Connection of Computers of Different Architecture to a LAN via a Gateway Process

YOSHIHIKO EBIHARA*, KATSUO IKEDA*, TOMOO NAKAMURA*, MASAMI OGAWA**
and TSUTOMU TAKIGUCHI***

This paper describes a scheme to connect full scale computers of different network architecture to a wideband local area network via a gateway process instead of a front end processor, and outlines the results of its application. This scheme has advantages that no reforming of the native operating system is necessary and that the system development is easy, fast and economical, though the system suffers slight performance reduction compared to a system of which network architecture is newly developed for an existing network. As a case study, a computer with different network architecture has been introduced to the LAN to examine its feasibility of the proposed scheme. For a full scale computer which can run different operating systems concurrently by simulating virtual machine systems, this gateway process scheme offers an efficient way to connect computer systems of different network architecture without badly degrading the man-machine interface response or the performance.

1. Introduction

Research in network interconnection techniques has been motivated by the desire to permit communication among geographically distributed computing resources and users [1]. The purpose of an internetwork architecture is to provide a uniform framework for communication with a heterogeneous computing, communication and application environment [2]. When a new computer of a different network architecture is to be connected to an existing network, we also encounter the same problem of a uniform framework. Two or more networks or computers are connected via a device called a gateway. Gateways may be thought of as having a section for each network or computer they interconnect. Each gateway-section contains procedures to convert from a specific protocol into a standard or other protocol and vice versa. Much research and many projects have been described concerning gateway architectures [1]-[11]. They show different individual gateway architectures. MENEHUNE ARPANET driver [10], implemented on a terminal concertrator of the ALOHA system, was a gateway processor to connect to the ARPANET which appears to each terminal user as a set of computers on the ALOHA system. Rochester's intelligent gateway [11] is a system to connect terminals to a variety of computers and computer networks (ARPANET, ETHERNET). Pup internet [2] connects computer networks by using the technique of encapsulation and decapsulation of a Pup header and tailer in another network's messages.

Here we discuss a scheme to connect a full scale computer of a different network architecture to a wideband local area network [12]-[15] via a gateway process, and the results of implementation of this scheme. An important feature of the gateway process is that most of the interconnection properties are primary artifacts of host software in a general-purpose time-sharing system. The general-purpose time-sharing system provides reasonable response to any user program and has many service facilities and utilities for implementing a gateway process. While Rochester's intelligent gateway and others [2], [10] have usually been implemented on a smaller, less powerful front end processor, front end processors sometimes become bottlenecks for communication throughput for bulk data transmission. The proposed gateway scheme has advantages that the gateway process can accommodate a number of existing terminals attached to the time-sharing system with slight modification of the native operating system and can provide system designers with powerful debugging tools and the rich utilities of the existing system. This scheme is easy, fast and economical for gateway construction, compared to any other gateway technique, for support of virtual terminal services.

Section 2 describes strategy for a gateway scheme and Section 3 discusses interface issues to connect a computer to a LAN. The gateway system structure is discussed in Section 4, protocol conversion issues in Section 5, system integration and verification in Section 6

* Institute of Information Sciences and Electronics University of Tsukuba, Ibaraki 305, Japan
** Facom-Hitac Ltd., Tokyo, Japan
*** Fujitsu Social Science Lab., Tokyo, Japan

and gateway performance in Section 7.

2. Strategy to Connect a Computer to LAN

A gateway scheme is practical architecture to connect a full scale computer to an established computer network at low cost. First, we describe the development environment for system construction and some ristrictions that come from this environment. Then, under the condition of these ristrictions, we discuss the feasibility and effectiveness of the proposed gateway scheme to connect a computer of different network architecture to computer networks.

We often encounter the following cases when a computer of a different network architecture is introduced within an existing computer network environment:

(1) A computer network with the original network architecture has already been in service. Besides, its communication system is fast enough to support a real time operation.

(2) Network architecture of the computer that is to be introduced is different from that of the existing computer network.

The following two development methods seem reasonable for introducing a new computer to an existing computer network:

(1) To replace the original network architecture of the computer with the existing network architecture. This scheme needs to rewrite the OS of the computer to adapt it for the network, thus much work is required for the development of the new OS. Besides, the finished goods will be adaptable only to the existing computer network.

(2) To employ a gateway that executes protocol conversion and faces the network architecture of the new computer and the native computer network architecture. This gateway scheme has advantages that both network architectures remain unchanged and the impact of modifying the OS of the computer on system reliability is much small.

As a result, the gateway scheme would be more practical and more efficient than the first scheme.

It seems to be quite reasonable to consider that there are generally two ways of implementation methods of a gateway as followings:

(1) Gateway realized by a front end processor (FEP)

A FEP scheme has an advantage that it has the least impact on the well-proven reliability of the existing OS of the computer to be connected because most of the gateway functions are implemented in the FEP. While this scheme has a disadvantage that an additional small scale computer is necessary as a FEP. Moreover it may become a bottleneck in real time data communication, depending on the throughput of the FEP. As the FEP resides in between the main computer and the network, the FEP requires twice as many i/o operations, one for the computer, the other for the computer network, as i/o operations for a direct channel connection.

(2) Gateway implemented in a computer

The gateway functions are executed by the gateway processes running in the computer to be connected. The computer is connected to the communication interface of the computer network through a direct channel access. This gateway process method has the following advantages:

(a) Very high speed i/o operation is possible due to the direct channel access which can support the high speed communication of the network. The direct channel access requires a DMA adaptor. However this requirement does not impose a burden on system implemetation because DMA techniques are well established.

(b) System programmer can utilize the good programming environment that has rich computer resources like peripheral devices and utility programs.

(c) Gateway process is designed to be controlled within limited circumstances so that no process has more access rights than required for its operation. Thus, the method will limit error propagation on the OS of the computer.

(d) Decrease of throughtput is fairly small especially when the gateway processes are implemented on a full scale computer.

From the above viewpoints of system implementation, the gateway process method is more practical and useful than other methods.

As a case study, a FACOM M-200 computer is introduced to the existing computer network in order to examine the feasibility and adaptability of the proposed gateway process method, and its architecture and performance are discussed.

3. Interface to Connect a Computer System to LAN

A computer system from a different manufacturer may have its own network protocol because the vendor may have adopted a different network architecture. Access to a remote terminal follows an access method which is generally called a virtual telecommunication access method (VTAM) and is employed in many operating systems. But VTAMs are different and have their own peculiarities which cause difficulties in the building of a heterogeneous network. In order to resolve such differences, a gateway is required to convert and translate protocol between one network and another network. The key point is how to implement the gateway into a computer. We can find several interfaces to insert a module that performs the needed protocol conversion.

3.1 Selection of Interface

Our problem is to offer and link a TSS server function to LAN. Normally, TSS data flows in the sequence of TSS-VTAM-NCP-TERMINAL as shown in Fig. 1(a). NCP is a network control processor that works as a ter-

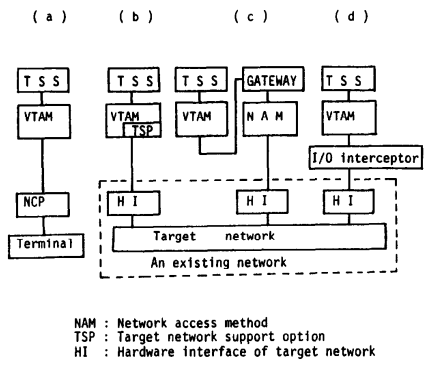

Fig. 1 Interfaces to connect to a network.

minal concentrator. If we don't want to modify an application module such as TSS, it is necessary to adhere to the protocol of the virtual telecommunication (VT) access method. We can find at least three levels of interfaces to link with LAN: VT access method, VTAM internal interface and VTAM i/o. The implications in using these three interfaces are as follows:

1) VT access method

1.1) Direct VTAM support

In order to utilize the VT access method directly, it is necessary to incorporate the network access method (NAM) of the target network into VTAM. In this scheme the native VTAM would include a target network support option as shown in Fig. 1(b), and controls the hardware interfaces of the network as if these were VTAM's local devices. This scheme seems best from the viewpoints of efficiency, structural clarity and initial implementation. It will, however, cause troubles in supporting and maintaining non-standard devices in a VTAM which is a standard system product of a manufacturer.

1.2) VTAM inter-application communication interface

An application process that uses VTAM can communicate with other application processes through the VTAM inter-application process communication facility. The other application process also uses VTAM macroinstructions as TSS. The gateway process can be executed in the normal user mode, thus there are no problems that might degrade system reliability. Fig. 1(c) illustrates the relation of TSS and the gateway processes interconnected through the VTAM inter-application process communication facility.

2) VTAM i/o

In order to utilize the VTAM i/o side interface, it is necessary to intercept VTAM i/o requests as shown in Fig. 1(d). In this scheme the i/o requests that are to be executed by VTAM for terminal control are intercepted and executed by the i/o interceptor; an interface module for the hardware interface of the network. Their control sequences are the basic interface for the terminal and are sometimes mixed up with the control scheme of VTAM. In addition, they are usually executed in a privileged processor mode; thus, the environment for execution and testing is rather complicated and must be prepared specially and carefully, and there is some danger of destroying the important area dedicated to system administration by bugs or failures.

The best interface of the three is the inter-application process communication facility of VTAM.

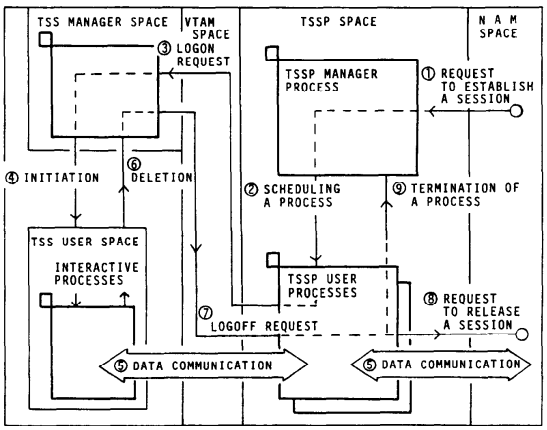

Fig. 2 TSSP structure.

4. System Structure

4.1 TSSP

The gateway process is located between VTAM and NAM, which have almost the same VT access functions; it converts VTAM protocol to NAM protocol and vice versa. As the gateway process is prepared to support TSS, it is called a TSS path-through process (TSSP). The conceptual structure of TSSP is shown in Fig. 2. TSSP consists of a TSSP manager process and a number of TSSP user processes. The TSSP manager process acts as a session controller against NAM, while each TSSP user process is responsible for each user's session control. Roughly speaking, TSSP user process is an application program which accesses (virtual) terminals of the target network through NAM macroinstructions. On the other hand, TSSP user process is also an application process that communicates with a VTAM application process such as a TSS process through the VTAM inter-application process communication facility.

1) TSSP user process as an NAM application process

TSSP user process controls and accesses (virtual) terminals of the target network through NAM macroinstructions. In this respect, TSSP user process works as an ordinary user process to NAM. In order to accept a session establishment request from the TIP subsystem which controls several terminals, a TSSP manager process is prepared. The TSSP manager process works as follows:

(1) A request for session establishment which is passed from NAM to TSSP user process is directed to the TSSP manager process.

(2) The TSSP manager process initiates a TSSP user process for the requesting user.

(3) The TSSP manager process gives a logical link that was originally connected to the TSSP manager process to this TSSP user process.

Then, this TSSP user process can control the interaction with a terminal of the target network. When a TSSP user process is notified of session termination by TSS, it releases the logical link of its process and closes the session. This structure has the following advantages:

(a) Both TSSP user process and TSS user process are kept independent of each other.

(b) The structure of the program is simplified and the development of the system is easier.

The TSSP processes are placed and executed in a single address space as shown in Fig. 2. However, there is no significant difference in development cost between single space and multiple space scheme. This scheme has the following advantages:

(1) Memory space and space management overhead are smaller than in a multiple space scheme.

(2) The TSSP procedure is transparent to users, and no user program runs in the TSSP space, thus there is no danger of sharing a common space after initial bugs are removed.

(3) Many users, enough for a single host to serve, can be supported even in a single space scheme.

2) TSSP user process as a VTAM application process

Each TSSP user process establishes a logical link between TSS user process and itself, and controls the interactive session with TSS through VTAM macroinstructions.

4.2 Function of TSSP User Process

The functions that a TSSP user process is responsible for are as follows:

(1) Control of a session and a logical link

(2) Conversion of terminal control protocol

TSSP user process recognizes the terminal type by the protocol, and connects a link to a VTAM terminal node. It also converts the code set and the protocol for controlling the terminal. It does not, however, handle actual data, but supports a transparent data path between NAM and VTAM.

(3) Data flow control

5. Protocol Conversion Issues

5.1 Basic Policy of Protocol Conversion

The basic policies for protocol conversion are as follows:

(1) Protocol conversion is transparent to users. Thus, the data that are included in a message are not processed.

(2) Throughput must not be degraded badly.

(3) Each link is handled independently so that security and ease of error recovery are maintained.

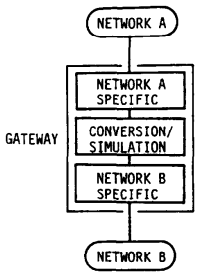

Fig. 3 Gateway structure.

5.2 System Independent Conversion Issues

Generally, a gateway may be thought of as having a half for each network as discussed by Boggs [2]. One could model the operation of a gateway as having each gateway-half contain to convert from a network specific protocol into another network specific protocol and vice versa (Fig. 3). Normally, protocol translation of a gateway is limited to functionality common to interconnected networks. System independent conversion issues are discussed in the following.

(1) Naming and addressing

A name is a symbol which identifies some resource. An address is a data structure which is used to specify the destination of a network. Each network have a local table mapping from names to unique addresses. If address forms are different from each other, a gateway must be concerned with address form translation.

(2) Control code and format conversion

According to each specification of network, a gateway is required to convert message format and translate control codes.

(3) Error control

A variety of error detection and correction are used for networks. Moreover, different recovery procedures are employed from network to network. Thus, a gateway is necessary to handle each different error control and error recovery procedures without any contradiction among different networks architectures.

(4) Flow control and congestion control

Flow control is a mechanism used to regulate the behavior of a specific source and destination pair so that the source does not send data at a rate greater than the receiver can process it. Acknowledgements are employed for flow control. Multiple messages in transit from source to destination are allowed, the number being determined by flow control window. If each network specifies a different window size, a gateway must be concerned about the interaction of each acknowledgement.

Congestion control is a network-wide mechanism,

used to control traffic in the network so as to prevent system overload. Each network has an independent congestion control. In some cases, a gateway as an intermediator is required to handle each congestion control. However, a gateway needs not take care of congetstion control because it is performed normally at a communication subsystem and message loss can be covered, if necessary, by acknowledgement.

(5) Message fragmentation

It is inevitable that some process will want to send a message of one network which is too large to be transmitted through another network that has a smaller maximum message size. This problem is usually approached with message fragmentation and reassembling by a gateway.

(6) Control sequence

A network may have a different control sequence. In this case, a gateway is required to replace message sequence used in one network to that used in another network with the same protocol semantics.

5.3 System Dependent Issues

As a case study, we introduced the FACOM M-200 FNA (Fujitsu network architecture) to the existing network GAMMA-NET [13]-[15]. The proposed gateway can deal with the general conversion issues discussed in the preceding section. However, it still contains the following system dependent issues unsolved.

5.3.1 Implementation Issues

Several problems arise from the actual implementation. These are mainly related to timing or delay, i.e. interrupt handling, abnormal session termination and echo back.

(1) Interrupt handling

Interrupts can be classified into two types; those caused in input mode and in output mode. Most interrupts are casused in output mode, like a BREAK to cause an immediate pause in a listing operation. In our system, response to such an interruption is not as fast as direct connection. TSS/VTAM of FNA sends chained elements in a 256 byte data message in which several lines are accommodated, and there is no straightforward method to know how many lines are to be sent. In addition, there is some delay because of the BREAK request passing. When the protocol converter receives an interrupt it processes the interrupt as one element, but it can not discriminate the proper breaking point as shown in Fig. 4. Therefore the user feels that the processing of BREAK is not effective or not appropriate. To avoid such ambiguity, TSSP must deblock messages from TSS/VTAM of FNA and send them to TIP one by one, but this will degrade throughput badly.

(2) Reuse of process identifier after abnormal session termination

Whenever TSSP manager process detects hardware errors by the terminal or the TIP, the TSSP user process is terminated. TSS, however, does not close the session

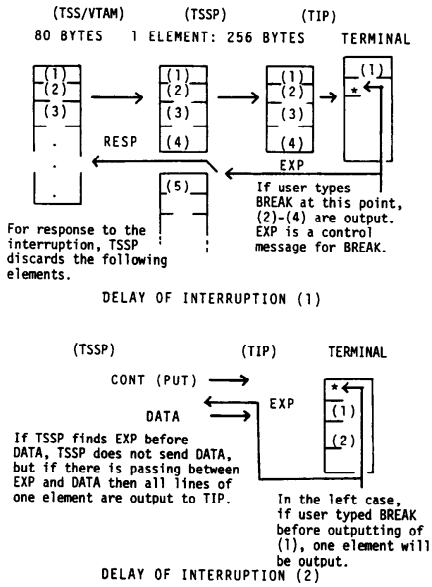

Fig. 4 Delay interruption.

immediately after the hardware errors are detected, but watches status for a given period of time. When a TSSP user process is initiated, an identifier is assigned to it and notified to TSS. If a TSSP user process is abnormally terminated, the assigned identifier is freed by TSSP manager process, but TSS still holds this identifier for a while. If TSSP manager process receives a new request before TSS frees this identifier and if TSSP manager process generates the same identifier, a duplicated assignment condition is caused. The probability of the occurrence of this condition is very low because of low probability of hardware errors. This condition can be avoided by reusing a freed identifier carefully according to the time elapsed since the identifier was freed.

(3) Echo back

Current virtual terminal protocol does not have control over echo back, so that echo suppression for passwords is not available. This problem is left for future work.

6. System Integration and Verification

6.1 Building Block Approach

In the multi-vendor environment of a heterogeneous computer network it is very important to provide a testing methodology which is closed by itself so that extra knowledge and experience are not necessary to carry out system testing. Therefore it is necessary to make a plan to carry out testing and to develop test aids. A building block approach that proceeds stepwise from the simplest environment to the actual operating environment is a reliable way for system integration as shown in Fig. 5.

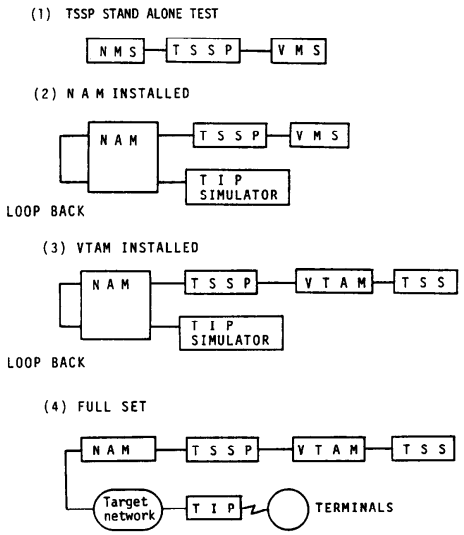

Fig. 5 Test methodology.

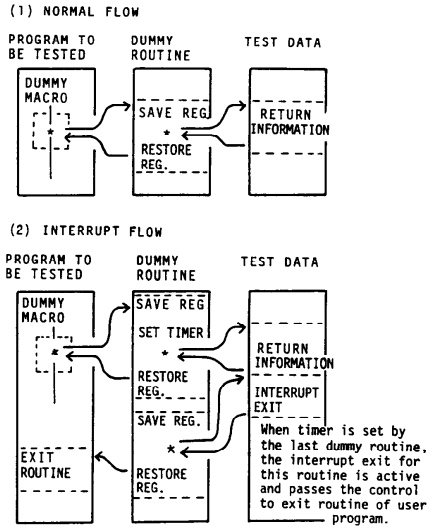

Fig. 6 Basic flow of test tool.

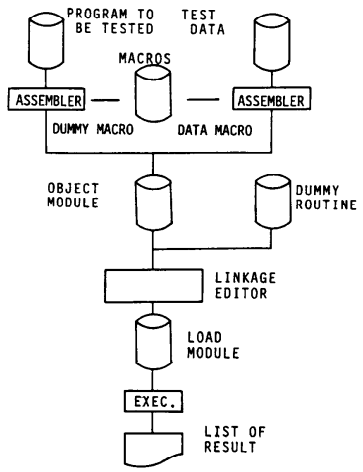

Fig. 7 Test procedure.

(1) The first step is to test TSSP alone, using a VTAM macro simulator (VMS) and NAM macro simulator (NMS). VMS and NMS are test aids that prepare an operating environment for TSSP. VMS issues VTAM macroinstructions to TSSP and accepts messages from TSSP. NMS issues messages to TSSP and accepts NAM macroinstructions from TSSP.

(2) The second step is to test TSSP with NAM, using a TIP simulator. This step really activates NAM by TSSP.

(3) In the third step VTAM is connected and TSSP is tested by the real TSS environment.

(4) The last step is to test TSSP in the real operating environment where all of the system components are included.

6.2 Test Aids

NMS and VMS are very similar in functions and, in due course, in structures because NAM is very similar to VTAM. Test aids should have these characteristics:

(1) It is not necessary to rewrite the source program for testing.

(2) Test data can be described by a macro language.

(3) The interrupt process can be tested easily.

(4) The error cases can be tested easily.

(5) Results of testing are arranged so that they can be examined easily.

In order to avoid rewriting the source program for testing, dummy macroinstructions are substituted for the VTAM and NAM macroinstructions when the test program is assembled, and dummy routines are invoked to simulate the function of VTAM and NAM when the test program is executed. In addition, a test data macro is prepared. This macro is used to specify the return code, to prepare message strings and to pass control to exit routines. Fig. 6 and 7 show basic structures.

7. Discussion of the Gateway Method and Its Performance

7.1 Gateway Process vs. Front End Processor (FEP)

Often a minicomputer is employed as a front end processor to connect a host computer to a network that has different network architecture. Generally, the network control layer and the lower layers are supported by the FEP, and the application layer is supported by the host computer.

The rationale for this scheme is as follows:

(1) To alleviate the load of the host computer for

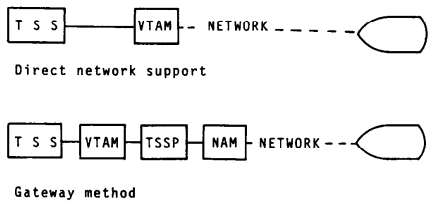

Fig. 8 Comparison of two schemes.

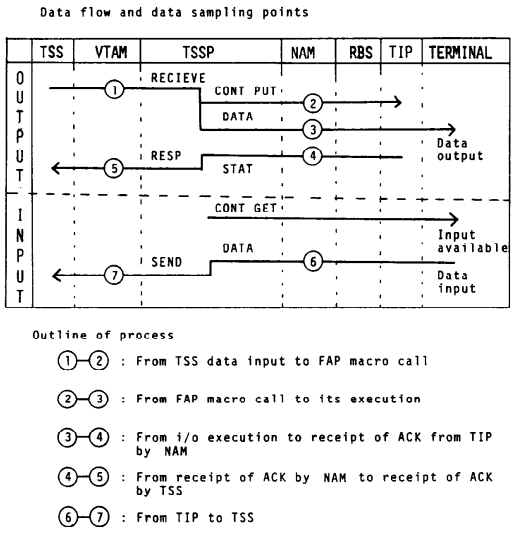

Fig. 9 Data flow and data sampling points.

controlling data links.

(2) To separate control logics between different network architectures, so as to make connection easier.

However this scheme has several drawbacks:

(1) A separate minicomputer system is required.

(2) Productivity of software on a minicomputer is lower than that of a general-purpose full scale computer, which has rich development tools and resources like printers, large file space, editors and utilities.

(3) A FEP method might become a bottleneck of communication especially in the case of a wideband network.

This gateway process method has the following merits:

(1) The existing native operating system can be used efficiently so that the gateway can share file access, file protection and account functions of the native OS.

(2) The full set of advanced utilities for software development is a vailable, thus productivity of software is excellent.

7.2 Performance of the Gateway Process

The most important issue of the gateway process method is how much of the performance degradation is caused by the gateway, compared to the most efficient scheme where VTAM directly controls the network (see Fig. 8). We measured and analized TSSP performance under the following conditions:

(1) OS IV/F4 E40 is employed, and the gateway runs on the advanced virtual machine (AVM).

(2) 10% of the M-200 CPU processing power is allocated to this AVM.

(3) 3MB memory space is allocated to this AVM.

These allocation are not for TSSP but for the AVM. This means that TSSP doesn't always use 10% of the CPU power. Data flow and data sampling points are shown in Fig. 9 and the measured result is shown in Table 1.

Table 1 Input/output processing times

Output processing times

Mean of 10 cases	RECIEVE→CONT→DATA→STAT→RESP ② ① ③ ② ④-③ ⑤-④				No. of charact.
0.207 (Sec.)	0.052	0.048	0.056	0.051	30.9
Ratio 100%	25.1%	23.2%	27.1%	24.6%	

Input processing times

Mean of 2 cases	SEND←DATA ⑦-⑥	No. of charact.
0.01 (Sec.)	0.01	4.5

Average execution time of an M-200 instruction in the on-line environment estimated from the Gibson-mix is about 1.6 to 2.1 microseconds. For an allocation of 10% of the CPU processing power, the numbers of dynamic steps for outward- and inward-data transfer control are estimated as 2400 to 3000 and 470 to 630, respectively. This figure is satisfactory and the performance of TSSP might be concluded excellent. We can conclude that the gateway process method is practical and realizes high efficiency.

8. Conclusion

We discussed the way to connect a computer to a LAN. The system structure and operating conditions of the gateway process method are discussed in detail, including those general issues of selection of interface, system structure, issues of protocol conversion, function of protocol converter as well as the system dependent conversion issues. Measurement results of system performance show that this gateway process method works efficiently and successfully.

Acknowledgements

The authors would like to acknowledge the contributions of Professor Kazuhiko Nakayama and staff of the Science Information Processing Center of the Univer-

sity of Tsukuba and Mr. Shinzawa, Mr. Nishikawa, Mr. Itoh and other staff members of the Mitsubishi Electric Corporation in preparing comfortable environment and encouragement to carry out this research project. The authors also received the technical support from Mr. Tsuru. Mr. Awata, other Fujitsu technical staff members and people of FHL. Mr. Morimatsu and his Iwatsu Software Company technical staff are thanked for their cooperation in system development. Finally, this paper has been greatly improved by the insight and comments of Professor James W. Higgins of the University of Tsukuba.

References

1. POSTEL, J. B. Internetwork Protocol Approaches, *IEEE Trans. on Comm.*, COM-**28** (1980), 604–611.
2. BOGGS, D. B., SCHOCH, J. F., TAFT, E. A. and METACALFE, R. M. Pup: An Internetwork Architecture, *IEEE Trans. on Comm.*, COM-**24** (1980), 612–624.
3. CERF, V. and KAHN, R. A Protocol for Packet Network Intercommunication, *IEEE Trans. on Comm.*, COM-**22** (1974), 637–648.
4. GROSSMAN, G., HINCHLEY, A. and SUNSHINE, C. Issues in International Public Data Networking, *Computer Networks*, **3** (1979), 259–266.
5. DICICCIO, V. SUNSHINE, C., FIELD, J. and MANNING, E. Alternatives for Interconnection of Public Packet Switching Data Networks, *Proc. of Sixth Data Comm. Symp.* (1979), 120–125.
6. MCKENZIE, A. Some Computer Network Interconnection Issues, *Proc. of Nat. Comput. Conf.* (1974), 857–859.
7. WILKES, M. V. The Impact of Wide-Band Local Area Communication Systems on Distributed Computing, *COMPUTER* (1980), 22–25.
8. SUNSHINE, C. Interconnection of Computer Networks, *Computer Networks*, **1** (1977).
9. CERF V. G. and KIRSTEIN, P. T. Issues in Packet-Network Interconnection, *Proc. of IEEE*, **66** (1978), 1386–1408.
10. EBIHARA Y. and WILSON, M. MENEHUNE ARPANET Driver, *ALOHA System Internal Document*, CCG/G-**56** (1974).
11. BALL, J. E., FELDMAN J., LOW, J. R., RASHID, R. and ROVNER, P. RIG, Rochester's Intelligent Gateway: System Overview, *IEEE Trans. on Soft. Eng.*, SE-2 (1976), 321–328.
12. IKEDA, K. EBIHARA, Y. NAKAMURA, T. ISHIZAKA, M. SHINZAWA, M. and NAKAYAMA, K. A Local Computer Network that Uses a Wideband Channel, *Japan Annual Reviews in Electrics, Computers and Telecommunications*, North-Holland, **7** (1980), 1–13.
13. IKEDA, K. EBIHARA, Y. NAKAMURA, T. FUJIMA, T. and NAKAYAMA, K. Computer Network Coupled by 100 MBPS Optical Fiber Ring Bus—System Planning and Ring Bus Subsystem Description-, *Proc. of COMPCON-80 Fall* (1980), 159–165.
14. EBIHARA, Y. IKEDA, K. NAKAMURA, T. ISHIZAKA, M. SHINZAWA M. and NAKAYAMA, K. GAMMA-NET: A Local Computer Network Coupled by High-Speed Optical Fiber Ring Bus—System Concept and Structure-, *Computer Networks*, **7** (1983), 375–388.
15. EBIHARA, Y. IKEDA, K. NAKAMURA, T. NAKATSUKA, S. and ISHIZAKA, M. Fault Diagnosis and Automatic Reconfiguration for a Ring Subsystem, *Computer Networks*, **10** (1985), 97–109.

(Received November 16, 1984; revised October 7, 1986)