data command" to ask the lower-layer subsystem to begin transferring these real-time data uninterruptedly, so that people can see these data changing in real-time; when people focus on other data, then the system sends "End to Query data command" to terminate the transferring of those data, and sends another "Begin to Query data command" to ask for the required data.

Ordinarily, the number of the real-time data represented is about 100, and usually the data are clustered as an apparatus. Most of the CMCS systems display no more than 200 control point data per display window. Is it necessary to indicate all these data in the PDU of "Begin to Query data command" or "End to Query data command"? It is unnecessary because we use the agile below.

In Section "Packet Format", we defined the type of Control_point_identification. The Control_point_identification includes five items: equipment_identification, equipment_trademark_identification, equipment_serial_No, module_serial_No and monitor_and_control_point. Control_point_identification is a hierarchical type, encoded sequentially from equipment to control point. If all the fields after some field in Control_point_identification are zero, then will all the control point data in this field start or terminate to transmit. For example, in "Begin to Query data command" or "End to Query data command", if the monitor_and_control_point field is the only field with zero value, will all the control points included in the module of equipment indicated by module_serial_No start or terminate to transmit; if all the fields have zero values, then all the control points in the subsystem (actual numbers and types of the data are determined by the specification in the configuration file) will start the transmission or terminate it. As people always focus on only one apparatus at a time, so the PDU of the command packet of "Begin to Query data command" or "End to Query data command" can only use one Control_point_identification to ask the lower-layer subsystem to start or terminate to transmit the whole data of an apparatus.

Let us take a scenario example to illuminate our strategy. If a computer in Ningbo city sounds the alarm "failure of bypass of UPS in Guoju Town", and the user heard it, he then clicks the linkage to show the situation of the alarm UPS. The subsystem sends "Begin to Query data command" command packet to the lower-layer subsystem in Xinqi (a county in between) and this low-layer subsystem passes over the command to the next lower-layer subsystem in Guoju, then Guoju transmits all data of this alarm UPS to the Xinqi subsystem and then passes over to the Ningbo subsystem. Finally, the entire data of the alarm UPS (including input AC pressure, input DC pressure, pressure of storage battery, temperature of storage battery, output AC pressure, output AC current, output frequency, etc.) are passed to the Ningbo subsystem and refreshed real-timely, and the user can see the data represented via table, figure, real-time trend curve, and so on. Later, if the user of Ningbo subsystem in the same computer wants to know data on a diesel generator in Xiaogang town, the subsystem sends "End to Query data command" to the subsystem in Guoju to terminate transmitting the UPS data, and sends "Begin to Query data command" to the subsystem in Xiaogang to ask to transmit the some data on the diesel generator (including output electric pressure, output electric current, output power, output rotate speed, temperature of water, pressure of engine oil, electric pressure of storage battery in startup, volume of diesel oil, etc.) determined by Control_point_identification in the PDU of "Begin to Query data command" command packet.

Mathematical analysis

The terminologies and definition of hierarchical CMCS are listed in Table 4. And terminologies of traffic and delay are listed in Table 5.

Suppose the hierarchical CMCS has n layer structure. We have the following formulae:

$$D_{\mathrm{in}_{ij}} = \sum_{(i-1)k \in S_{ij}} D_{out_{(i-1)k}}, \ 1 \leqslant i < n \quad (1)$$

$$D_{\mathrm{out}_{ij}} = D_{in_{ij}} + D_{p_{ij}}, 1 < i < n \quad (2)$$

$$D_{\mathrm{out}_{nj}} = Dp_{nj} \quad (3)$$

$$T_{d_{ij}} = D_{out_{ij}} / C_{ij}, \ 1 < i \leqslant n \quad (4)$$

$$Lcp_i = C_{p_i} \times number + Packet\_header \quad (5)$$

Here, *number* is for the number of command parameters; *Packet_header* is the number of bytes of packet length except PDU, that is, 14 bytes.

$$Dp_{ij} = \sum_{k=1}^{n_{cmd_i}} Lcp_k \times P_{ijk}, 1 < i \leqslant n \quad (6)$$

Table 4 Terminologies of hierarchical CMCS

Symbol	Definition	Remark
N_{all}	Number of all subsystems	
S_{Leaf}	Set of leaf subsystems	
N_{leaf}	Number of leaf subsystems	
S_{non_leaf}	Set of non-leaf subsystems	
N_{non_leaf}	Number of non-leaf subsystems	
N_i	Number of the i-th layer subsystems	
Sub_{ij}	Subsystem identified by substring ij	i is the layer order counted from root subsystem and j is the order in the i-th layer. see Fig.1
S_{ij}	Set of substrings of the direct children of Sub_{ij}	S_{ij} = ((i − 1) k | $Sub_{(i-1)k}$ ∈ the direct children of Sub_{ij}).
$Spath_{ijkl}$	Set of substrings of the subsystems that are in the shortest path of CMCS tree	$Spath_{ijkl}$ = (st | Sub_{st} ∈ the subsystems that are in the shortest path of CMCS tree from Sub_{ij} to Sub_{kl}).

Table 5 Terminologies of traffic and delay

Symbol	Meaning
$D_{in_{ij}}$	Data bytes Transferred to Sub_{ij}
$D_{out_{ij}}$	Data byte sent out by Sub_{ij}
Dp_{ij}	Data byte produce by Sub ij
Ts_{ij}	Send Delay of Sub_{ij}
Td_{ij}	Transmission Delay of Sub_{ij}
C_{ij}	Transmission Rate of Sub_{ij}
Tp_{ij}	Data Producing Delay of Sub_{ij}
Lcp_i	The length of packet with PDU of Command parameter i
Cp_i	The length of command parameter
P_{ijk}	The probability of command k in Sub_{ij}

The traffic that is received by root subsystem is given by formula

$$D_{\mathrm{in}11} = \sum_{i=2}^{n} \sum_{j=1}^{N_i} Dp_{ij} \quad (7)$$

The total delay of traffic from subij to subkl is given by Eq.(8)

$$T_{\mathrm{all}\,ijkl} = T_{pij} + \sum_{st \in Spath_{ijkl}} (Ts_{st} + T_{dst} + T_{est}) + T_{ekl} \quad (8)$$

We now only consider the delay from leaf subsystems to root subsystem. So,

$$T_{allnj11} = Tp_{nj} + \sum_{st \in Spath_{nj11}} (Ts_{st} + T_{dst} + T_{est}) + T_{e11} \quad (9)$$

We use a three layer CMCS example to illustrate the protocol performance. In this example, there is a root node, which has six child nodes, and these child nodes also have 20 child nodes each.

For easier analysis, we just consider transmission delay, and we suppose transmission rate are all same; all leaf subsystems produce the same traffic; all non-leaf subsystems do not produce traffic; a control point datum needs 12 bytes (8 for ID, 4 for data); and the protocol presented here needs no more than 200 control points at the same time.

As Figs. 6 and 7 show, the traffic and transmission delay of the example system with the protocol we proposed remains unchanged when control point number increases; whereas the traffic and transmission delay with all data transferred protocol is proportional to the number of control points.

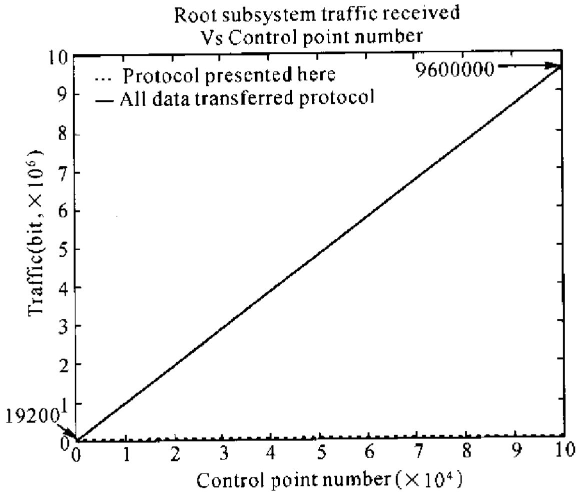

Fig. 6 Relation between Control point number and traffic received by root subsystem

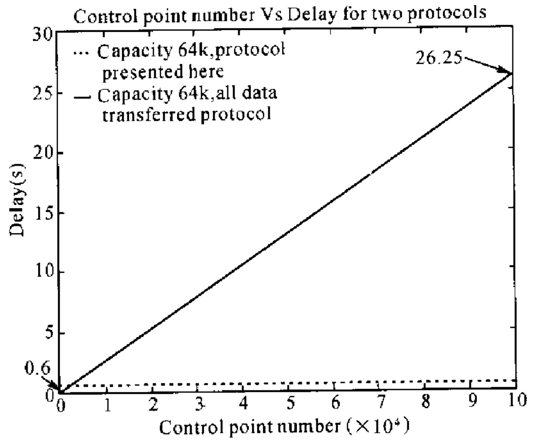

Fig. 7 Relation between Control point number and Delay

Fig. 8 shows that the relationship between control point number and capacity needs less than one second delay in the example system. Our protocol needs capacity of 38400 bit/s; whereas all data transferred protocol needs more than 1.68M bit/s when control point number is 100000. This case is possible because each of the 120 leaf subsystems only has 833 control points.

We use this architecture and protocol in CMCS for telecommunication supplies and air-conditioners in the Ningbo Telecommunication Bureau. The system architecture is a three-layer system, which has one root subsystem (Supervision Center) in Ningbo City, 6 middle-layer subsystems (Supervision Station) in the related counties, and average 20 leaf subsystems each Supervision Station (supervision Unit) in related

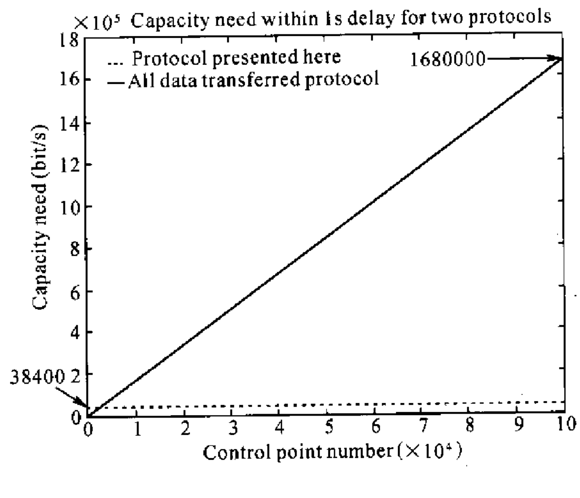

Fig. 8 Relation between Control point number and capacity needed

towns. The system uses DDN with 64 Kbit/s capacity to transmit real-time data. The test results of response time are shown as Table 6.

Table 6 Response of CMCS of Ningbo Telecommunication Bureau (test point randomly chosen from 5000 control points by the group of the project-test experts

Test type	Test numbers	Response time
Remote sensing	173	< 1 s
Remote signaling	113	< 8 s
Remote control	12	< 5 s

CONCLUSIONS

The hierarchical architecture of CMCS makes it possible to connect heterogeneous and distributed centralized monitoring and controlling systems into an integrated system, which is required in the Internet era. CMCS with this architecture has two main advantages. Firstly, its hierarchical control gives convenient management of a large number of distributed equipment. Secondly, its hierarchical remote and centralized monitoring enables the outline understanding of the entire system.

The protocol solves the important problem of how to interconnect heterogeneous and distributed subsystems into a centralized and hierarchical system. This is due to the following reasons: (1) The protocol gives a unified specification, which stipulates different subsystems with different functions and protocols, and it enables constructing the centralized monitoring and controlling system from heterogeneous and distributed subsystems because it is based on ASN.1 standard and conversion of IDs. (2) The protocol enables the heterogeneous and distributed subsystems to retain all their functions. (3) The protocol has high-performance and real-time properties. (4) The subsystems require little change in code to implement the protocol without changing previous system structure, especially if the protocol is wrapped by using software component technologies such as COM, JavaBean or CORBA. Because the subsystems are always constructed by different companies prior to the plan as a whole or due to the consideration of avoiding monopoly, and central management is a natural way in many fields, the protocol we proposed here that enables inter-connection of these heterogeneous systems has great practical significance.

By introducing the strategy that only the "useful" data that people currently pay attention to are transferred to the upper-layer and all the other "unuseful" real-time data are ignored, the protocol reduces the communication traffic largely and remains constant traffic with the increase of control points. Unlike request and respond mode in SNMP, only the "useful" real-time and alarm data subscribed by "Begin to Query data command" are transmitted to the receiver automatically in this protocol. This reduces the traffic caused by transmitting the query command every several seconds and the traffic caused by the "unuseful" data. Moreover, the coding principle of control point identification can also reduce the amount of traffic, in which a group of data can be asked by only one proper Control_point_identification.

The protocol has real-time and high performance properties. Besides decreasing the amount of traffic as described above, the protocol uses asynchronous mode to send command and announce the executed results of the command. So, if the command packet is sent and transmitted erroneously, it can be resent immediately, which can omit several minutes of waiting for receiving of the command executed result. Also, the alarms can be sent automatically. Moreover, the commands are sent with priority.

The protocol is also scalable. Because the format of DataField in the protocol packet is specific to command ID, and the function of a system is determined by command IDs and control point identifications, we can add new command IDs and the related formats of DataField and control point identifications to enhance the system functions, or we can choose the set of command IDs and control point identifications to reduce the system functions.

ACKNOWLEDGMENT

The authors thank Dr. Siliang Yang in the USA for carefully revising and editing this paper.

References

Belina, F., Hogrefe, D. and Sarma, A., 1991. SDL with Applications from Protocol Specification. Carl Hansiger Verlag and Prentice Hall International.

Huang, L.C., 1999. SC-SS interconnected protocol of centralized monitoring and controlling system of power supply and air-conditioner (temporary specifications). Zhejiang Post and Telecommunication Management Bureau in China.

Huang, L.C. and Wu, Z.H., 2001. An SC-SS interconnected protocol between heterogeneous and distributed monitoring and controlling systems. Proceedings of the Sixth International Conference for young computer scientist ICYCS2001, International Academic Publishers, p. 188 – 192.

ITU-T, 1992. Message Sequence Charts (MSC). International Standard ITU-T Recommendation Z.120, 1992.

ITU-T, 1995. ITU Recommendation Z.105, SDL Combined with ASN.1 (SDL/ASN.1), 1995.

ITU-T, 1998a. Information Technology-Abstract Syntax Notation One (ASN.1): Specification of Basic Notation., International Standard ITU-T Rec. X.680, 1997 | ISO/IEC 8824-1, 1998.

ITU-T, 1998b. Information technology - ASN.1 encoding rules: Specification of Packed Encoding Rules (PER)., International Standard ITU-T Rec. X.691, 1997 | ISO/IEC 8825-2, 1998.

ITU-T, 2000. ITU-T Recommendation Z.100: Specification and Description Language (SDL), ITU, Geneva.

Khendek, F. and Vincent, D., 2000. Enriching SDL specifications with MSCs. 2nd Workshop on SDL and MSC, Grenoble, p.26 – 28.

Ministry of Posts and Telecommunications of China, 1997. Technological requirements of centralized monitoring and controlling system of telecommunication power supply and air-conditioner (temporary specifications).

Olivier, D., 2002. ASN.1: The solution for efficient communication between systems, http://asn1.elibel.tm.fr/biblio/asn1-presentation_fichiers/frame.htm.

Sidnie, M. F., 1995. SNMP: A Guide to Network Management. New York, McGraw-Hill.

Telecommunication General Bureau of China, 1997a. Centralized monitoring and controlling system of telecommunication power supply and air-conditioner (temporary standards).

Telecommunication General Bureau of China, 1997b. Technological requirements for selecting systems of centralized monitoring and controlling system of telecommunication power supply and air-conditioner.

Telelogic Company, 2002. Telelogic Tau SDL suite, http://www.telelogic.com/