単一チャネル通信網上の全順序優先度順放送通信 (PriTO) プロトコル

中村章人† 滝沢誠†

グループウェアシステム等の，複数の応用エンティティが協調動作を行う分散型応用システムでは，二つのエンティティ間の1対1通信に加えて，複数の宛先にデータを転送するための，高信頼放送通信サービスが必要となる．また，応用によっては，テキスト，音声，動画等の複数種類のデータ転送を，同時に行う必要がある．このための方法として，各プロトコルデータ単位 (PDU) に優先度を与え，優先度の高いものを，低いものより先に宛先に届けることがある．本論文では，分散型応用システムを構成する応用エンティティの集合を群とし，群内の全応用エンティティが，放送されたPDUを紛失なく，優先度に基づく同一の順序で受信する優先放送通信サービスを定義し，このサービスを提供するためのプロトコルを提案する．優先度に基づいてPDUを配送するサービスでは，低優先度のPDUが最悪の場合，無限に待ち続けてしまう問題がある．本論文では，この問題を解決するために，連の概念を導入する．連は，優先度順に整列されたPDUの系列であり，応用エンティティは，連の系列としてPDUを優先度順に受信する．一定時間内に応用に渡されないPDUを検出した場合，そのPDUを含む連を終了し，新たな連を開始することで，この問題を解決する．

Priority-Based Total Ordering Broadcast (PriTO) Protocol on the One-Channel Network

Akihito Nakamura† and Makoto Takizawa†

In distributed applications like groupware systems, it is required to provide reliable broadcast service, by which application entities send data to multiple destinations reliably and efficiently. Also, various kinds of data like text, voice, and video may be exchanged. In such kinds of applications, some protocol data units (PDUs) have to be delivered to destinations earlier than another PDUs. One approach to providing such communication service by using a single channel is to give a priority to each PDU and to deliver higher-priority PDUs to the destinations earlier than lower-priority ones. In this paper, we discuss distributed broadcast protocols which provide priority-based receipt ordering of PDUs for entities by using a single channel system like Ethernet and radio systems. In the priority-based ordering service, there is a *starvation* problem, i. e. lower-priority PDUs can be left waiting indefinitely in the receipt queue since higher-priority PDUs jump over the lower-priority PDUs. In this paper, we present a method by which even lower-priority PDUs are delivered to the application in some pre-defined time by partitioning the receipt sequence of PDUs into *runs*, where each run is priority-based ordered.

1. はじめに

各計算機システム内の応用エンティティは，通信システムが提供する通信サービスを使用し，互いに情報の交換を行う．グループウェアシステム[4),15)]等の，複数の応用エンティティが協調動作を行う分散型応用システムでは，二つのエンティティ間の1対1通信[3)]に加えて，複数の宛先にデータを転送するための，高信頼放送通信サービスが必要となる．文献1)，2)，5)，6)，9)，10)，14)，16)～18)，20)，22)～25)では，複数のエンティティに対して，プロトコルデータ単位 (PDU) を紛失なく，重複することなく，ある順序で転送する方式が述べられている．

応用によっては，テキスト，音声，動画等の複数種類のデータ転送を，混在して行う必要がある．あるいは，実時間，システム制御，利用者等の，優先度の異なるデータ転送要求が，応用システム内で同時に発生する可能性がある．通信システムは，このような要求に対して，優先度の高いものを低いものより先に宛先に届ける必要がある．本論文では，分散型応用システムを構成する応用エンティティの集合を群とし，群内の全応用エンティティが，PDUを紛失することな

† 東京電機大学理工学部経営工学科
Department of Computers and Systems Engineering, Faculty of Science and Engineering, Tokyo Denki University

く，優先度に基づく同一の順序で受信する優先放送通信サービスを定義し，このためのプロトコルを提案する．優先データ転送サービスを実現する方法として，システム内で最も優先度の高いデータを持つエンティティに次の送信権を与える方法がある．この方法は，一般に，伝送媒体を共有する通信網の，媒体アクセスレベルのプロトコルで用いられている[11),13),21)]．

本論文では，別の方法として，PDU を受信した後，受信バッファ内の PDU を優先度に基づいた順序に並び換える方法を用いる．前者の方法では，送信した PDU と，それ以降に送信する PDU 間で優先度の比較が行われない．後者の方法では，ある PDU の送信後に，それ以上に優先度の高い PDU を送信した場合，これらの順序を入れ換えることができる．また，送信権を制御するために，特別な PDU を送受信する必要がない．

優先データ転送では，低優先度の PDU は，高優先度の PDU がすべて応用に渡されるまで，配送待ちの状態となる．本論文では，低優先度の PDU が無限に配送待ちとなる問題を解決する方法として，一定時間内に応用に渡されない PDU を，より高い優先度の PDU がバッファ内にあるなしに関係なく，応用に渡すことを考える．また，このときのデータ受信順序が，全応用エンティティで同一となることを保障する．下位の通信網としては，Ethernet のような単一の通信チャネルから成る網を考える．

本論文の構成は，以下のようである．2章では，通信サービスのモデルを示し，放送通信サービスの信頼性について述べる．3章では，優先度に基づいた放送通信サービスを形式的に定義し，4章でこのサービスを提供するプロトコルを示す．5章では，PDU が一定時間内に上位層に渡されない問題を解決するためのプロトコルを示す．

2. モデル

2.1 群

通信システム M は，複数の通信エンティティから構成される．群 C を，M が提供するサービスアクセス点 (SAP)[8)] の部分集合 $\{S_1, \cdots, S_n\}$ とする．群は，一対の SAP 間に定義されるコネクション[8)]を，複数の SAP 間に拡張した概念である[22),23)]．各 S_i は，通信エンティティ E_i により提供される．上位層の応用エンティティの要求により，複数の通信エンティティ $E_1, \cdots, E_n$ が $S_1, \cdots, S_n$ 間に群を確立し，各 S_i を通じて，各応用エンティティ A_i に通信サービスを提供する $(i=1, \cdots, n)$．このとき，C は $E_1, \cdots, E_n$ により提供され，各 A_i は C 内にあるとする．エンティティの障害としては，停止障害のみが起き，エンティティが障害を起こした場合，群はアボートするものとする．

2.2 放送通信サービスの信頼性

群では各 PDU が複数のエンティティに送信されるので，受信確認を分散型制御により実現するためには，各々のエンティティが，全宛先での PDU の受信を確認する必要がある．ここで，群 C は，n 個の通信エンティティ $E_1, \cdots, E_n$ により提供されるとする．各 E_i は，受信したある PDU p が，群内の全エンティティで受信されているかに関して，以下の3段階のいずれかの段階の知識を持つ[17~19),22),23)]．

(1) E_i は p を受信しているが，全エンティティで p が受信されているかを知らない．このとき，E_i は p を受理 (accept) しているという．

(2) E_i は，「C 内の全エンティティが p を受信している」ことを知っている．このとき，E_i は p を前確認 (pre-acknowledge) しているという．

(3) E_i は，「C 内の全エンティティが p を前確認している」ことを知っている．このとき，E_i は p を確認 (acknowledge) しているという．

(2)において，E_i が p を前確認していても，他のある E_j が p を前確認しているとは限らない．このような状況は，E_i は p に対する受信通知を全エンティティから受信したが，E_j はこの中の一つ以上を受信できなかった場合に起こる．(3)では，E_i は，『全エンティティが「全エンティティで p が受信されている」ことを知っている』ことがわかる．

2.3 放送通信サービス

群通信サービスを，ログ集合としてモデル化する．ログ L は，m (≥ 0) 個の PDU の系列 $<p_1 \cdots p_m]$ である．L 内で，p_i は i 番目 $(i=1, \cdots, m)$，p_1 は先頭，p_m は最後尾の PDU である．L 内の先頭の PDU p_1 を $top(L)$，最後尾 p_m を $last(L)$ により示す．L の部分系列 $<p_i\ p_{i+1} \cdots p_{j-1}\ p_j]$ を，記法 $L|_i^j$ $(1 \leq i \leq j \leq m)$ により示す．L 内の p_i と p_j について，$i<j$ ならば，p_i は p_j に先行する $(p_i \rightarrow_L p_j)$[12)] という．また，二つのログ $L_1 = <p_1 \cdots p_{m1}]$ と $L_2 = <q_1 \cdots q_{m2}]$ の連結 $<p_1 \cdots p_{m1}\ q_1 \cdots q_{m2}]$ を $L_1 \| L_2$ と書く．

各 E_i は，送信ログ (Sending Log) SL_i と受信ログ (Receipt Log) RL_i を持つ．SL_i と RL_i は，E_i

が送受信した PDU の履歴である．送受信ログ間には，以下の関係が定義される[17),18),25)]．

- E_i が，各 E_j からの PDU を，送信された順序で受信しているならば，RL_i は順序保存である．
- RL_i が，$SL_1, \cdots, SL_n$ 内のすべての PDU を含むならば，RL_i は情報保存である．
- RL_i と RL_j で，これらの両方に含まれる任意の PDU p と q の順序が同じならば，RL_i と RL_j は順序同値である．
- RL_i と RL_j が，同一の PDU を含むならば，RL_i と RL_j は情報同値である．
- RL_i が，順序保存でかつ情報保存ならば，RL_i は保存されている．RL_i と RL_j が，情報同値でかつ順序同値ならば，RL_i と RL_j は同値である．

下位の通信サービスとして，Ethernet 等の LAN の MAC 副層[7)]と，無線サービスを考える．これらのサービスを，単一チャネルサービスとしてモデル化する．

[定義] 各受信ログが，順序保存で，かつ互いに順序同値であるとき，このサービスを単一チャネル (1C) サービスという． □

1C サービスを利用した場合，各エンティティは，同一の順序で，かつ送信順に PDU を受信する．しかし，ある PDU の受信に失敗する場合がある．

PDU の紛失がないサービスとして，以下を考える．

[定義]

(1) 送信順序保存放送通信 (OP) サービスとは，各受信ログが保存されているサービスである．

(2) 全順序放送通信 (TO) サービスとは，各受信ログが互いに順序同値である OP サービスである． □

3. 優先度に基づく放送通信サービス

本章では，放送通信における優先度に基づいたデータ転送サービスを考える．各エンティティ E_i が送信する各 PDUp に，シーケンス番号 $p.SEQ$ と優先度 $p.PRI$ (>0) を付与する．ある E_i で，PDUq が p の後に送信されたならば，$p.SEQ<q.SEQ$ である．$p.PRI>q.PRI$ ならば，p は q よりも優先度が高い．優先度 0 は，システム内でのみ用いる．ここで，記法 $p_{[r]}$ は $p.RPI=r$ を示し，また，$p.SRC$ は，p の送信元エンティティ E_i を示す．

[定義] ログ L は，以下の条件を充足するとき，優先度順整列されている．ここで，p と q を L 内の任意の PDU とする．

(1) $p.PRI>q.PRI$ ならば，$p \rightarrow_L q$，かつ

(2) $p.PRI=q.PRI$ で，$p.SRC=q.SRC$ かつ $p.SEQ<q.SEQ$ ならば，$p \rightarrow_L q$ である． □

優先度順整列されたログを，優先度順ログという．

[定義] 優先度順ログ L_1 と L_2 が互いに情報同値ならば，L_1 と L_2 は優先度同値である． □

[優先度同値の例] **図 1** に，優先度同値なログの例を示す．各エンティティ E_i の受信ログ RL_i は，優先度順整列されており，かつ同一の PDU を含んでいる ($i=1,2,3$)．ただし，x と z は E_3 により送信され，同じ優先度 1 を持つので，送信順に受信されている． □

[定義] ログ L の部分系列 R が優先度順整列されているとき，R を L 内の連 (run) という ($R \sqsubseteq L$ と書く)．L 内の連 R_i ($=L|_{i_1}^{i_2}$) と R_j ($=L|_{j_1}^{j_2}$) について，$j_1=i_2+1$ ならば，R_j は R_i に連接しているという． □

[連の例] **図 2** の R_1, R_2, R_3 は，優先度順整列されているので L の連である．しかし，R_4 は，$e.PRI$ ($=1$)$<f.PRI$ ($=2$) であり，優先度順整列されていないので，L の連ではない．R_3 は R_2 に連接している． □

[定義] ログ L を，連接する連 $(R_1 \| \cdots \| R_k)$ $(k \geq 1)$ に分割したとき，この $(R_1 \| \cdots \| R_k)$ を L の連分割という． □

[定義] 以下の条件を充足するならば，ログ L の連分割 $(R_1 \| \cdots \| R_k)$ を最大という．

(1) $k=1$，または

$$RL_1 = \langle c_{[3]}\ b_{[2]}\ y_{[2]}\ p_{[2]}\ a_{[1]}\ x_{[1]}\ q_{[1]}\ z_{[1]} \rangle \qquad SL_1 = \langle a_{[1]}\ b_{[2]}\ c_{[3]} \rangle$$
$$RL_2 = \langle c_{[3]}\ y_{[2]}\ b_{[2]}\ p_{[2]}\ x_{[1]}\ q_{[1]}\ z_{[1]}\ a_{[1]} \rangle \qquad SL_2 = \langle p_{[2]}\ q_{[1]} \rangle$$
$$RL_3 = \langle c_{[3]}\ p_{[2]}\ b_{[2]}\ y_{[2]}\ q_{[1]}\ x_{[1]}\ z_{[1]}\ a_{[1]} \rangle \qquad SL_3 = \langle x_{[1]}\ y_{[2]}\ z_{[1]} \rangle$$

図 1 優先度同値
Fig. 1 Priority-equivalency.

$$L = \langle a_{[3]}\ b_{[2]}\ c_{[2]}\ d_{[1]}\ e_{[1]}\ f_{[2]}\ g_{[1]}\ h_{[1]} \rangle$$
$$R_1 = L|_1^3 = \langle a_{[3]}\ b_{[2]}\ c_{[2]} \rangle \sqsubseteq L$$
$$R_2 = L|_2^5 = \langle b_{[2]}\ c_{[2]}\ d_{[1]}\ e_{[1]} \rangle \sqsubseteq L$$
$$R_3 = L|_6^8 = \langle f_{[2]}\ g_{[1]}\ h_{[1]} \rangle \sqsubseteq L$$
$$R_4 = L|_5^8 = \langle e_{[1]}\ f_{[2]}\ g_{[1]}\ h_{[1]} \rangle \not\sqsubseteq L$$

図 2 ログ内の連
Fig. 2 Runs.

$$L_1 = < \overbrace{a_{[3]}\ b_{[2]}}^{S_{11}}\ \overbrace{c_{[2]}\ d_{[1]}\ e_{[1]}}^{S_{12}}\ \overbrace{f_{[2]}\ g_{[1]}\ h_{[1]}}^{S_{13}} >$$

$$L_2 = < \overbrace{a_{[3]}\ c_{[2]}}^{S_{21}}\ \overbrace{b_{[2]}\ e_{[1]}\ d_{[1]}}^{S_{22}}\ \overbrace{f_{[2]}\ h_{[1]}\ g_{[1]}}^{S_{23}} >$$

図 3 連 分 割
Fig. 3 Run-partition.

(2) $k \geq 2$ のとき，すべての連接する連 R_i $(=L|_{i_1}^{i_2})$ と R_j $(=L|_{j_1}^{j_2})$ $(j_1=i_2+1)$ について，$p_{i_2}.PRI < p_{j_1}.PRI$ である． □

あるログ L に対して，一つ以上の連分割が存在するが，最大連分割は一つである．

[連分割の例] **図 3** で，L_1 の連分割 $(R_{11}\|R_{12})$ は最大であるが，$(S_{11}\|S_{12}\|S_{13})$ はそうでない．同様に，$(R_{21}\|R_{22})$ は L_2 の最大であるが，$(S_{21}\|S_{22}\|S_{23})$ はそうでない． □

[定義] ログ L_i と L_j の最大連分割を，各々 $(R_{i_1}\|\cdots\|R_{i_k})$ と $(R_{j_1}\|\cdots\|R_{j_k})$ とする．ここで，以下の条件を充足するとき，L_i と L_j は連同値である．

(1) $i_k=j_k$ $(=m)$，かつ

(2) $t=1,\cdots,m$ について，R_{i_t} と R_{j_t} は優先度同値である． □

[連同値の例] 図 3 で，最大連分割 $(R_{11}\|R_{12})$ と $(R_{21}\|R_{22})$ における R_{11} と R_{21}，R_{12} と R_{22} がそれぞれ優先度同値なので，L_1 と L_2 は連同値である． □

優先度の概念を導入した放送通信サービスとして，以下の二つを定義する．

[定義]

(1) 群内の全受信ログが情報保存でかつ互いに連同値であるとき，このサービスを優先度順放送通信 (PriO) サービスという．

(2) 群内の全受信ログが互いに順序同値である PriO サービスを，全順序優先度順放送通信 (PriTO) サービスという． □

[優先度順放送通信サービスの例] PriO サービスと PriTO サービスの例を，**図 4** (1) と (2) に示す．群は，三つのエンティティ E_1, E_2, E_3 から構成され，それぞれ PDU a と b, c と d, e と f を送信し，図 4 に示す順序でこれらを受信したとする．(1) と (2) の各受信ログは，送信された全 PDU を含んでいるので情報保存である．各受信ログ RL_i は，b, c, d, e を含む連 R_{1i} と，a と f を含む連 R_{2i} に最大連分割される $(i=1,2,3)$．それぞれの連は，各エンティティ間で互いに優先度同値である．よって，(1) と

$$E_1 \quad RL_1: < \overbrace{b_{[3]}\ c_{[2]}\ e_{[2]}\ d_{[1]}}^{R_{11}}\ \overbrace{a_{[2]}\ f_{[2]}}^{R_{12}} > \qquad SL_1: < a_{[2]}\ b_{[3]} >$$

$$E_2 \quad RL_2: < \overbrace{b_{[3]}\ e_{[2]}\ c_{[2]}\ d_{[1]}}^{R_{21}}\ \overbrace{a_{[2]}\ f_{[2]}}^{R_{22}} > \qquad SL_2: < c_{[2]}\ d_{[1]} >$$

$$E_3 \quad RL_3: < \overbrace{b_{[3]}\ e_{[2]}\ c_{[2]}\ d_{[1]}}^{R_{31}}\ \overbrace{f_{[2]}\ a_{[2]}}^{R_{32}} > \qquad SL_3: < e_{[2]}\ f_{[2]} >$$

(1) 優先度順 (PriO) サービス
(1) Priority-based ordering (PriO) service

$$E_1 \quad RL_1: < \overbrace{b_{[3]}\ c_{[2]}\ e_{[2]}\ d_{[1]}}^{R_{11}}\ \overbrace{a_{[2]}\ f_{[2]}}^{R_{12}} > \qquad SL_1: < a_{[2]}\ b_{[3]} >$$

$$E_2 \quad RL_2: < \overbrace{b_{[3]}\ c_{[2]}\ e_{[2]}\ d_{[1]}}^{R_{21}}\ \overbrace{a_{[2]}\ f_{[2]}}^{R_{22}} > \qquad SL_2: < c_{[2]}\ d_{[1]} >$$

$$E_3 \quad RL_3: < \overbrace{b_{[3]}\ c_{[2]}\ e_{[2]}\ d_{[1]}}^{R_{31}}\ \overbrace{a_{[2]}\ f_{[2]}}^{R_{32}} > \qquad SL_3: < e_{[2]}\ f_{[2]} >$$

(2) 全順序優先度順 (PriTO) サービス
(2) Priority-based total ordering (PriTO) service

図 4 優先度順 (PriO) サービスと全順序優先度順 (PriTO) サービス
Fig. 4 Priority-based ordering (PriO) and priority-based total ordering (PriTO) services.

(2) で，各受信ログは互いに連同値である．(1) の PriO サービスでは，各連内の等しい優先度を持つ PDU，つまり c と e，a と f の受信順序は同じでない．しかし，(2) の PriTO サービスでは，全エンティティで PDU の受信順序が同じである． □

4. 全順序優先度順 (PriTO) プロトコル

1C サービスを用いて，PriTO サービスを提供するためのプロトコルを示す．

4.1 変 数

各エンティティ E_i は，以下の変数を持つ．

- $SEQ=E_i$ が次に送信予定の PDU のシーケンス番号．
- $REQ_j=E_i$ が，E_j から次に受信予定の PDU のシーケンス番号．
- AL_{jk}=「E_k が次に E_j から受信予定である」と，E_i が認識している PDU のシーケンス番号．
- $minAL_j=AL_{j1},\cdots,AL_{jn}$ 内の最小値．
- PAL_{jk}=「E_k が次に E_j から前確認予定である」と，E_i が認識している PDU のシーケンス番号．
- $minPAL_j=PAL_{j1},\cdots,PAL_{jn}$ 内の最小値．
- $PEQ_j=E_i$ が，次に前確認予定の，E_j からの PDU のシーケンス番号．

各 E_i が送信する各 PDUp は，送信元 $p.SRC$，シーケンス番号 $p.SEQ$，優先度 $p.PRI$ に加えて，以下の制御情報を持つ．

• $p.ACK_j = E_i$ が，E_j から次に受信予定の PDU のシーケンス番号 ($j=1,\cdots,n$).

4.2 送 受 信

各 E_i での，PDU の送受信手続きを示す．E_i は，応用エンティティからデータ送信要求を受けたとき，PDUp を組み立て，以下の送信手続きにより p を放送する．ここで，$enqueue(L,p)$ は，ログ L の最後尾に p を追加する演算である．$broadcast(p)$ は，1C サービスにより p を放送する．

[送信手続き] $p.SEQ := SEQ$; $SEQ := SEQ+1$;
$p.ACK_j := REQ_j$ ($j=1,\cdots,n$); $enqueue(SL_i, p)$;
$broadcast(p)$; □

PDU を優先度順に受信ログに格納するために，以下の優先度順挿入演算$\triangleleft$を定義する．

[優先度順挿入] PDUp の優先度順ログ $L=<p_1 \cdots p_m]$ への優先度順挿入 $L\triangleleft p$ を，優先度順ログ $<p_1 \cdots p_{i-1}\ p\ p_i \cdots p_m]$ (ただし，$p.PRI > p_i.PRI$) とする．□

例えば，$<a_{[4]}\ b_{[3]}\ c_{[1]}]\triangleleft d_{[2]} = <a_{[4]}\ b_{[3]}\ d_{[2]}\ c_{[1]}]$，$<a_{[4]}\ b_{[3]}\ c_{[1]}]\triangleleft e_{[3]} = <a_{[4]}\ b_{[3]}\ e_{[3]}\ c_{[1]}]$ である．

E_i は，PDUp ($p.SRC=E_j$) を受信したとき，以下の手続きにより p を受理する．ここで，p と同じ制御情報を持つがデータを含まない PDU を，p の擬 PDU (p^* と書く) とする．擬 PDU には優先度 0 が付与される．各 E_i は，受理した PDU を格納するためのログ PRL_i (Receipt Log for Pre-acknowledged PDUs) と RRL_i (Receipt Log for Accepted PDUs) を持つ．PRL_i と RRL_i は，RL_i の連接する部分系列である．

[受理手続き]
if($p.SEQ = REQ_j$){
  $RRL_i \triangleleft p^*_{[0]}$; $(PRL_i \| RRL_i) \triangleleft p$;
  $REQ_j := p.SEQ+1$;
  $AL_{hj} := p.ACK_h$ ($h=1,\cdots,n$);
} □

各 PDUp は優先度順に受信ログに格納される．擬 PDU は，優先度が 0 であることから，優先度順挿入の結果，RRL_i の最後尾に追加される．つまり，擬 PDU の系列は，各 PDU の受信順序を示す．

各 E_i は，受理手続きでの AL への代入演算により，ある $minAL_j$ が更新されたならば，以下の前確認手続きを実行する．$dequeue(L)$ は，L の先頭の PDU を取り出す演算である．

[前確認手続き]
while (($p=top(RRL_i)$ が擬 PDU でない) **or**
  ((p が擬 PDU) **and**
   ($p.SEQ < minAL_j$, ただし $p.SRC=E_j$))){
  $p := dequeue(RRL_i)$; $enqueue(PRL_i, p)$;
  if(p が擬 PDU){
    $PEQ_j := p.SEQ+1$;
    $PAL_{hj} := p.ACK_h$ ($h=1,\cdots,n$);
  }
} □

PDUp ($p.SRC=E_j$) について，$p.SEQ < AL_{jk}$ ならば，E_i は，E_k から p に対する受信通知を受信している．このとき，E_i は，「E_k が p を受理している」ことを知っている．つまり，条件 $p.SEQ < minAL_j$ が成立するならば，すべての E_k について $p.SEQ < AL_{jk}$ であり，p に対する受信通知を全エンティティから受信している．よって，p が全エンティティで受理されていることがわかる[22),23)]．ログ PRL_i には，擬 PDU の中で前確認されているものだけが格納される．

各 E_i は，応用エンティティ A_i に PDU を優先度順に渡すためのログ ARL_i (Receipt Log for Acknowledged PDUs) を持つ．E_i は，以下の確認手続きにより，PDU を ARL_i に移動する．ここで，$RL_i = ARL_i \| PRL_i \| RRL_i$ である．A_i は，ARL_i 内の PDU を格納されている順に取り出すことにより，PDU を優先度順に受信する．ここで，$delete(L,p)$ はログ L 内の PDU p を消去する演算である．

[確認手続き] $NotEnd$:=TRUE;
while($NotEnd$){
  if($p=top(PRL_i)$ が擬 PDU でない){
    if($p.SEQ < minPAL_j$, ただし $p.SRC-E_j$){
      $p := dequeue(PRL_i)$; $enqueue(ARL_i, p)$;
      $delete(PRL_i, p^*)$;
    }**else** $NotEnd$:=FALSE;
  }
} □

$p.SEQ < PAL_{jk}$ ならば，E_i は，条件 $p.SEQ < q.ACK_j$ を満たす E_k から受信した PDU q を前確認している．このとき，E_i は，「q が全エンティティで受理されている」ことを知っている．つまり，確認手続き内の条件 $p.SEQ < minPAL_j$ が成立するならば，すべての E_k について $p.SEQ < PAL_{jk}$ であり，『各エンティティは，「p が全エンティティで受理されている」ことを知っている』ことがわかる[22),23)]．よって，p は

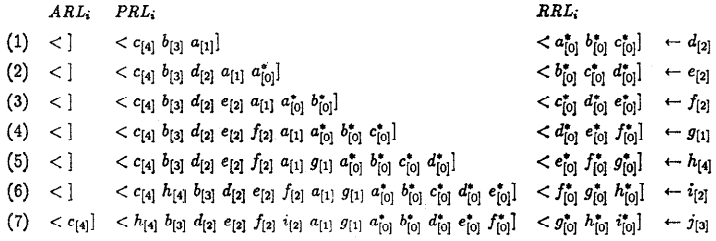


図 5 データ転送
Fig. 5 Data transmission.

確認される．ログ ARL_i には，確認された PDU だけが格納される．

[PDU の送受信の例] 図 5 に，PriTO プロトコルのデータ転送の例を示す．

(1) E_i は，PDU a, b, c を受理し，図に示す順序で PRL_i に優先度順挿入される．擬 PDU a^*, b^*, c^* は，受信順に RRL_i に格納される．各擬 PDU の優先度は 0 である．この後，$d_{[2]}$ を受信する．

(2) $d_{[2]}$ と $d^*_{[0]}$ が，それぞれ PRL_i と RRL_i に優先度順挿入される．このとき，a が前確認されたとする．a^* は，RRL_i から PRL_i に移動される．$e_{[2]}$ を受信したとする．

(3) $e_{[2]}$ が PRL_i に，$e^*_{[0]}$ が RRL_i に優先度順挿入される．ここで，b が前確認されたとする．この後，$f_{[2]}$ を受信したとする．

(4) $f_{[2]}$ は，PRL_i に優先度順挿入され，c が前確認されたとする．この後，$g_{[1]}$ を受信する．

(5) $g_{[1]}$ が PRL_i に優先度順挿入され，d が前確認されたとする．a は，d を受理したときに前確認されたので，a が確認される．しかし，a よりも優先度が高い PDU が PRL_i 内に存在するので，a は ARL_i に移動されない．$h_{[4]}$ を受信したとする．

(6) $h_{[4]}$ は，PRL_i 内の c と b の間に挿入される．e が前確認されたとする．e の前確認により b が確認されるが，c と h が確認されていないので，ARL_i に移動されない．ここで，$i_{[2]}$ を受信したとする．

(7) $i_{[2]}$ は，PRL_i に優先度順挿入される．f が前確認されたとする．f を受理したときに，c が前確認されたので，c が確認される．c は PRL_i の先頭なので，PRL_i から ARL_i へ移動される． □

4.3 障害回復

下位層が提供する 1 C サービスでは，PDU の紛失が起きる可能性がある．各 E_i は，PDU の紛失を，シーケンス番号を調べることにより検出する[25]．まず，全エンティティ間で，どの PDU が紛失したかについて合意がとられ，紛失 PDU 以降に受信された全 PDU が全エンティティで廃棄される．その後，廃棄された PDU を再送する．いわゆる，*go-back-n* 再送方法[26]を用いる．

5. 連同期プロトコル

これまでに示した手続きでは，各 PDU が一定時間内に上位層に渡されることを保証していない．低優先度の PDU は，より高い優先度の PDU が確認されるまで，受信ログ内で待ち続ける．E_i で，確認された後，一定時間内に ARL_i に移動されない（上位層に渡されない）PDU を，強制的に ARL_i に移動することにより，この問題を解決する．

各 PDUp が確認されたら，p に対してタイマが始動される．p が ARL_i に移動されたら，このタイマを停止する．以下の連同期手続きにより，タイムアウトした PDU を全エンティティに通知し，各 E_i で連を分割する．このとき，PriTO サービスの性質から，各 E_i で同一の連を作る必要がある．連同期手続きを使用するか否かは応用が決定するものとし，タイムアウト時間も群確立時に応用が与えるものとする．各 E_i は変数 $TOSEQ_h$ を持ち，連同期手続きを開始する前は，$TOSEQ_h = NIL$ $(h=1, \cdots, n)$ とする．

[連同期手続き]

(1) E_i で，PDUp $(p.SRC = E_h)$ がタイムアウトしたとする．E_i は，$TOSEQ_h := p.SEQ$ とし，前確認および確認手続きを停止する．PDU の送受信は停止しない．$s.TOSEQ_j = TOSEQ_j$, $s.PEQ_j = PEQ_j$ $(j=1, \cdots, n)$ である Run-Sync PDU s を放送し，タイムアウトした PDU と前確認されている PDU を，全エンティティに通知する．

(2) E_j は，E_i から Run-Sync PDU s を受信したならば，確認手続きと前確認手続きを停止する．$TOSEQ_h = NIL$ または $TOSEQ_h < s.TOSEQ_h$ ならば，$TOSEQ_h := s.TOSEQ_h$ とす

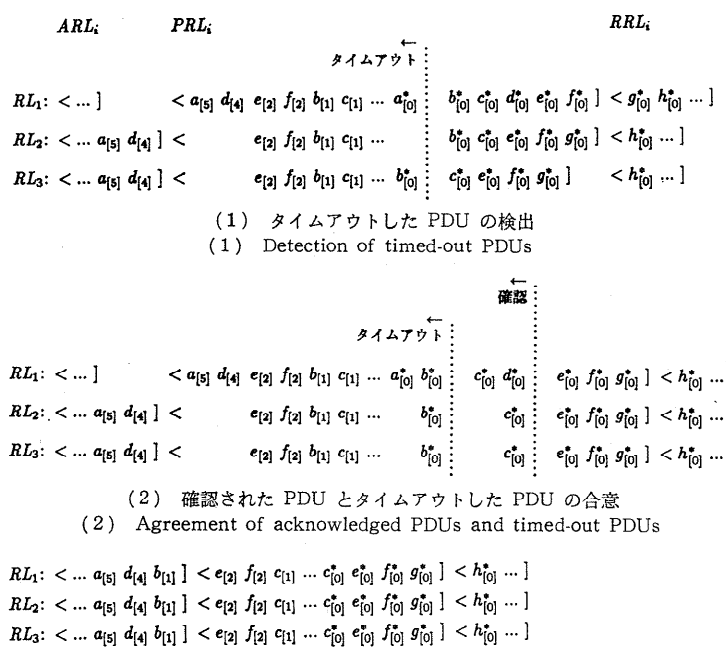
る ($h=1,\cdots,n$). E_j 内で，ある PDUq ($q.SRC=E_k$) がタイムアウトしており，$TOSEQ_k<q.SEQ$ならば，$TOSEQ_k:=q.SEQ$ とする ($k=1,\cdots,n$). つまり，E_k から受信した PDU の中で，E_i がタイムアウトを検出した PDU よりも後から受信した PDU がタイムアウトしているならば，$TOSEQ_k$ の値を更新する．E_j は，$sp.TOSEQ_h=TOSEQ_h$, $sp.PEQ_h=PEQ_h$ ($h=1,\cdots,n$) である Run-Sync-PACK PDU sp を放送する．

(3)各 E_j は，全エンティティから Run-Sync または Run-Sync-PACK PDU を受信したならば，$sa.TOSEQ_h=TOSEQ_h$, $sa.PEQ_h=PEQ_h$ ($h=1,\cdots,n$) である Run-Sync-ACK PDU sa を放送する．

(4)各 E_j が，全エンティティから Run-Sync-ACK を受信したとする．このとき全エンティティは，同一の $TOSEQ_h$ と PEQ_h ($h=1,\cdots,n$) を持つ．各 E_j は，前確認動作を行い，前確認されている擬 PDU を RRL_j から PRL_j に移動する．次に確認手続きを行い，確認されている PDU を PRL_j から ARL_j に移動し，その擬 PDU を削除する．次に，$p.SEQ\leq TOSEQ_h$ である p($p.SRC=E_h$) を，ARL_j に移動する ($h=1,\cdots,n$). 連同期手続きを終了し，通常の確認および前確認手続きを再開する．

□

(1)～(3)が終了した時点で，各 E_i は，群内の各エンティティでタイムアウトしている PDU と，前確認されている PDU がわかる．各 E_i は，PDU の受信を停止しないが，前確認と確認の処理を停止する．このため，新たに PDU を受信しても，ARL_i と PRL_i の内容は変化しない．(4)で，タイムアウトしている PDU と，前確認されている PDU について合意がとられた後，確認されていてかつタイムアウトしている PDU が，優先度順に ARL_i に移動される．各 E_i は，Run-Sync, Run-Sync-PACK, Run-Sync-ACK を送信するとき，放送するデータがあれば，これらの PDU に含める．



(1) タイムアウトした PDU の検出
(1) Detection of timed-out PDUs

(2) 確認された PDU とタイムアウトした PDU の合意
(2) Agreement of acknowledged PDUs and timed-out PDUs

(3) 連同期手続きの終了
(3) Termination of the run synchronization procedure

図 6 連同期の例
Fig. 6 An example of the run synchronization.

[連同期の例] 連同期の例を，**図 6** に示す．群は，三つのエンティティ E_1, E_2, E_3 から構成され，各送受信ログは，図6(1)に示す PDU を含んでいるとする．

(1)E_1 は，a が確認されてから，一定時間経過しても a が上位層に渡されないので，Run-Sync PDU を放送し，連同期手続きを起動する．E_2 と E_3 では，a と d が既に ARL に移動されている．Run-Sync PDU を受信した E_3 が，b のタイムアウトを検出したとする．各エンティティは，Run-Sync-PACK PDU を放送する．

(2)各エンティティは，全エンティティから Run-Sync または Run-Sync-PACK PDU を受信することにより，E_1 と E_3 でそれぞれ a と b がタイムアウトしていることを知る．また，E_1 では f までが，E_2 と E_3 では g までが前確認されていることを知る．各エンティティは前確認手続きを行い，g までが前確認される．次に確認手続きを行うことにより，a, b, c, d が確

認される.

(3)各エンティティは，a, d, b を優先度順に ARL に移動する．c も確認されているが，e と f が確認されておらず，タイムアウトしていないので，ARL に移動されない. □

連同期手続きが起動されたとき，各エンティティは，まず Run-Sync または Run-Sync-PACK を送信する．次に，これらを各エンティティから受信したとき，Run-Sync-ACK を送信する．群内のエンティティ数を n としたとき，全体として $2n$ 個の PDU が送信される．しかし，これらの PDU にデータを含められるので，連同期のためだけに送信される PDU 数は，$2n$ 以下である．また，優先度のレベル数と PDU の遅延時間の関係は，文献19)で述べている．

PDU の紛失がない場合，PriTO プロトコルが PriTO サービスを提供するのは明らかである．PDU が紛失した場合，その検出と再送が行われる．また，連同期手続きが開始されるときにある PDU が紛失していたとしても，あるエンティティで前確認されている PDU は，全エンティティで前確認されるので，全エンティティで同一の PDU が確認される．また，各エンティティでタイムアウトした PDU の合意がとられる．従って，各 E_i は，同一の PDU を，同一の順序で優先度順に ARL_i に移動する．

本方式は，連同期の起動をタイムアウトにより起動する．これ以外の起動方法として，確認後も ARL_i に移動されない PDU が一定数以上に達した時点が考えられる．本論文では，連同期による遅延時間の保障よりも，優先度を受信順序に反映させることを重要と考え，PDU 数ではなくタイムアウトにより手続きを起動することとした．また，利用者の要求によって起動することが考えられるが，これは将来の検討課題とする．

6. まとめ

本論文では，放送通信における優先サービスのモデルと，優先サービスを提供するためのプロトコルの設計について述べた．優先度の低い PDU が一定時間以上，最悪の場合には無限に待ち続けてしまう問題を解決するために連の概念を導入し，一定時間内に応用エンティティに渡す方法を示した．各連内の PDU は，優先度順に整列されている．応用エンティティは，同一の連の系列により，PDU を同一の順序で受信する．本プロトコルにより，一つの群内で種々のデータが放送される応用に対して，優先度順に PDU を転送するサービスが提供される．

参考文献

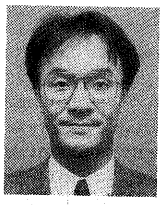
1) Birman, K., Schiper, A. and Stephenson, P.: Lightweight Causal and Atomic Group Multicast, *ACM Trans. Comput. Syst.*, Vol. 9, No. 3, pp. 272-314 (1991).
2) Chang, J. M. and Maxemchuk, N. F.: Reliable Broadcast Protocols, *ACM Trans. Comput. Syst.*, Vol. 2, No. 3, pp. 251-273 (1984).
3) Defense Communications Agency: *DDN Protocol Handbook*, Vol. 1-3, NIC 50004-50005 (1985).
4) Ellis, C. A., Gibbs, S. J. and Rein, G. L.: Groupware: Some Issues and Experiences, *Comm. ACM*, Vol. 34, No. 1, pp. 38-58 (1991).
5) Garcia-Molina, H. and Kogan, B.: An Implementation of Reliable Broadcast Using an Unreliable Multicast Facility, *Proc. of the 7th IEEE Symp. on Reliable Distributed Systems (ICDCS)*, pp. 428-437 (1988).
6) Garcia-Molina, H. and Spauster, A.: Message Ordering in a Multicast Environment, *Proc. of the 9th IEEE ICDCS*, pp. 354-361 (1989).
7) IEEE: IEEE Standard 802.3 Carrier Sense Multiple Access with Collision Detection (1988).
8) ISO: Open Systems Interconnection—Basic Reference Model, ISO 7498 (1987).
9) 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, pp. 5-19 (1989).
10) Kaashoek, M. F. and Tanenbaum, A. S.: Group Communication in the Amoeba Distributed Operating System, *Proc. of the 11th IEEE ICDCS*, pp. 222-230 (1991).
11) Kurose, J. S., Schwartz, M. and Yemini, Y.: Multiple-Access Protocols and Time-Constrained Communication, *ACM Comput. Surv.*, Vol. 16, No. 1, pp. 43-70 (1984).
12) Lamport, R.: Time, Clocks, and the Ordering of Events in Distributed Systems, *Comm. ACM*, Vol. 21, No. 7, pp. 558-565 (1978).
13) Liu, M. and Papantoni-Kazakos, P.: A Random Access Algorithm for Data Networks Carrying High Priority Traffic, *Proc. of the 9th IEEE INFOCOM*, pp. 1087-1094 (1990).
14) Luan, S. W. and Gligor, V. D.: A Fault-Tolerant Protocol for Atomic Broadcast, *IEEE Trans. Parallel and Distributed Systems*, Vol. 1, No. 3, pp. 271-285 (1990).
15) 松下 温（編著）：図解 グループウェア入門，

オーム社 (1991).
16) Melliar-Smith, P. M., Moser, L. E. and Agrawala, V.: Broadcast Protocols for Distributed Systems, *IEEE Trans. Parallel and Distributed Systems*, Vol. 1, No. 1, pp. 17-25 (1990).
17) Nakamura, A. and Takizawa, M.: Reliable Broadcast Protocol for Selectively Ordering PDUs, *Proc. of the 11th IEEE ICDCS*, pp. 239-246 (1991).
18) Nakamura, A. and Takizawa, M.: Design of Reliable Broadcast Communication Protocol for Selectively Partially Ordered PDUs, *Proc. of the IEEE COMPSAC '91*, pp. 673-679 (1991).
19) Nakamura, A. and Takizawa, M.: Priority-Based Total and Semi-Total Ordering Broadcast Protocols, *Proc. of the 12th IEEE ICDCS*, pp. 178-185 (1992).
20) Schneider, F. B., Gries, D. and Schlichting, R. D.: Fault-Tolerant Broadcasts, *Science of Computer Programming*, Vol. 4, No. 1, pp. 1-15 (1984).
21) Sharrock, S. M. and Du, D. H. C.: Efficient CSMA/CD-Based Protocols for Multiple Priority Classes, *IEEE Trans. Comput.*, Vol. 38, No. 7, pp. 943-954 (1989).
22) Takizawa, M.: Cluster Control Protocol for Highly Reliable Broadcast Communication, *Proc. of the IFIP Conf. on Distributed Processing*, pp. 431-445 (1987).
23) Takizawa, M.: Design of Highly Reliable Broadcast Communication Protocol, *Proc. of IEEE COMPSAC '87*, pp. 731-740 (1987).
24) Takizawa, M. and Nakamura, A.: Partially Ordering Broadcast (PO) Protocol, *Proc. of the 9th IEEE INFOCOM*, pp. 357-364 (1990).
25) Takizawa, M. and Nakamura, A.: Reliable Broadcast Communication, *Proc. of IPSJ Int'l. Conf. on Information Technology (Info-Japan)*, pp. 325-332 (1990).
26) Tanenbaum, A. S.: *Computer Networks* (2nd edition), Prentice-Hall, Englewood Cliffs, N.J. (1989).

(平成4年10月23日受付)
(平成5年 4 月 8 日採録)

中村 章人 (正会員)

1966年生. 1989年東京電機大学理工学部経営工学科卒業. 1991年同大学院工学研究科修士課程修了. 現在, 同大学院理工学研究科博士後期課程在学中, 分散型システム, 通信プロトコル等に興味をもつ.



滝沢 誠 (正会員)

1950年12月6日生. 1973年東北大学工学部応用物理学科卒業. 1975年東北大学大学院工学研究科応用物理学専攻修士課程修了. 同年(財)日本情報処理開発協会入社. 1986年東京電機大学理工学部経営工学科講師, 1987年より同助教授. 工学博士. 1989年9月より1年間ドイツ国立情報処理研究所 (GMD) 客員教授. 1990年7月よりKeele大学 (英国) 客員教授. 分散型データベースシステム, 通信網, 分散型システム, 知識ベースシステム等の研究に従事. 電子情報通信学会, 人工知能学会, ACM, IEEE 各会員. 「知識工学基礎論」オーム社 (共著), 「データベースシステム入門技術解説」ソフトリサーチセンタ, 「分散システム入門」近代科学社 (共著).