システムソフトウェアと オペレーティング・システム 74－1
マルチメディア通信と分散処理 81－1
（1 9 9 7． 2． 2 7）

# Data Transmission for Wide-area Group Communication

**Takayuki Tachikawa and Makoto Takizawa**
**{tachi, taki}@takilab.k.dendai.ac.jp**
**Dept. of Computers and Systems Engineering**
**Tokyo Denki University**

Group communication protocols support the ordered and reliable delivery of messages to multiple destinations in a group of processes. The group communication protocols discussed so far assume that the delay time between every two processes is almost the same. In world-wide applications using the Internet, it is essential to consider a wide-area group where the delay times among the processes are significantly different. We discuss protocols for distributing messages to multiple destinations and retransmitting messages to processes losing the messages in the wide-area group. We present the evaluation of the protocols in the world-wide environment.

# 広域グループ通信におけるデータ転送手続き

立川 敬行　滝沢 誠
東京電機大学理工学部経営工学科

複数プロセスが協調動作を行うためのグループ通信では，メッセージの配送順序と全宛先での受信を保障する必要がある．これまでのグループ通信は，LAN を用いた，伝搬遅延時間の差が小さい環境が考えられてきた．本論文では，インターネットで広域に分散しているプロセス間のグループ通信を考える．このような広域環境では，グループ内のプロセス間でのメッセージの伝搬遅延時間とメッセージの紛失率が地理的な環境により異なる．広域環境で従来のグループ通信プロトコルを適用すると，遅延時間が大きいプロセスに合わす形でグループ全体の転送遅延時間が増加する．本論文では，プロセス間の伝搬遅延時間とメッセージ紛失率が異なるもとでの効率的なグループ通信を行うための再送方式，送信方式を示す．

## 1 Introduction

In distributed applications like teleconferences, a group of multiple processes, i.e. process group [13] is established and the processes in the group are cooperated. Group communication protocols support a group of processes with the reliable and ordered delivery of messages to multiple destinations. Transis [2], ISIS(CBCAST) [4], Psync [17], and others [1,16,22] support the causally ordered delivery. Totem [3], ISIS(ABCAST) [4], Ameoba [12], Trans/Total [15], and others [5,20] support the totally ordered delivery.

Group communication protocols discussed so far assume that every pair of processes have almost the same delay time and reliability. Here, let us consider a world-wide teleconference among five processes $K$, $U$, $O$, $T$, and $H$ in Keele of UK, UCLA and Ohio State Univ. of the USA, and Tohoku Univ. and Hatoyama of Japan, respectively. In the Internet, it takes about 60 msec to propagate a message in Japan while taking about 240 msec between Tokyo and Europe. In addition, the longer the distance is, the more messages are lost. For example, over than 10% of the messages are lost between Japan and Europe while less than 1% are lost in Japan. Thus, it is essential to consider a group communication where the delay times between the processes are significantly different [8,9,11], i.e. not neglectable compared with the processing speed. Such a group of processes is named a *wide-area* group. In the wide-area group, the time for delivering messages to the destinations is dominated by the longest delay between the processes. For example, if $T$ sends a message $m$ to $H$ and $K$, $T$ has to wait for the response from $K$ after having received the response from $H$. Next, suppose that $K$ sends a message $m$ to $H$ and $T$, respectively. If $T$ loses $m$, $T$ requires the sender $K$ to resend $m$. The delay time between $T$ and $K$ is about four times longer than $T$ and $H$. If $H$ resends $m$, the time for retransmitting $m$ can be reduced. Thus, the delivery time can be reduced by the *destination* retransmission.

Suppose that $T$ sends $m$ to $H$, $U$, and $K$. On receipt of $m$, the destination processes send the receipt confirmation messages to $T$. Here, let us consider a way that $K$ sends the confirmation to $U$ instead of directly sending to $T$ and then $U$ sends the confirmation back to $T$. Even if $U$ loses $m$, the delay time can be reduced if $K$ retransmits $m$ to $U$ as presented before. A wide-area group $G$ can be decomposed into disjoint subgroups $G_1$, ..., $G_{sg}$ ($sg \geq 2$) [8,24] where each $G_i$ has one coordinator process. Messages sent by a process are exchanged by the coordinators of the subgroups. Holbrook, et. al. [9] presents a way where each subgroup has a message log to retransmit messages. The protocols [8, 9, 11, 24] are discussed to reduce the number of messages in large-scale groups.

Jones, et. al. [11] discuss the saturation protocol where the sender sends multiple copies of a message $m$ to the destinations. In this paper, we discuss the *destination replication* where the destinations forward $m$ to the other destinations on receipt of $m$.

In sections 2 and 3, we present a system model and the measurement of the delay time and message loss rate in the network. In section 3, we discuss ways to reliably deliver messages in the wide-area group. In section 5, we present protocols in the wide-area group.

## 2 System Model

A distributed system is composed of *application*, *transport*, and *network* layers. A group of $n$ ($\geq 2$) application processes $AP_1$, ..., $AP_n$ are communicated by using the underlying group communication service provided by transport processes $TP_1$, ..., $TP_n$. A group $G$ of the transport processes ($G = \{ TP_1, ..., TP_n\}$) is considered to support each pair of processes $TP_i$ and $TP_j$ with a logical channel. Data units transmitted at the transport layer are *packets*. $TP_i$ sends a packet to $TP_j$ by the channel. The network layer provides the IP service [18] for the transport layer.

The cooperation of the processes at the transport layer is coordinated by *group communication* (GC) and *group management* (GM) protocols. The GC protocol establishes a group $G$ and reliably and causally [4] delivers packets to the processes in $G$. The GM protocol is used for monitoring and managing the membership of $G$. $AP_i$ requests $TP_i$ to send an application message $s$. $TP_i$ decomposes $s$ into packets, and sends them to the destinations in $G$. The destination $TP_j$ assembles the packets into an application message $s_j$, and delivers $s_j$ to $AP_j$. Packets decomposed from the application message are *messages*. Let $dest(m)$ be a set of destination processes of a message $m$ in $G$.

A transport process $TP_i$ has to know the delay time $\delta_{ij}$ with each $TP_j$ in $G$. In the GM protocol, $TP_i$ requests the network layer to transmit two kinds of ICMP [19] packets: "Timestamp" and "Timestamp Reply". $TP_i$ can know when "Timestamp" sent by $TP_i$ is received by $TP_j$, and "Timestamp Reply" received by $TP_i$ is sent by $TP_j$ by using the time information. $TP_i$ calculates $\delta_{ij}$, i.e. round trip time. $TP_i$ sends periodically the ICMP packets to all the processes in $G$. Here, $TP_j$ is *nearer* to $TP_i$ than $TP_k$ if $\delta_{ij} < \delta_{ik}$. In addition, the GM protocol monitors the ratio $\varepsilon_{ij}$ of packets lost between each pair of $TP_i$ and $TP_j$. $\overline{\delta}_{ij}$ and $\overline{\varepsilon}_{ij}$ show the averages of $\delta_{ij}$ and $\varepsilon_{ij}$, respectively. Here, we assume that $\overline{\delta}_{ij} = \overline{\delta}_{ji}$ and $\overline{\varepsilon}_{ij} = \overline{\varepsilon}_{ji}$ for every pair of $TP_i$ and $TP_j$.

We make the following assumptions about packets sent by $TP_i$:

- Packets may be lost and duplicated.
- Packets can be sent to any subset $V$ of destination processes in a group $G$ ($V \subseteq G$).
- Packets sent to $V$ are not received by processes which are not included in $V$.
- Packets sent by the same process may be received by the destination processes not in the sending order (not assuming FIFO arrival).

## 3 Network Measurement

We measure the message loss rates and the delays among seven UNIX processes in SPARC workstations, i.e. three (*ktsun0*, *kelvin*, *ccsun*) in Hatoyama, one (*tu*) in Sendai, Japan, two (*ucla*, *osu*) in the USA, and one (*des*) in Keele, UK. Here, *ktsun0* sends 5000 messages of 128 bytes, one message every one second to *tu*, *ucla*, *osu*, and *des*. Figure 1 shows the receipt ratio $R(t)$ of each process for the delay time $t$. For example, *ucla* receives about 30% of the messages sent by *ktsun0* in 120 to 150 msec. The average delay time $\overline{\delta}_{ktsun0,des}$ is about 240 msec. The figure shows that the longer the distance is, the more messages are lost and the bigger the variance of $R(t)$ is.

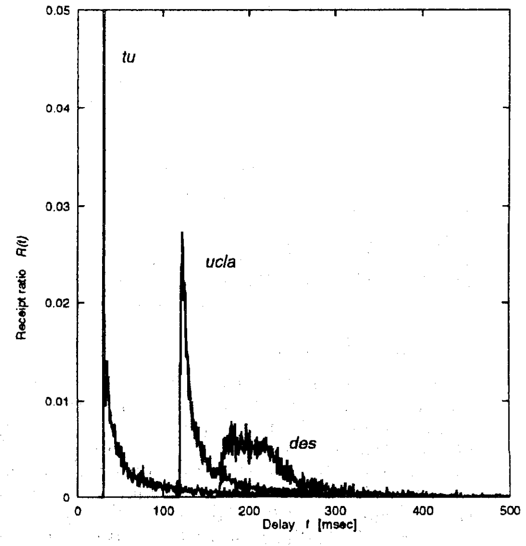

Figure 1: Message receipt ratio v.s. delay

Next, we measure how reliably the destination can receive a message $m$ by the replication. *ktsun0* sends $h$ copies of $m$ to *tu*, *ucla*, and *des*. Table 1 shows a rate that each destination receives at least one copy for $h$ = 1, 2, 3.

Table 1: Replication [%]

| $h$ | 1 | 2 | 3 |
|---|---|---|---|
| tu | 99.08 | 99.96 | 99.96 |
| ucla | 89.74 | 99.16 | 99.84 |
| des | 88.36 | 98.20 | 99.84 |

## 4 Reliable Delivery

### 4.1 Transmission and confirmation

In the group communication, a message $m$ sent by a process $TP_i$ is sent to multiple destination processes in a group $G = \{TP_1, ..., TP_n\}$. Here, let $s$ be the number of the destinations of $m$, i.e. $s = |dest(m)|$. There are two points to be discussed to realize the reliable receipt of $m$ in $G$:

(1) how to deliver $m$ to the destinations, and
(2) how to deliver the receipt confirmation of $m$ to the sender $TP_i$ and the destinations.

There are *direct* and *hierarchical* ways [Figure 2] for (1). In the direct multicast, $TP_i$ sends $m$ directly to all the destinations. In the hierarchical multicast, $TP_i$ sends $m$ to a subset of the destinations. On receipt of $m$ from $TP_i$, $TP_j$ forwards $m$ to other destinations. The propagation tree based routing algorithms [5] are discussed so far. Another example is to decompose $G$ into disjoint subgroups $G_1$, ..., $G_{sg}$ ($sg \geq 2$) [24]. Each $G_i$ has one coordinator process. $TP_i$ sends $m$ to the coordinator and the coordinator forwards $m$ to the destinations in the subgroups.

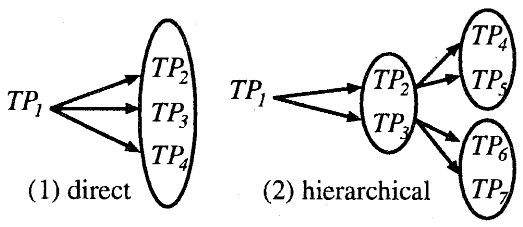

Figure 2: Distribution ways

There are two ways to deliver the confirmation [Figure 3]. In the *decentralized* way [4], $TP_i$ sends $m$ to the destinations and the destinations send back the receipt confirmation of $m$ to $TP_i$. If $TP_i$ receives all the confirmations, $TP_i$ informs all the destinations of the reliable receipt of $m$. Totally $3s$ messages are transmitted and it takes three rounds. In the *distributed* way [20,22], every destination $TP_j$ sends the receipt confirmation of $m$ to all the destinations and $TP_i$ on receipt of $m$. If each $TP_j$ receives the confirmations from all the destinations, $TP_j$ reliably receives $m$. Here, $O(s^2)$ messages are transmitted and it takes two rounds. In the paper [22], the number of messages transmitted in $G$ can be reduced to $O(s)$ by adopting the *piggy back* and the *deferred confirmation*.

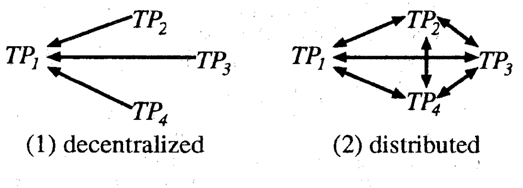

Figure 3: Confirmation ways

There are the following protocols to distribute messages and confirm the receipts:

(1) Direct multicast and distributed confirmation (DD).
(2) Direct multicast and decentralized confirmation (DC).
(3) Hierarchical multicast and distributed confirmation (HD).
(4) Hierarchical multicast and decentralized confirmation (HC).

(1) is named a *distributed* protocol [16]. (2) is adopted by ISIS [4] and others [2, 12, 15].

Next, we consider when each destination process can deliver messages received. Here, let $m_1$ be a message received by $TP_i$. $TP_i$ can deliver $m_1$ if (1) $TP_i$ had delivered every message $m_2$ such that $m_2 \rightarrow m_1$ and (2) $m_1$ is reliably received by all the destinations. How long it takes to reliably receive messages depends on the maximum delay time among the processes in $G$. Hence, the delay for delivering messages is increased if $G$ includes more distant processes. Since the processes are assumed to be not faulty, messages are eventually received by all the destinations. Hence, $TP_i$ can deliver $m_1$ if $TP_i$ delivers every message $m_2$ destined to $TP_i$ such that $m_2 \rightarrow m_1$ even if $m_1$ is not reliably received.

(1) A message $m$ is guaranteed to be buffered by at least one process $TP_j$ in $G$. If $m$ is lost by some process, $TP_j$ can retransmit $m$.
(2) $m$ is removed from the buffer if $m$ is reliably received, i.e. no need to retransmit $m$.

Hence, only destination to retransmit $m$ and the sender of $m$ need to know whether or not $m$ is reliably received. Not necessarily all the destination processes need to buffer $m$.

### 4.2 Retransmission

In the underlying network, messages are lost due to buffer overruns, unexpected delay, and congestion. Hence, the processes have to recover from the message loss. Let us consider a group $R = \{ H, U, O, K \}$. Figure 5 shows a *process graph* of $R$ where each node denotes a process. The weight of the channel $\langle a, b \rangle$ indicates the average delay time $\overline{\delta}_{ab}$. In Figure 5(2), a directed edge $a \rightarrow b$ means that $b$ is the nearest to $a$. Suppose that $H$ sends a message $m$ to $U$, $O$, and $K$, but $O$ fails to receive $m$. In the traditional protocols, the sender $H$ retransmits $m$ to $O$ and it takes $2\overline{\delta}_{HO}$. On the other hand, if $U$ forwards $m$ to $O$, it takes $2\overline{\delta}_{UO}$. Since $\overline{\delta}_{HO} > \overline{\delta}_{UO}$, we can reduce time for retransmission of $m$ if $U$ forwards $m$ to $O$. Thus, if $TP_j$ loses $m$, $TP_k$ whose $\overline{\delta}_k \cdot (1 + \overline{\varepsilon}_{kj}) / (1 - \overline{\varepsilon}_{kj})$ is the minimum can send $m$ to $TP_j$. Thus, there are two ways to retransmit $m$ if $TP_j$ loses $m$.

(1) Sender retransmission: The sender $TP_i$ retransmits $m$ to $TP_j$ [Figure 4(1)].
(2) Destination retransmission: Some destination $TP_k$ forwards $m$ to $TP_j$ [Figure 4(2)].

### 4.3 Replication

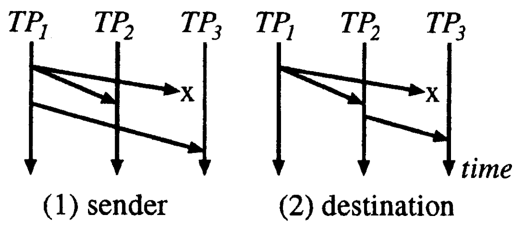

Figure 4: Retransmission

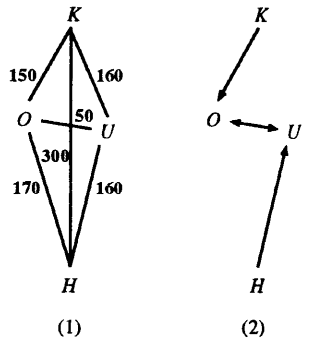

Figure 5: Process graph of the group $H$

In Figure 5, if $H$ sends multiple copies of $m$ to $U$, there is bigger possibility that $U$ can receive one copy of $m$. Thus, one way to prevent from the message loss is that the sender $TP_i$ of $m$ sends multiple copies of $m$ to the destinations. Another way is that a destination $TP_k$ forwards $m$ to another destination $TP_j$ while $TP_i$ sends $m$ to $TP_j$. $TP_j$ receives $m$ from $TP_i$ and $TP_k$. For example, $U$ sends $m$ to $O$ on receipt of $m$ while $H$ directly sends $m$ to $O$. $O$ can receive $m$ from $U$ even if $m$ sent by $H$ is lost. There are the following ways to replicate messages:

(1) Sender replication: $TP_i$ sends multiple copies of $m$ to $TP_j$.
(2) Destination replication: $TP_k$ receiving $m$ sends $m$ to $TP_j$.

The protocols with the replication are named *replicated* protocols. The sender replication is similar to the saturation protocol [11].

There are two kinds of the destination replication. In the first way, one destination $TP_k$ sends one, possibly multiple copies of $m$ to $TP_j$ on receipt of $m$ while $TP_i$ sends $m$ to $TP_j$ [Figure 6(1)]. In another way, multiple destination processes, say $TP_k$ and $TP_l$ send copies to $TP_j$ [Figure 6(2)]. $TP_j$ receives multiple copies from $TP_i$ and $TP_k$.

In the replication, $TP_i$ sends multiple copies of $m$ to $TP_j$. The ***continuous*** replication is to send the copies of $m$ continuously to $TP_j$. There is no gap between messages. In another ***discrete*** replication, $TP_i$ sends the succeeding copies of $m$ some time units after $TP_i$ sends each copy of $m$. If the message loss occurs in a burst manner, the retransmission with the discrete replication has

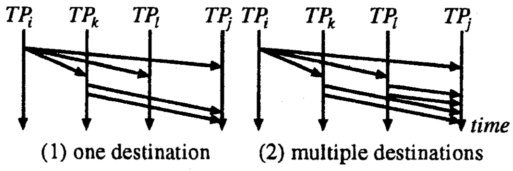

Figure 6: Destination replication

to be adopted.

Suppose that $TP_i$ sends $h_{ij}$ copies of $m$ to $TP_j$ in the sender replication. If $TP_j$ receives no copy of $m$ from $TP_i$, $TP_i$ sends $h_{ij}$ copies of $m$ to $TP_j$ again. The expected time $T_{ij}$ for $TP_j$ to receive at least one copy of $m$ from $TP_i$ is $\delta_{ij} \cdot (1 + \bar{\varepsilon}_{ij}^{h_{ij}}) / (1 - \bar{\varepsilon}_{ij}^{h_{ij}})$. The probability $P_{ij}$ that $TP_j$ receives at least one copy of $m$ is $1 - \bar{\varepsilon}_{ij}^{h_{ij}}$. The cost $C_{ij}$ for $TP_i$ to send $m$ to $TP_j$ is defined to be $h_{ij} \cdot \delta_{ij}$. In the destination replication, $TP_i$ sends $h_{ik}$ copies of $m$ by the sender replication. On receipt of at least one copy of $m$, $TP_k$ sends $h_{kj}$ copies of $m$ to $TP_j$. The expected time $T_{ikj}$ for $TP_j$ to receive at least one copy of $m$ forwarded by $TP_k$ is $\bar{\delta}_{ik} \cdot (1 + \bar{\varepsilon}_{ik}^{h_{ik}}) / (1 - \bar{\varepsilon}_{ik}^{h_{ik}}) + \bar{\delta}_{kj} \cdot (1 + \bar{\varepsilon}_{kj}^{h_{kj}}) / (1 - \bar{\varepsilon}_{kj}^{h_{kj}})$. The cost $C_{ikj}$ is $h_{ik} \cdot \delta_{ik} + h_{kj} \cdot \delta_{kj}$. The probability $P_{ikj}$ that $TP_j$ receives at least one copy of $m$ is $(1 - \bar{\varepsilon}_{ij}^{h_{ij}}) + \bar{\varepsilon}_{ij}^{h_{ij}} \cdot (1 - (1 - \bar{\varepsilon}_{ik}^{h_{ik}}) \cdot (1 - \bar{\varepsilon}_{kj}^{h_{kj}}))$. The destination replication is more efficient than the sender replication if $T_{ij} \geq T_{ikj}$, $P_{ij} \leq P_{ikj}$, and $C_{ij} \geq C_{ikj}$. In the time critical applications, the destination one can be adopted if $T_{ij} \geq T_{ikj}$ even if $C_{ij} \leq C_{ikj}$. $TP_k$ is selected to forward $m$ to $TP_j$ if $T_{ikj}$ is the minimum among the destinations of $m$. Here, no sender replication is adopted if $h_{ij} = 1$ and $h_{ik} = 1$.

Let $\varepsilon$ be a maximum allowable loss ratio. If $\bar{\varepsilon}_{ij} \leq \varepsilon$, $TP_i$ does not need to send multiple copies of $m$ to $TP_j$. If $\bar{\varepsilon}_{ij} > \varepsilon$, $TP_i$ has to send $h_{ij}$ ($> 1$) copies of $m$ whether or not $TP_i$ retransmits $m$. It is required that $\bar{\varepsilon}_{ij}^{h_{ij}} \leq \varepsilon$. Hence, $h_{ij} \geq$ $log\varepsilon$ / $log\bar{\varepsilon}_{ij}$ if $\bar{\varepsilon}_{ij} \leq \varepsilon$, otherwise $h_{ij} = 1$. For example, $h_{ij} = 3$ if $\varepsilon = 0.01$ and $\bar{\varepsilon}_{ij} = 0.2$. We can consider a way that $TP_i$ sends $m$ to $TP_l$, $TP_l$ sends $m$ to $TP_k$, and finally $TP_k$ sends $m$ to $TP_j$. However we assume that there is only one process $TP_k$ forwarding a copy of $m$ to $TP_j$.

## 5 Protocols

Suppose that a process $TP_i$ sends $m$ to a subset $V_m$ of the destination processes in the group $G$. There are the following protocols.

(1) Basic (B) protocol: distributed protocol with sender retransmission.
(2) Modified (M) protocol: distributed protocol with destination retransmission.
(3) Nested group (N) protocol: hierarchical multicast and decentralized confirmation with destination retransmission.

(4) Decentralized (D) protocol: direct multicast and decentralized confirmation with sender retransmission.

There are two further types, i.e. replicated and non-replicated ones for each protocol. Here, BR, MR, NR, and DR are replicated versions of B, M, N, and D protocols.

**[Basic (B) protocol]**

(T1) $TP_i$ sends $m$ to every destination process in $V_m$ $(\subseteq G)$.

(T2) On receipt of $m$, each process $TP_j$ in $V_m$ sends the receipt confirmation to $TP_i$.

(T3) On receipt of the confirmations from all the processes in $V_m$, $TP_i$ reliably receives $m$.

(R) If some $TP_j$ fails to receive $m$, $TP_i$ sends $m$ to $TP_j$ again. □

In the BR protocol with the sender replication, $TP_i$ sends $h_{ij}$ copies of $m$ to $TP_j$. In the BR protocol with destination replication, a destination $TP_k$ sends $h_{ij}$ copies of $m$ to $TP_j$ on receipt of $m$.

The modified (M) protocol is the same as B except that the destination retransmission is adopted.

**[Modified (M) protocol]**

(R) If $TP_j$ fails to receive $m$, destination $TP_k$ nearest to $TP_j$ sends $m$ to $TP_j$. If all the destinations lose $m$, T1 is executed again. □

In the N protocol, $G$ is decomposed into disjoint subgroups $G_1$, ..., $G_{sg}$ $(sg \geq 2)$. Each $G_i$ is composed of the processes $TP_{i1}$, ..., $TP_{il_i}$ $(l_i \geq 1)$ where $TP_{i1}$ is a coordinator.

**[Nested group (N) protocol]**

(T1) $TP_{ij}$ sends $m$ to the coordinator $TP_{i1}$. Let $DC_i$ be a set of the coordinators whose subgroups include the destinations of $m$. $TP_{i1}$ forwards $m$ to the coordinators in $DC_i$.

(T2) On receipt of $m$, the coordinator $TP_{k1}$ sends $m$ to the destinations in $G_k$. On receipt of $m$, $TP_{kh}$ sends the confirmation back to $TP_{k1}$. On receipt of the confirmations from all the destinations in $G_k$, $TP_{k1}$ sends the confirmation to the coordinators in $DC_i$.

(T3) On receipt of the confirmations from all coordinators in $DC_i$, $TP_{k1}$ sends the confirmation to the destinations in $G_k$. On receipt of the confirmation from $TP_{k1}$, $TP_{kh}$ reliably receives $m$.

(R) If $TP_{kh}$ fails to receive $m$, $TP_{k1}$ resends $m$ to $TP_{kh}$. □

In the D protocol, only the sender $TP_i$ can know whether each destination receives $m$ or not. Hence, the sender retransmission is adopted. T1 and R are the same as the B protocol.

**[Decentralized (D) protocol]**

(T2) On receipt of $m$, $TP_j$ sends the confirmation back to $TP_i$.

(T3) On receipt of all the confirmations, $TP_i$ sends the acceptance $a$ to all the processes in $B$.

(T4) On receipt of $a$, $TP_j$ accepts $m$. □

Figure 7(1), (2), and (4) show the B, M, and D protocols where $H$ sends a message $m$ to $U$, $O$, and $K$ but $K$ loses $m$. In the M protocol, $O$ forwards $m$ to $K$ since $O$ is the nearest to $K$. In the replicated version of D, $O$ forwards $m$ to $K$ on receipt of $m$ from $H$. Here, $\overline{\varepsilon}_{OK} > \overline{\varepsilon}_{HK}$ and $\overline{\delta}_{OK} < \overline{\delta}_{HK}$. Figure 7(3) shows the N protocol with three subgroups $\langle H \rangle$, $\langle U, O \rangle$, and $\langle K \rangle$ where $H$, $U$, and $O$ are the coordinators. $U$ receives $m$ but $O$ loses $m$. Here, $U$ resends $m$ to $O$.

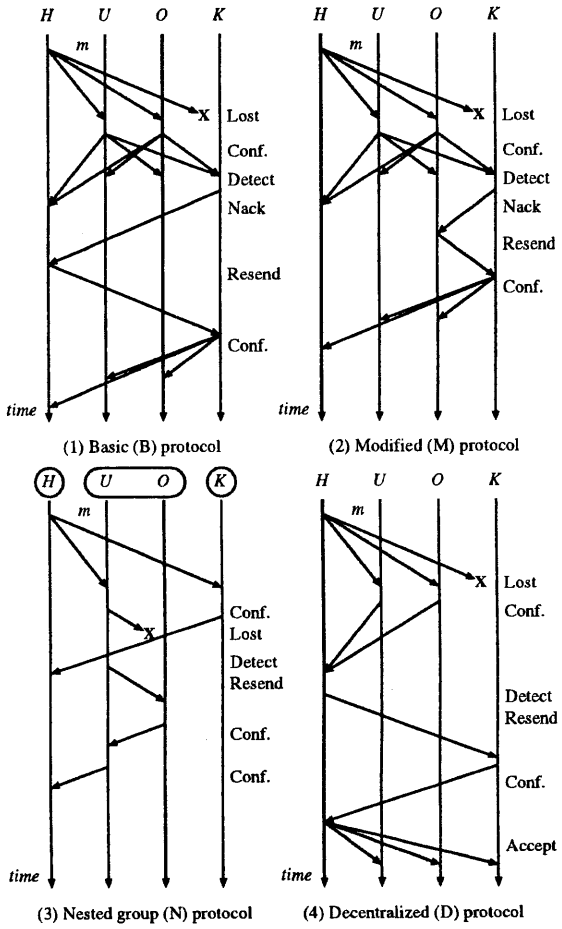

Figure 7: Protocols

In the B and D protocols, only $H$ is required to buffer $m$ since $H$ retransmits $m$. All the processes may retransmit $m$. Hence, every process has to buffer $m$. In the N one, $H$ sends $m$ only to $U$, and then $U$ forwards $m$ to $O$ and $K$. If either $O$ or $K$ loses $m$, $U$ retransmits $m$. Hence, the coordinators have to have buffers. The B and D ones imply fewer buffers than the others.

## 6 Concluding Remarks

We have discussed the wide-area group communication including multiple processes interconnected by the Internet. Here, each logical channel between the processes has a different delay time and message loss ratio. In this paper, we have presented ways to reduce the delay time of messages and improve the reliability in the wide-area group, i.e. destination retransmission and replication. We have presented four protocols, i.e. basic, modified, nested group, and decentralized protocols. We are planning to evaluate the protocols in the wide-area environment including processes in Dendai, Tohoku Univ., UCLA, OSU, and Keele.

## Acknowledgment

We would like to thank Prof. Lewis A. Davis, Tokyo Denki Univ. for his useful help and encouragement in writing this paper. We also would like to thank Prof. N. Shiratori, Tohoku Univ., Prof. M. Liu, OSU, Prof. M. Gerla, UCLA, and Prof. S. M. Deen, Keele Univ. for their cooperation of the internetwork experiment.

## References

[1] Aiello, R., Pagani, E., and Rossi, G. P, "Causal Ordering in Reliable Group Communications," *ACM. SIGCOMM'93*, 1993, pp.106–115.

[2] Amir, Y., Dolev, D., Kramer, S., and Malki, D., "Transis: A Communication Sub-System for High Availability," *Proc. of IEEE FTCS-22*, 1993, pp.76–84.

[3] Amir, Y., Moser, L. E., Melliar-Smith, P. M. Agarwal, D. A., and Ciarfella, P., "The Totem Single-Ring Ordering and Membership Protocol," *Trans. on ACM Computer Systems*, Vol.13, No.4, 1995, pp.311–342.

[4] Birman, K., Schiper, A., and Stephenson, P., "Lightweight Causal and Atomic Group Multicast," *ACM Trans. Computer Systems*, Vol.9, No.3, 1991, pp.272-314.

[5] Chang, J. M. and Maxemchuk, N. F., "Reliable Broadcast Protocols," *ACM Trans. Computer Systems*, Vol.2, No.3, 1984, pp.251–273.

[6] Florin, G. and Toinard, C., "A New Way to Design Causally and Totally Ordered Multicast Protocols," *ACM Operating Systems Review*, Vol.26, No.4, 1992, pp.77–83.

[7] Garcia-Molina, H. and Spauster, A., "Ordered and Reliable Multicast Communication," *ACM Trans. Computer Systems*, Vol.9, No.3, 1991, pp.242–271.

[8] Hofmann, M., Braun, T., and Carle, G., "Multicast Communication in Large Scale Netwoks," *Proc. of 3rd IEEE Workshop on High Performance Communication Subsystems(HPCS)*, 1995.

[9] Holbrook, H. W., Singhal, S. K., and Cheriton, D. R., "Log-Based Receiver-Reliable Multicast for Distributed Interactive Simulation," *Proc. of ACM SIGCOMM'95*, 1995, pp 328–341.

[10] Jia, X., "A Total Ordering Multicast Protocol Using Propagation Trees," *IEEE Trans. Parallel and Distributed Systems*, Vol.6, No.6, 1995, pp.617–627.

[11] Jones, M., Sorensen, S., and Wilbur, S., "Protocol Design for Large Group Multicasting: The Message Distribution Protocol," *Computer Communications*, Vol. 14 No. 5, 1991 pp.287–297.

[12] Kaashoek, M. F., Tanenbaum, A. S., Hummel, S. F., and Bal, H. E., "An Efficient Reliable Broadcast Protocol," *ACM Operating Systems Review*, Vol.23, No.4, 1989, pp.5–19.

[13] Liang, L., Chan, S. T., and Neufeld, G. W., "Process Groups and Group Communications: Classifications and Requirements," *IEEE Computer*, Vol.23, No.2, 1990, pp.56–66.

[14] Mattern, F., "Virtual Time and Global States of Distributed Systems," *Parallel and Distributed Algorithms* (Cosnard, M. and Quinton, P. eds.), *North-Holland*, 1989, pp.215–226.

[15] Melliar-Smith, P. M., Moser, L. E., and Agrawala, V., "Broadcast Protocols for Distributed Systems," *IEEE Trans. on Parallel and Distributed Systems*, Vol.1, No.1, 1990, pp.17–25.

[16] Nakamura, A. and Takizawa, M., "Causally Ordering Broadcast Protocol," *Proc. of IEEE ICDCS-14*, 1994, pp.48–55.

[17] Peterson, L. L., Buchholz, N. C., and Ghemawat, S., "Preserving and Using Context Information in Interprocess Communication," *ACM Trans. on Computer Systems*, Vol.7, No.3, 1989, pp.217–246.

[18] Postel, J., "Internet Protocol," *RFC-791*, 1981.

[19] Postel, J., "Internet Control Message Protocol," *RFC-792*, 1981.

[20] Tachikawa, T. and Takizawa, M., "Selective Total Ordering Broadcast Protocol," *Proc. of IEEE ICNP-94*, 1994, pp.212–219.

[21] Tachikawa, T. and Takizawa, M., "Multimedia Intra-Group Communication Protocol," *Proc. of IEEE HPDC-4*, 1995, pp.180–187.

[22] Tachikawa, T. and Takizawa, M., "Distributed Protocol for Selective Intra-group Communication," *Proc. of IEEE ICNP-95*, 1995, pp.234–241.

[23] Tachikawa, T. and Takizawa, M., "Communication Protocol for Wide-area Group," *Proc. of the International Computer Symposium (ICS'96)*, 1996, pp. NM 158–165.

[24] Takamura, A., Takizawa, M., and Nakamura, A., "Group Communication Protocol for Large Group," *Proc. of IEEE Conf. on Local Computer Networks (LCN-18)*, 1993, pp.310–319.