Multimedia Group Communication *

3L−04

Kenichi Shimamura, Katsuya Tanaka, and Makoto Takizawa †
Tokyo Denki University ‡

Abstract

In distributed applications like teleconferences, a group of multiple processes are cooperating, where messages exchanged among the processes are required to be causally delivered. The processes are exchanging kinds of multimedia objects in addition to traditional text data. The multimedia objects are longer than traditional messages and are structured. In this paper, we discuss new types of causally precedent relations among multimedia objects transmitted in the network. We discuss a protocol to causally deliver multimedia objects in a group of processes.

1 Introduction

In distributed applications, a group of multiple processes are cooperating. Various kinds of group protocols are discussed so far. In the group communication, a *group* is first established among multiple processes and then messages sent by the processes are *causally*, *totally* delivered to the destination processes in the group. A message m_1 *causally precedes* another message m_2 if a sending event of m_1 *happens before* a sending event of m_2. In the totally ordered delivery, even messages not to be causally ordered are delivered to every common destination of the messages in a same order. In the protocols, messages transmitted at the network level are ordered independently of what kinds of information are included in the messages.

In distributed applications, various kinds of multimedia objects like image and video are exchanged among multiple processes in the group. As discussed in MPEG-4, each multimedia object is composed of component objects. Thus, multimedia objects are structured and are larger than the traditional data messages. In addition to causally delivering objects, a multimedia object received has to satisfy quality of service (QoS) like frame rate and number of colors required by the destination processes.

An object is decomposed into a sequence of messages in order to transmit the object in a network. A message is a unit of data transmitted in the network. If a pair of objects o_1 and o_2 are transmitted by processes p_1 and p_2, respectively, the messages decomposed from o_1 and o_2 are causally delivered to every common destination process p_3 of o_1 and o_2 according to the traditional group protocols. The messages of the object o_1 can be delivered independently of the object o_2 if o_1 is manipulated independently of o_2 in an application. In another application, the top message of the message sequence of the object o_1 is required to be delivered before the top of o_2 while the other messages can be delivered in any order. Thus, we define new types of precedent relations named *O-precedent* relations of messages based on the object concept. According to the precedent relations, the destination process delivers messages of objects to the application. A pair of messages not to be ordered in the precedent relations can be delivered in any order even if one of the messages causally precedes the other according to the traditional network-level destination. We discuss a protocol which supports the types of causally precedent relations, named *causally ordered multimedia* (*COM*) group protocol, where a fewer number of messages are causally ordered than the traditional network-level group protocols.

2 System Model

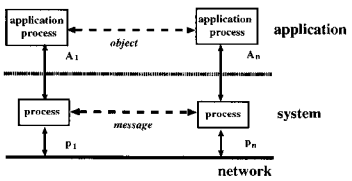

Figure 1: System layers.

Distributed applications are realized by cooperation of a group of application processes A_1, ..., A_n ($n \geq 1$). Application processes exchange objects including multimedia data with the other processes in the group by using the network.

An application process A_t is supported by a system process p_t ($t = 1, ..., n$). p_t takes an object from the application process A_t and then delivers the object to the system processes supporting the destination application processes by using the basic communication service supported by the network. From here, let a term *process* mean a system process.

A data unit exchanged among the processes is referred to as *message*. We assume the network supports processes with synchronous communication. That is, messages are not lost and maximum delay time between every pair of processes is bounded in the network. In our implementation, a transport protocol lik TCP is used as the network service.

An object o is decomposed into a sequence $\langle m_1, ..., m_h \rangle$ of messages by a source process and the messages are delivered to the destination processes. Here, m_1 is the top message and m_h is the last message of the object o. A destination process p_t assembles received messages into an object and then delivers the object to the application process A_t. The cooperation of the processes supporting the group of the application processes is coordinated by a *group protocol* which supports the reliable, efficient communication service by taking usage of the network service. We discuss a group protocol for delivering multimedia objects to processes in a group.

3 Object-Precedency

We discuss how a process sends and receives multimedia objects in a group G of multiple processes p_1, ..., p_n ($n > 1$). In order to increase the throughput and reduce the

*マルチメディアグループ通信
†島村 健一, 田中 勝也, 滝沢 誠
‡東京電機大学

response time, sending and receiving events of objects are interleaved if there is no precedent relation among objects.

As presented here, a way on how a pair of objects o_1 and o_2 are interrelated depends on when processes p_s and p_t start and finish sending and receiving the objects. We discuss how a pair of objects o_1 and o_2 can be causally ordered. Let $ss_t(o)$ and $es_t(o)$ denote events that a process p_t starts and finishes sending an object o, respectively. In fact, $ss_t(o)$ and $es_t(o)$ show events that the top and last messages of the object o are sent by p_t, respectively. $sr_t(o)$ and $er_t(o)$ also mean the receipt events of the top and last messages of the object o, respectively. Let $sr_t(o)$ and $er_t(o)$ denote events that p_t starts and finishes receiving the object o.

[Definition] Let o_1 and o_2 be a pair of objects o_1 and o_2 sent by processes p_s and p_t, respectively:

1 o_1 *top-precedes* o_2 ($o_1 \rightarrow o_2$) iff
   - $\diamond$ $sr_t(o_1)$ happens before ($\prec$) $ss_t(o_2)$ if $p_s \neq p_t$.
   - $\diamond$ $ss_s(o_1) \prec ss_t(o_2)$ if $p_s = p_t$.

2 o_1 *tail-precedes* o_2 ($o_1 \rightarrow o_2$) iff
   - $\diamond$ $er_t(o_1) \prec es_t(o_2)$ if $p_s \neq p_t$.
   - $\diamond$ $es_s(o_1) \prec es_t(o_2)$ if $p_s = p_t$.

3 o_1 *partially precedes* o_2 ($o_1 \rightarrow o_2$) iff $o_1 \rightharpoonup o_2$, $o_1 \rightharpoondown o_2$, and o_1 is interleaved with o_2 ($o_1 \| o_2$).

4 o_1 *fully precedes* o_2 ($o_1 \Rightarrow o_2$) iff
   - $\diamond$ $er_s(o_1) \prec ss_t(o_2)$ if $p_s \neq p_t$.
   - $\diamond$ $es_s(o_1) \prec ss_t(o_2)$ if $p_s = p_t$.

5 o_1 *inclusively precedes* o_2 ($o_1 \supset o_2$) iff $o_1 \rightharpoonup o_2$ and $o_1 \rightharpoondown o_2$.

6 o_1 *exclusively precedes* o_2 ($o_1 \sqsupset o_2$) iff $o_1 \rightharpoondown o_2$ and $o_2 \rightharpoonup o_1$.□

The top, tail, fully, partially, inclusively, and exclusively precedent relations defined here are referred to as *object-causally precedent* (*O-precedent*) relation. Here, $o_1 \rightsquigarrow o_2$ shows that o_1 *O*-precedes o_2, i.e. $\rightsquigarrow \in \{\rightharpoonup, \rightharpoondown, \Rightarrow, \rightarrow, \supset, \sqsupset\}$. The process p_u is required to deliver messages of objects o_1 and o_2 so as to satisfy the *O*-precedent relation $\rightsquigarrow$ between o_1 and o_2.

4 COM Protocol

We present a *causally ordered multimedia* (*COM*) protocol for supporting the *O-precedent* delivery of multimedia objects for a group G of multiple processes p_1, ..., p_n ($n > 1$).

A message m sent by a process p_s carries a sequence number *seq*. *seq* is incremented by one each time p_s sends a message. Here, it is noted again that each process can simultaneously send multiple objects. Two types of vectors of variables $V = \langle V_1, ..., V_n \rangle$ and $A = [A_1, ..., A_n]$ are manipulated for each process p_t in the group G. A pair of the vectors V and A are manipulated in a way similar to the vector clock.

Initially, $V = \langle 0, ..., 0 \rangle$ and $A = [0, ..., 0]$ in every process. First, suppose a process p_t starts sending a segment of an object o. That is, a *syn* message of o the object o is sent. Here, the t-th elements V_t and A_t of the vectors V and A are incremented by one:

$\bullet V_t := V_t + 1;$ $\bullet A_t := A_t + 1;$

The process p_t eventually finishes sending a segment of an object o. Only the variable A_t is incremented by one when p_t finishes sending an object o. However, V_t is not changed.

- $A_t := A_t + 1;$

Thus, V_t and A_t show how many segments of objects a process p_t starts sending and how many segments of objects p_t starts and finishes sending, respectively. Here, let $o.SA$ and $o.SV$ show values of the vectors A and V, respectively, when p_t starts sending an object o. Let $o.EV$ and $o.EA$ show the values of the vectors V and A, respectively, when p_t finishes sending the object o. Hence, let $o.V$ and $o.A$ be variables showing the values of the vector V and A of the object o, respectively. $o.V = o.SV$ and $o.A = o.SA$ since p_t starts sending messages of the object o. $o.V = o.EV$ and $o.A = o.EA$ since p_t finishes sending messages of o. The object o carries the vector information $o.V$ and $o.A$ to the destination processes. If each message of the object o carries the current values of $o_1.V$ and $o_1.A$, the communication overheads are increased. In order to reduce the communication overheads, $o.SV$ and $o.SA$ are carried by a top message of the object o. That is, $m.V = o.SV$ and $m.A = o.SA$. Every message m following the top message is considered to carry $m.V = o.SV$ and $m.A = o.SA$. The value $o.EA$ is carried by a last message of the object o. Some messages may be lost due to unexpected delay in the network. In order to increase the reliability, $o.SV$ and $o.SA$ can be carried by multiple messages, e.g. the top message and one message after the top. $o.EV$ and $o.EA$ can also be carried by multiple messages.

On receiving a top message of an object o from a process p_s, variables V and A are manipulated in a process p_t as follows:

- $V_s := \max(V_s, o.SV_s)$ ($s = 1, ..., n, s \neq t$);
- $A_s := \max(A_s, o.SA_s)$ ($s = 1, ..., n, s \neq t$);

On receiving a last message of the object o, the variable A is changed as follows:

- $A_s := \max(A_s, o.EA_s)$ ($s = 1, ..., n, s \neq t$);

The following properties among the *object-precedent* (*O*-precedent) relations and the vectors hold:

[Theorem] Suppose a process p_s sends an object o_1 to a pair of process p_t and p_u, and another process p_t sends another object o_2 to p_u.

- $o_1 \Rightarrow o_2$ iff $o_1.EA_v \leq o_2.SA_v$(v=1, ..., n, $v \neq$s).
- $o_1 \rightharpoonup o_2$ iff $o_1.SV_v \leq o_2.SV_v$(v=1, ..., n, $v \neq$s).
- $o_1 \rightharpoondown o_2$ iff $o_1.EA_v \leq o_2.EA_v$(v=1, ..., n, $v \neq$s).
- $o_1 \rightarrow o_2$ iff $o_1.EA_v \geq o_2.SA_v$, $o_1.EA_v < o_2.EA_v$, and $o_1.SV_v \leq o_2.SV_v$(v=1, ..., n, $v \neq$s).
- $o_1 \supset o_2$ iff $o_1.SV_v \leq o_2.SV_v$ and $o_2.EA_s \leq o_1.EA_s$ (v=1, ..., n, $v \neq$s).
- $o_1 \sqsupset o_2$ iff $o_1.EA_v \leq o_2.EA_v$ and $o_1.SV_s > o_2.SV_s$. (v=1, ..., n, $v \neq$s).□

The objects received are ordered by using the vectors V and A according to the rules on the vectors presented in the theorem.

5 Concluding Remarks

We defined novel types of causally precedent relations among multimedia objects, i.e. top ($\rightarrow$), tail ($\rightharpoondown$), partially ($\rightharpoonup$), fully ($\Rightarrow$), inclusive ($\supset$), and exclusive ($\sqsupset$) precedent relations.

References

[1] Shimamura, K., Tanaka, K., and Takizawa, M., "Group Protocol for Exchanging Multimedia Objects in a Group," *Proc. of ICDCS Int'l Workshop on Group Communications and Computations* (*IWGCC*), 2000, C33–C40.