Ethernet 上で QoS を保証する通信方法の設計と実装

中野隆裕[†] 岩嵜正明[†]
中原雅彦[†] 竹内理[†]

現在，LAN に最も普及している Ethernet は帯域幅だけを考慮すれば，MPEG 形式などに圧縮された連続メディアデータを転送可能である．しかし，高トラフィック下においては，送信衝突や再送によるパケットの送信遅延や欠落が発生するため，通信品質の保証が困難である．我々は，アイソクロナス・スケジューリング機構を利用し，Ethernet 上で QoS を保証する通信方法を開発した．本方式は，セグメント上の総送信要求量を制御する制御プロトコルと，各ノードの送信データ量を抑制するモジュールからなる．本方式を用いたビデオデータ転送実験では，高負荷下においても，ビデオ再生の品質が保たれ，QoS 保証が可能であることを確認した．

Design and Implementation of a QoS Assurance Mechanism on Ethernet

Takahiro Nakano,[†] Masaaki Iwasaki,[†] Masahiko Nakahara[†]
and Tadashi Takeuchi[†]

Ethernet is the most popular network for Local Area Network, and it has enough bandwidth to transmit compressed continuous media data like a MPEG format. However, it is difficult to assure a quality of transmission under heavy load because its collision detection and re-transmission mechanism causes delay and packet loses. In this paper, we propose a QoS assured transmission mechanism on Ethernet using isochronous scheduling. The mechanism consists of two parts, a control protocol to limit the total traffic in a segment, and a transfer module that shape its traffic into isochronous flow on each node. Furthermore, we implement the mechanism on PCs, and experiment with video data transmission over Ethernet. The experimental results show that the mechanism enables QoS assured transmission under heavy load.

1. はじめに

近年，ネットワークのハードウェア技術が目覚ましく進歩し，ATM ネットワークやギガビット Ethernet をはじめとする高速ネットワークが数多く出現してきた．これらのネットワークは MPEG 形式などに圧縮された連続メディアデータを転送するのに十分なハードウェア性能を備える．これにともない，インターネットを介して，高品質な連続メディアデータの送受信を実現したい，という要求が高まりつつある．

上記要求を満たすためには，その通信経路上にあるすべてのネットワークにおいて通信の QoS を保証しなければならない．しかし，現状では，その経路の一部に Ethernet など，通信の QoS 保証が困難とされるネットワークが含まれるケースが多い．

Ethernet は，安価，高速，敷設・管理が容易なため，現在，LAN に最も普及しているネットワークのハードウェアであり，連続メディアデータを転送可能な帯域幅を備えている．しかし，Ethernet は，そのハードウェアの性質上，他ノードが生じるトラフィックに影響され，連続メディアデータ転送の QoS を保証できなくなることが知られている．

特に，伝送メディアを共有する Shared Ethernet は，スイッチング HUB を用いた Switched Ethernet と比べ，伝送メディアのアクセス競合が頻繁に発生するため，通信品質を保証することは困難となる．

本研究では，通信の QoS を保証することが困難とされている Shared Ethernet のハードウェアを変更することなく，固定ビットレート・データ転送における QoS 保証を可能にする通信方法の実現を目的としている．

本論文では，2 章にて，Shared Ethernet のハードウェア特性について検討する．3 章にて，Ethernet に

† 株式会社日立製作所システム開発研究所
Systems Development Laboratory, Hitachi Ltd.

おいて QoS を保証する通信方式を提案する．4 章にて，3 章において提案した方式の実装について述べ，5 章にて実験を行い，その結果を評価する．

2. Ethernet のハードウェア特性の検討

本章では，Shared Ethernet の動作概要，および，特性について記述する．

2.1 Shared Ethernet の動作概要，特性

Shared Ethernet は，伝送メディアを複数のノードにて共有し，その伝送メディアを共有するノード間の通信を可能にする．複数のノードが，同時に伝送メディアにデータを送信すると，伝送メディア上にてデータが干渉し合い，通信が正しく行えなくなる．Shared Ethernet では，CSMA/CD☆1方式と呼ばれる MAC☆2技術を用い，この送信衝突を回避している．

以下に，Ethernet における CSMA/CD によるフレーム送信の手順を示す．

(1) 他ノードが送信中であった場合，送信が終わるまで待つ．
(2) 一定時間（フレーム間ギャップに相当），送信が行われないことを確認し，送信を開始する．
(3) 送信中は，伝送メディア上のデータと送信したデータの一致を監視する．これにより，他ノードと同時に送信を開始した場合に起きる送信衝突を検出することができる．
(4) 送信衝突が発生しなければ，送信は完了する．
(5) 送信衝突が発生した場合，送信を中止し，ジャム信号を送る．これにより，受信中であったノードにパケットを破棄させる．
(6) 待ち時間（backoff）の後，(1) に戻り再送を試みる．再送は，15 回まで試行されるが，15 回連続して送信衝突が発生した場合，そのフレームの送信を中止する．

上記 (6) の待ち時間は，Slot Time☆3を再送回数の冪乗を上限とするランダム倍した時間に決定される．

CSMA/CD 方式は，このような再送の backoff アルゴリズムにより，送信要求が集中した場合，自動的に各ノードの送信間隔を広げ，各ノードからの送信トラフィックを低減し，ジャム状態を回避する機能を提供している．

しかし，この backoff アルゴリズムによるジャム状態回避機能が，通信の QoS 保証（2.2 節）を困難にしている．たとえば，連続メディアデータを一定レートにて送信しようと試みても，同一伝送メディア上にある他のノードが FTP などの大量データ転送を開始すると，トラフィックが増大し，送信衝突が頻繁に発生する．送信衝突の発生が連続すると，ジャム状態回避機能が働き，送信間隔が広げられ，送信レートを維持できなくなる．さらに，15 回の再送失敗が起きると，TCP など上位プロトコルによる再送が発生し，さらに遅延時間が増大する．

2.2 通信の QoS 保証

VOD☆4システム，および，ビデオ会議システムなどにおいて，高品質なサービスの提供，たとえば，連続メディアデータ再生品質の維持を実現するには，アプリケーション，OS，ネットワークを含めた end-to-end の QoS 保証が必要となる．

一般に，連続メディアデータの転送における，通信の QoS として，帯域幅保証，遅延時間保証，遅延分散保証，紛失率保証が必要とされる[1]．

- 帯域幅保証：指定された最大送信レートを保証する
- 遅延分散保証：遅延時間のゆらぎの幅を保証する
- 遅延時間保証：データの送信要求から受信までの遅延時間を保証する
- 紛失率保証：パケット欠落の発生確率を保証する

アプリケーションにおいては，連続メディアデータの再生に必要なデータが，必要な時刻に転送されることが求められる．上記システムにおいては，この再生アプリケーションが必要とするデータを，継続的に遅延なく供給する必要があり，帯域幅保証が必要となる．

また，データ到着遅延のゆらぎをバッファリングにより吸収するには，バッファ容量を正確に予測するために，遅延分散保証が必要となる．

さらに，ビデオ会議システムは，自然な対話が行える必要があり，遅延時間の保証が必要となる．一般に，人間が支障なく会話できる遅延は，およそ 150 ミリ秒程度までといわれている[2]．

これまでの Shared Ethernet においては，送信衝突にともなう backoff の待ち時間の上限が，再送回数とともに急激に増大することや，送信失敗にともない上位プロトコルが行う再送の遅延などにより，これら通信の QoS を保証することは困難であった．

3. TTCP/ITM 方式

本章では，Shared Ethernet 上にて通信の QoS を保証する通信方法である TTCP/ITM（Total Traffic Control Protocol with Isochronous Transfer Mode）

☆1 Carrier Sense Multiple Access with Collision Detection
☆2 Media Access Control
☆3 512 bit のデータを送信する時間に相当．
☆4 Video on Demand

方式を提案する．本方式は，Ethernet のハードウェア，および，TCP などの上位通信プロトコル，既存の通信アプリケーションを変更することなく，これらと共存可能な QoS 保証通信を実現する．

まず，TTCP/ITM 方式の基本原理である周期送信と総帯域管理による遅延時間低減について記述し，次に，TTCP/ITM 方式の設計について記述する．

3.1 周期送信と総帯域管理による遅延時間低減

総帯域管理は，セグメント内の各ノードが送信するデータ量を割り当てられた帯域幅に制限し，セグメント内の総トラフィックを抑制する帯域管理方法である．具体的には，QoS を保証する通信〔以下，RT（Real Time）通信〕の帯域割当てとともに，QoS 保証を必要としない通信〔以下，NRT（Non Real Time）通信〕に対しても帯域を割り当て，RT 通信，および，NRT 通信に割り当てる帯域幅の総量を一定値以下に抑える．

2.1 節に示すとおり，Shared Ethernet は，フレーム送信において，RT 通信用のフレームを優先する機能を持たない．このため，RT 通信，NRT 通信を問わず，フレームの送信衝突が発生する．送信衝突が連続的に発生すると，送信遅延が増加する問題があり，総帯域管理のみでは QoS 保証は困難である．TTCP/ITM では，この解決のため，以下に示す周期送信を用いて，TrafficShaping を行う．

周期送信は，送信しようとする複数のパケットをいったんキューに蓄え，各周期ごとに一定量のパケットずつ送信する送信方法である．**図 1** に，周期送信の送信イメージを示す．他の送信ノードが存在しない場合，この送信方法を用いることで，ノードが 1 周期ごとに送信するデータ量を一定量に抑制できる．

TTCP/ITM において，周期送信の送信契機を決める周期は，一定の間隔を各ノードがインターバルタイマなどを用いて独自に生成する．ノード間の同期をとらないため，各ノードの送信契機には，「周期のずれ」が生じる．この「周期のずれ」により，送信契機が分散し，送信衝突の発生確率が低下するが，逆に「周期のずれ」が小さいノード間においては，送信衝突が発生する確率が高まる．ただし，送信衝突が発生し，MAC 層での再送が行われ，送信に遅延が発生しても，その送信が次の周期までに完了すれば，帯域幅を満足可能である（**図 2**）．総帯域管理を用いて，周期あたりに各ノードから送信される総トラフィックを一定値以下に抑えることにより，伝送メディアにデータが送信されない時間を各周期ごとに設け，再送の成功確率を高めることができる．さらに，この周期に用いる時間間隔を Ethernet の再送アルゴリズムに基づき，次の周期までに送信が十分高い確率にて完了する値に設定することにより，通信の QoS を保証可能となる．

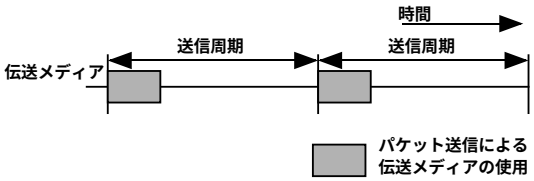


図 1 周期送信
Fig. 1 Isochronous transmission.

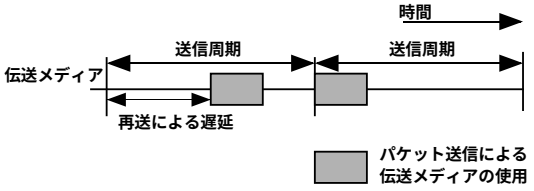


図 2 許容される送信遅延
Fig. 2 Permissible delay to keep bandwidth.

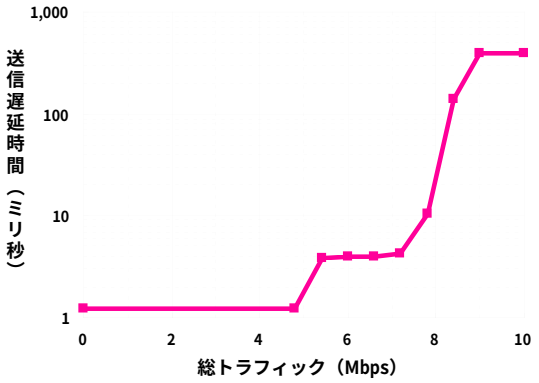


図 3 Ethernet の特性
Fig. 3 Ethernet characteristics.

図 3 に，10 Mbps Ethernet を用い，40 ミリ秒☆を周期とする周期送信を行った場合の総トラフィックと，送信遅延時間の関係を，計算機シミュレーションによって定量評価した結果を示す．総トラフィックは，セグメント内のノードに割り当てた帯域幅の総和を示す．送信遅延時間は，セグメント内のすべてのノードが送信を要求したパケットのうち，遅延時間が小さいほうから 99%の分布範囲における最大値を示している．

図 3 から，総トラフィックを 6〜7.5 Mbps（物理帯域幅の 60〜75%）に抑制することにより，10 ミリ秒を超える遅延が 99%の確率で発生しないことが分かる．周期送信に用いた周期 40 ミリ秒に比べ，遅延が

☆ およそ再送 8 回の最長待ち時間に相当．

小さいことから，総トラフィックを上記の値に抑制することにより，99%以上の確率にて，遅延分散 40 ミリ秒以下を保証する通信が可能になるといえる．

3.2 TTCP/ITM の設計概要

ここでは，3.1 節にて示した原理に基づく TTCP/ITM 方式の設計について記述する．

TTCP/ITM 方式は，Ehternet のハードウェア仕様を変更することなく，Ethernet 上で QoS 保証を施した通信を利用可能にする．

帯域管理サーバ Ethernet セグメント内の帯域割当て情報を集中管理（総帯域管理）するサーバ．各ノードからの要求メッセージに応じ，セグメント内の帯域割当てや帯域解放を行う．

TTCP 帯域管理サーバに帯域割当てや帯域解放を要求する帯域管理制御プロトコル．固有の Ethernet フレームタイプを割り当て，特定の上位通信プロトコルに依存しない☆制御プロトコルとし，複数の上位通信プロトコルが混在するネットワークにおいても，総帯域管理を実現可能としている．

ITM 周期送信を用い単位時間あたりの送信データ量を抑制する Traffic Saping 技術．Ethernet セグメント内のすべてのノードにデバイスドライバの一部として組み込み，1 周期ごとの送信量を帯域管理サーバが指定したトラフィック量に抑制する．

3.2.1 帯域管理サーバ

帯域管理サーバは，Ethernet セグメントの 1 ノードに設け，セグメント内の総帯域管理を行うモジュールである．セグメント内の総送信要求量が一定値以下になるよう各ノードの要求に応じ帯域を割り当てる．

3.2.1.1 帯域割当て方針

RT 通信と，NRT 通信では，帯域の指定方法や，帯域割当て・解放の要求タイミングが異なる．TTCP/ITM 方式は，この違いを考慮し，RT 通信，NRT 通信に割り当てる帯域を，以下に示す方針に従って扱う．

3.2.1.1.1 RT 通信

RT 通信の帯域割当て方針は，アプリケーションが指定した帯域をポートごとに必要な時間利用可能にすることである．

アプリケーションは，RT 通信に先立ち，必要な帯域幅を指定し，TTCP に帯域割当てを要求する．また，RT 通信が完了すると，TTCP に帯域解放を要求する．TTCP は，アプリケーションからの帯域割当て/解放の要求に応じて，帯域管理サーバに帯域割当て/解放を要求する．

帯域管理サーバは，式 (1) が成立する場合，要求された帯域（RT_{new}）が割当て可能であると判断し，帯域を割り当てる．ここで，$\sum_i RT_i$ は RT 通信用に割当て済みの総帯域幅，PBW は全物理帯域幅，BBW は余裕帯域幅☆☆，$LNBW$ は NRT 通信用最低保証帯域幅を示す．

$$RT_{new} + \sum_i RT_i < PBW - BBW - LNBW \tag{1}$$

3.2.1.1.2 NRT 通信

NRT 通信の帯域割当て方針は，各ノードに 1 つ NRT 通信を行うための帯域を割り当てる．帯域の割当てをノードごとにすることで，帯域管理サーバは，管理する NRT 通信の帯域数が減り，負荷が軽減される．各ノードでは，NRT 通信を帯域が制限された仮想的なネットワークとして扱い，既存通信プロトコルがこの仮想的なネットワークに対して送信することで，既存通信プロトコルそのものに変更を加えることなくトラフィックを抑制できる．

各ノードは，上位通信プロトコルから NRT 通信が要求されると，帯域管理サーバに帯域割当てを要求する．また，各ノードは，上位通信プロトコルから NRT 通信が一定時間以上要求されない場合に，帯域管理サーバに帯域解放を要求する．帯域管理サーバは，各ノードの要求に応じて，ノードに NRT 通信の帯域を割り当てる．NRT 通信の帯域は，RT 通信と異なり，帯域管理サーバの指示により帯域幅が指定される．

帯域管理サーバは，全物理帯域幅から余裕帯域幅と RT 通信用に割当て済みの総帯域幅を引き，NRT 通信用の総帯域幅を計算する．帯域管理サーバは，NRT 通信用の総帯域幅を，NRT 通信を要求するノードに等配分する．このため，RT 通信を含むすべての帯域割当て/解放にともない，NRT 通信を要求する全ノードに帯域幅の変更を通知する．1 ノードに割り当てる帯域幅 NRT_{node} は，NRT 通信を要求するノード数を n として，式 (2) にて決定する．

$$NRT_{node} = \frac{PBW - BBW - \sum_i RT_i}{n} \tag{2}$$

3.2.1.2 パラメータ

TTCP/ITM 方式では，帯域管理用のパラメータとして，余裕帯域幅（BBW），NRT 通信用最低保証帯域幅（$LNBW$）が，セグメントごとに指定可能である．また，全物理帯域幅（PBW）は，物理層の種類によって決定される．

☆ たとえば，IP アドレスの使用は IP プロトコルに依存する．

☆☆ 総トラフィック抑制のため，帯域を割り当てない帯域幅

表 1 帯域割当て要求の例
Table 1 An example of bandwidth requests.

ノード名	種類	要求帯域幅
A	RT	1.5 Mbps
B	RT	3.0 Mbps
B	NRT	–
C	NRT	–
D	NRT	–

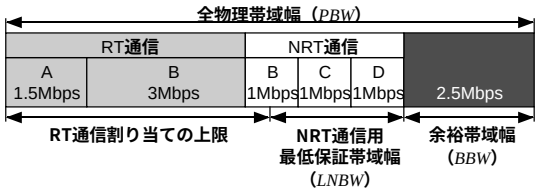


図 4 帯域割当て状況の例
Fig. 4 An example of bandwidth allocation.

たとえば，物理帯域幅 10.0 Mbps の Ethernet に，余裕帯域幅 2.5 Mbps，NRT 通信用最低保証帯域幅 2.5 Mbps をとり，**表 1** の帯域割当て要求を受け付けた場合の帯域割当て状況を，**図 4** に示す．

NRT の帯域割当ては，最近，NRT 通信を用いた送信を行ったノードに均等に割り当てる．また，最後に送信が行われてから一定時間，帯域を保持する．これにより，NRT 通信を必要としないノードに帯域を割り当てる無駄を省くとともに，送信が間欠的に起きるリモート端末アプリケーションなどで，帯域の割当て/解放処理の発生頻度を低減できる．

3.2.2 Total Traffic Control Protocol

TTCP/ITM 方式では，セグメント上の 1 ノードに帯域管理サーバを設け，セグメント内のすべてのノードからの帯域割当て要求を収集し，総帯域管理を行う．セグメント内の各ノードは，このサーバに対して，TTCP を用いて帯域の割当て要求や解放要求を行う．TTCP では，**表 2** に示す種類の制御パケットを設けている．

制御パケットの欠落がない場合の TTCP におけるコマンド交換手順を**図 5** に示す☆．

図 5 では，送信ノードから帯域管理サーバに帯域割当て要求コマンド（ReserveReq）が送られる．帯域管理サーバは，この要求を受け付け，返答として帯域割当て通知コマンド（ReserveAck）をブロードキャストする．このブロードキャストは，RT 通信の帯域割当てにともない，NRT 通信の帯域幅に変化が生じるため，NRT 通信を使用中の全ノードに変化後の帯域幅を通知し，更新させる機能を兼ねる．

帯域管理サーバは，送信ノードのダウンに備え，割り当てた帯域ごとに強制解放時刻を設定する．強制解放時刻を経過しても，送信ノードから帯域使用継続コマンドが到着しなかった場合，送信ノードがダウンしたと判断し，帯域を解放する．帯域の強制解放に先立ち，強制解放の予告を送信ノードに通知し，帯域使用継続コマンドの送信を促す．送信ノードから帯域使用継続コマンドが到着すると，強制解放時刻を再設定し，帯域の使用時間を延長する．

送信ノードは，サーバが設定した強制解放時刻より長く帯域を使用する場合，強制解放の予告が通知される前に，帯域使用継続コマンドを送信する．これにより，強制解放の予告コマンド送信処理を省き，サーバ側の処理負担を軽減する．

送信ノードとサーバ間のタイマのずれや，帯域継続コマンドの紛失が発生すると，送信ノードは，サーバ

☆ 制御パケットの欠落は，Ack 到着まで再送することで対処する．

表 2 TTCP コマンドの種類
Table 2 Sorts of TTCP command.

名前	機能	
	帯域管理サーバ宛（Req）	サーバからの応答（Ack）
Arp	アドレス要求	–
ArpReply	–	アドレス通知
Reserve	帯域割当て要求	帯域割当て通知
Cancel	帯域解放要求	帯域解放通知
Continue	帯域使用継続要求	–
Reply	–	OK/NG
Timeout	–	強制解放の予告

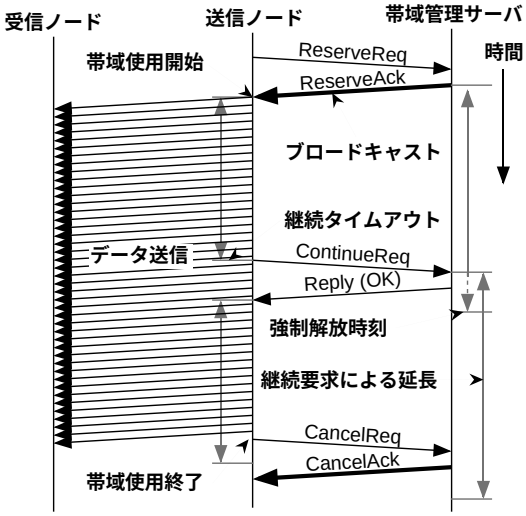


図 5 制御コマンドの交換手順
Fig. 5 TTCP negotiation.

から強制解放の予告コマンドを受け取る．この場合，送信ノードは，帯域使用継続コマンドを無条件で送信し，帯域管理サーバに帯域を強制解放しないよう通知する．

3.2.3 Isochronous Transfer Mode

ITM は，一定周期の送信と，Ethernet の持つ再送メカニズムを利用し，送信量抑制，最大遅延時間保証を実現する Traffic Shaping 方式である．

各ノードは，帯域管理サーバから割り当てられた帯域に従い，各周期ごとのデータ送信量を抑制することで，セグメント内の総帯域使用率を一定値以下に抑えるとともに，アプリケーションから要求された送信を最大遅延時間以内に完了させる．

各ノードでは，一定周期ごとに，1 周期分に相当する送信データの送信を 1 周期以内に行う．たとえば，10 ミリ秒周期に 5 パケット分の帯域を確保している場合，最初の周期では 5 パケット送信し，6 パケット目の送信は，次の周期まで待つ．すなわち，最初の 5 パケットを周期内に送信できることを保証し，帯域幅の保証を実現する．

以上に述べた設計により，TTCP/ITM では，次のような QoS を保証する．

(1) 帯域幅 ：アプリケーションが指定した値
(2) 遅延分散：ITM の周期以下
(3) 遅延時間：ITM の周期以下
(4) 紛失率 ：総帯域管理（余裕帯域幅）の設定により変更可能

4. TTCP/ITM の実装

ここでは，連続メディア処理向きマイクロカーネル HiTactix [3)~7)] への ITM と帯域管理サーバの実装，および，FreeBSD への ITM 実装について述べる．

4.1 HiTactix への ITM 実装

HiTactix への実装では，**図 6** に示す一般的な IP プロトコルの実装に，TTCP/ITM の機能を提供する ITM モジュールを，**図 7** に示すようにデバイスドライバ層とプロトコル層の間に挿入する．この方式では，デバイスドライバ，既存プロトコル双方とも変更不要である．

ITM モジュールは，HiTactix が提供するアイソクロナス・スケジューリング機構[6),8)]を利用して，10 Mbps Ethernet の場合には 40 ミリ秒，100 Mbps Ethernet の場合には 10 ミリ秒☆の周期で起動する．アイソクロナス・スケジューリング機構は，スレッド起動周期の誤差が CPU 利用率にかかわらず，数百 μ 秒以下を保証可能[8)]であり，ITM モジュールは，高い精度で一定間隔に起動される．

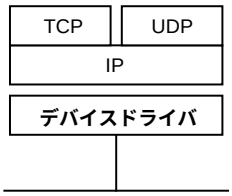


図 6 一般的な IP プロトコル階層
Fig. 6 General IP protocol stack.

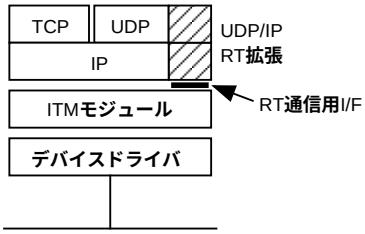


図 7 TTCP/ITM HiTactix への実装
Fig. 7 TTCP/ITM implementation for HiTactix.

ITM モジュールには，RT 通信ポートごと，および，NRT 通信用に 1 つ，送信要求を一時格納するキューを持つ．ITM モジュールが起動するたびに，割り当てられた帯域に相当するデータ量の送信要求をキューから取り出し，デバイスドライバに送信を依頼する．送信の依頼を受けたデバイスドライバは，ただちにハードウェアを起動し送信を行う．

このように ITM モジュールは，ITM モジュール起動ごとに送信するデータ量を，割り当てられた帯域幅に抑え，トラフィックを一定値以下に抑えている．

さらに ITM モジュールは，RT 通信の機能を利用する上位層のために，以下に示す RT 通信用インタフェースを提供している．

(1) 帯域確保：`allocate_bandwidth`
指定した Ethernet 上に，指定した帯域☆☆を確保し，その識別子を返す．
(2) 帯域解放：`cancel_bandwidth`
指定した帯域を解放する．
(3) 送信要求：`rt_send`
指定した帯域に対する送信要求を ITM モジュールに渡す．

HiTactix のソケット・インタフェースでは，データグラム転送（UDP/IP）にオプションとして帯域幅を

☆ およそ再送 9 回の最長待ち時間に相当．

☆☆ アプリケーションの送信周期と，その周期に送信する最大データ量．

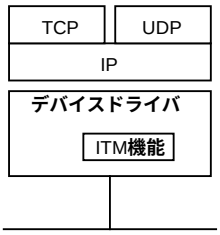


図 8 TTCP/ITM FreeBSD への実装
Fig. 8 TTCP/ITM implementation for FreeBSD.

指定可能にした RT 拡張を施し，QoS を保証した送信機能を提供する．本機能により，受信ノードは，QoS を保証したデータグラムを，RT 通信用の拡張を施すことなく，既存 UDP/IP プロトコルスタックにより受信できる．

4.2 FreeBSD への ITM 実装

FreeBSD は，高精度な周期スケジューリング機構を持たない汎用 OS である．一般に，汎用 OS で，他プロセスの負荷にかかわらず，ITM モジュールを正確に周期実行することは容易ではない．

この問題を，FreeBSD への実装では，**図 8** に示すようにデバイスドライバに ITM 機能を組み込むことにより解決している．このデバイスドライバでは，インターバル・タイマ機能を持つ Ethernet コントローラを用い，ITM の送信契機に必要な周期ごとに割込みを発生させ，ITM 機能を周期的に起動する．

この実装方法を採用すると，デバイスドライバが置き換え可能な多くの汎用 OS に対して，OS 内を改造することなく，TTCP/ITM 機能☆を組み込むことが可能である．すなわち，上位プロトコルや，アプリケーションに影響なく，TTCP/ITM の NRT 通信機能を組み込むことが可能であり，他ノードの RT 通信を阻害することなく，Telnet や FTP など，既存の非リアルタイム・アプリケーションを利用することができる．

4.3 HiTactix への帯域管理サーバ実装

帯域管理サーバは，固有の Ethernet フレームタイプを割り当てた TTCP プロトコル・スタック上のサーバ・モジュールとして OS 内に実装した．3.2.1.2 項にて述べたとおり，帯域管理サーバの起動時に，オプションとして，余裕帯域幅，NRT 用最低保証帯域幅を指定できる．

このほか，現実装において，帯域管理サーバは，3.2.2 項に示した帯域ごとに設定する強制解放時刻を，帯域割当て通知の約 10 秒後としている．また，強制解放の予告を，強制解放時刻の 1 秒前としている．

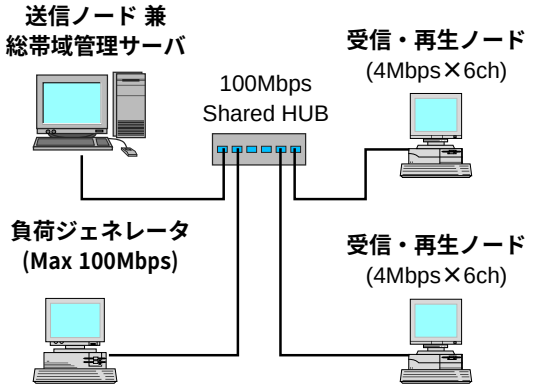


図 9 ビデオデータ転送実験の構成
Fig. 9 Configration of experimental VOD system.

5. TTCP/ITM の評価

ここでは，上述の TTCP/ITM の実装を用い，100 Mbps Ethernet 上で行った実験結果を示す．実験は，ビデオデータ転送実験，および，8 ノードを用いた連続衝突計測実験の 2 種類である．

5.1 ビデオデータ転送実験

ビデオデータ転送実験は，ビデオデータ転送アプリケーションが，他ノードからのネットワーク負荷によって受ける影響を調べ，TTCP/ITM 方式の有効性を調べる．

図 9 は，実験システムの構成を表している．なお，帯域管理サーバは，全物理帯域幅 100 Mbps に対して，余裕帯域幅を 10 Mbps，NRT 用最低保証帯域幅を 10 Mbps に設定している．また，ITM の周期は，10 ミリ秒である．

ビデオデータ転送アプリケーションは，送信ノードが，1 ストリームあたり約 4 Mbps☆☆の非圧縮ビデオデータを，2 台の受信・再生ノードに各々6 ストリームずつ，合計 48 Mbps☆☆☆を送信し続け，再生ノードが非圧縮ビデオデータを受信，再生し，同時にデータ到着率の表示を行うシステムである．負荷ジェネレータは，HiTactix が提供する UDP パケットのバースト送信機能を用い，連続的にデータを送信し続けるアプリケーションにより，100 Mbps Ethernet を飽和可能なネットワーク負荷を生成する．

図 10 は，負荷ジェネレータに ITM を実装しない場合について，送信ノード，および，負荷ジェネレータの送信データ量を 10 ミリ秒間隔で実測した結果で

☆ NRT 通信のみ．RT 通信を扱うには，上位プロトコルスタックなどの拡張が必要．

☆☆ 160×120 pixel，16 bit/pixel，12.5 fps

☆☆☆ MPEG1 に換算すると，32 チャネル相当．

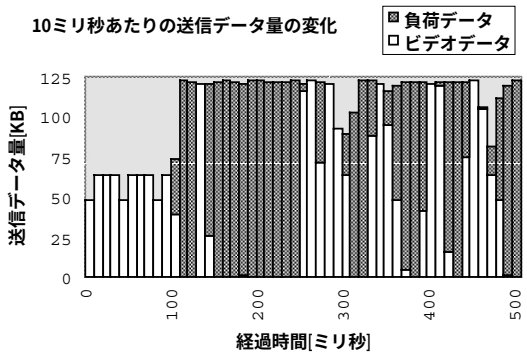


図 10 ITM なしの送信データ量
Fig. 10 Video-data transfer interfered by NRT traffic.

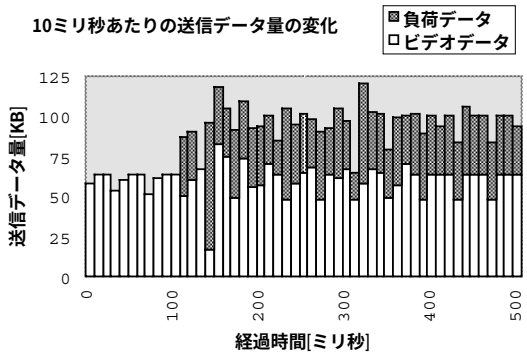


図 11 ITM ありの送信データ量
Fig. 11 Video-data transfer using TTCP/ITM.

ある．負荷ジェネレータが送信を開始する（経過時間 100 ミリ秒付近）と同時に，送信ノードからの送信量が影響を受けている．再生ノードでのビデオデータの再生は，コマ落ちが激しく，データの到着率を示す値は 50〜80%であった．

一方，**図 11** は，負荷ジェネレータに ITM を実装した場合について，送信ノード，および，負荷ジェネレータの送信データ量を 10 ミリ秒間隔で実測した結果である．負荷ジェネレータの送信により，送信開始に遅れが発生しているが，いずれも次の送信タイミングまでに回復しており，ITM 周期以内に送信が完了している．送信ノードからの送信量，および，負荷ジェネレータからの送信量は，前後の平均をとるとほぼ一定に保たれ安定している．再生ノードでのビデオデータの再生は正常で，データの到着率を示す値も 100%を維持していた．

この結果から，TTCP/ITM 方式によって，QoS 保証が実現可能であることが分かる．

5.2 連続衝突計測実験

連続衝突計測実験は，総帯域使用率の変化によって，

表 3 通信品質の実測結果（%）
Table 3 Mesured successive collision and loss ratio (%).

総帯域使用率	57.6	67.2	76.8	86.4	96.0
10 回以上	0.34	0.80	2.51	3.25	3.93
13 回以上	0.02	0.11	0.69	1.71	2.75
16 回（紛失）	0.00	0.02	0.19	0.88	1.92
遅延発生率	0.00	0.00	0.00	0.10	4.21

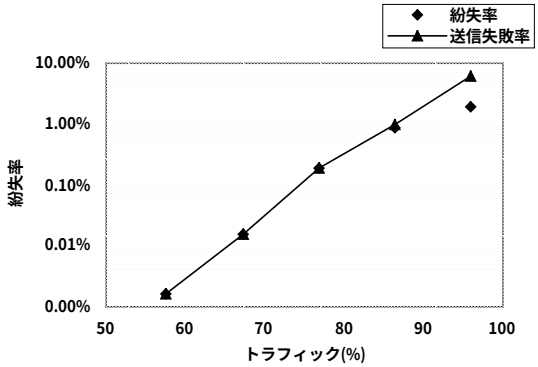


図 12 トラフィック–紛失率の関係
Fig. 12 Traffic vs. packet loss ratio.

遅延時間の達成率，パケットの紛失率☆がどのように変化するかを調べる．ここでは，送信時の連続衝突発生回数を計測し，遅延発生の評価指標とした．

100 Mbps 用共有ハブに Pentium☆☆ 90〜200 MHz の PC を 8 ノード接続し，全ノードが同時に，同一帯域幅の送信を行った．帯域幅の指定を変え，総帯域使用率を変化させたときの，各ノードが送信する全パケットの送信衝突回数を，全ノードで計測した．なお，余裕帯域幅を 0 に設定し，およそ 120 秒間計測を行った．

表 3 は，この結果のうち，連続衝突 10，13，16 回以上の発生率，および，遅延発生率を示している．遅延発生率とは，ITM の送信周期を超えても前周期の送信が完了しないデッドライン・ミスの発生率を表している．デッドライン・ミスが発生した場合には，1 周期分の送信データが紛失する．

図 12 は，連続 16 回衝突が発生し，送信できなかったパケットの発生率（紛失率）を示している．

この結果から，トラフィックを 67.2%以下に抑えた場合，パケットの紛失率を 0.02%以下に抑えることが可能であることが分かる．また，デッドライン・ミスは，トラフィックが 76.8%以下では発生していない．

この結果から，余裕帯域幅の設定を変更し，最大遅

☆ ここでは送信時の紛失に限定し，受信バッファの不足などによる紛失は，考慮しない．
☆☆ Pentium は，米国 Intel Corporation の登録商標である．

表 4 SB 機能を適用した場合の実測結果（%）
Table 4 Mesured successive collision and loss ratio with SB mechanism (%).

SB 機能	全ノード適用			1 ノード適用		
総帯域使用率	76.8	86.4	96.0	76.8	86.4	96.0
10 回以上	0.00	0.00	0.02	0.00	0.00	0.00
13 回以上	0.00	0.00	0.01	0.00	0.00	0.00
16 回（紛失）	0.00	0.00	0.01	0.00	0.00	0.00
遅延発生率	0.00	0.00	2.83	0.00	0.00	0.10

延時間や，紛失率の QoS を制御可能であることが分かる．LAN に TTCP/ITM を適用する場合，要求品質に応じて，ネットワーク管理者が余裕帯域幅を設定できる．

5.3 特殊モード

市販されている Ethernet コントローラには，Ethernet の規格外の高性能化機能をオプションとして備えているものがある．ここでは，Ethernet コントローラ（Intel 2114x）[9]に備えられている SB☆機能を用いた場合の結果について記述する．

SB 機能は，2.1 節の (6) に示す衝突検出後の待ち時間を計測するタイマのカウントダウンを，他ノードが送信を行っている間，停止する．衝突検出後の待ち時間が増加し，規格範囲を超過する．

表 4 は，5.2 節と同一条件を用いた実験の結果を示している．「全ノード適用」は，全ノードに SB 機能を適用した場合の実測結果を示す．また，「1 ノード適用」は，1 ノードのみ SB 機能を適用し，残り全ノードを従来方式を適用した場合の，SB 機能を適用したノードの実測結果を示す．なお，SB 機能を無効にしたノードの実測結果は，表 3 の値にほぼ一致した．

全ノード適用の場合と，1 ノード適用の場合，双方において，SB 機能が適用されたノードの送信品質は高い．これは，以下の理由による．

伝送メディアにフレームが送信されていない場合，複数のノードが送信を開始する時刻がわずかでもずれていれば，送信衝突は起らない．しかし，伝送メディアにフレームが送信中の場合，そのフレーム送信中に送信を開始しようとしたすべてのノードが，そのフレームの直後に送信を開始し，必ず衝突を起してしまう．Ethernet の仕様に従うと，backoff 待ちが終了した瞬間，フレーム送信中であれば再び送信衝突を起す可能性が高まる．これに対し，SB 機能を適用すると，フレーム送信中に backoff 待ちが終了することがなくなり，次の送信が確実に遅延される．SB 機能と TTCP/ITM 方式を組み合わせた場合，送信周期あたりの送信要求データ量が制限されており，送信周期内に伝送メディア上にフレームが存在しない時間が確実に生じ，送信が成功する確率が高まる．このことから，SB 機能によって，送信衝突の連続発生が回避されていると考えられる．

先に記述したとおり，SB 機能は Ethernet の規格外の仕様であるが，他ノードの動作に影響を及ぼすことがなく，混在利用が可能である．

6. 関連研究

6.1 RTP, RSVP

連続メディアを含む，マルチメディア・データの通信に適した Transport Protocol として，RTP [10]が提案されている．しかし，RTP では，データ転送に関して，受信品質のフィードバック情報を提供するが，QoS 保証は，下位レイヤの実装に依存している．RTP の下位レイヤとして，TTCP/ITM を組み合わせることにより高品質化が可能となる．

また，帯域管理に関しては，RSVP（Resource ReSerVation Protocol）[11]の標準化が進められている．これは，通信経路上のルータに対する資源予約を主な目的としたシグナリング・プロトコルである．これを，Ethernet などの共有メディア型 LAN に対して適用する方式として，SBM（Subnet Bandwidth Manager）[12]が検討されている．SBM は，TTCP 相当の機能のみを提供し，ITM 相当の機能を欠いている．スイッチング・ハブの機能を拡張し，パケットの優先度付けを可能にするハードウェア機能を利用する検討が行われているが，Shared Ethernet における QoS 保証は考慮されていない．TTCP/ITM は，SBM と協調することにより，RSVP を補完し，Shared Ethernet に対して資源予約を可能とすることができる．

6.2 RETHER

RETHER [13]は，TTCP/ITM と同様に，Ethernet のハードウェア仕様を変更せず，リアルタイム通信を可能にすることを目的としている．両者の違いは，TTCP/ITM が総帯域管理と Traffic Shaping を用いて QoS を保証するのに対し，RETHER がトークンを周期的に巡回させ，送信衝突を回避し，QoS を保証する点である．

しかし，RETHER のトークン・パッシング方式は，ノードの処理能力や負荷状況によって，トークン・パッシングのオーバヘッドによる遅延時間が変化すること，また，トークンが失われた場合に，回復するまでリアルタイム通信の QoS 保証が困難なことなどが問題となる．TTCP/ITM では，各ノードが自律的に周期送

☆ Start/Stop Backoff Counter

信するため，あるノードの障害が他ノードに影響を及ぼさない．

7. おわりに

本論文では，Ethernet のハードウェア仕様や，上位プロトコルを変更することなく，Ethernet 上で QoS 保証を実現する TTCP/ITM 方式を提案した．TTCP/ITM 方式は，セグメント内の総トラフィックを一定値以下に抑えるとともに，各ノードにおいて送信データに Traffic Shaping を施すことにより，遅延時間，および，遅延分散の増大を回避可能にする．

さらに，TTCP/ITM 方式を HiTactix に実装し，非圧縮ビデオデータ転送実験を行い，その有効性を評価した．また，複数ノードを用いた送信実験を行い，総割当て帯域を全物理帯域の 76.8%に抑えることで，パケットの紛失率を 0.2%以下に，また，総割当て帯域を全物理帯域の 67.2%に抑えることで，パケットの紛失率を 0.02%以下に抑えられることを示した．

Ethernet は，そのハードウェアの特性上，QoS 保証が困難なネットワークとして考えられていたが，TTCP/ITM を組み込むことで，高品質なビデオデータの転送に利用可能であることを確認した．

参考文献

1) Mehra, A., et al.: Structuring Communication Software for Quality-of-Service Grarantees, *IEEE Proc. 17th RTSS*, Washington, DC, pp.144–154 (Dec. 1996).
2) Partridge, C.: Gigabit Networking, Addison-Wesley (May 1994).
3) Iwasaki, M., et al.: Isochronous Scheduling and its Application to Traffic Control, *IEEE Proc.19th RTSS*, Madrid, pp.14–25 (Dec.1998).
4) Iwasaki, M., et al.: A Micro-kernel for Isochronous Video-Data Transfer, *Proc. WWCA '97*, Tsukuba, Lecture Notes in Computer Science, vol.1274, pp.334–349, Springer (Mar. 1997).
5) 岩嵜ほか：連続メディア処理向きマイクロカーネル HiTactix の設計と評価，コンピュータシステム・シンポジウム論文集，pp.99–104 (Nov. 1996).
6) 中原ほか：連続メディア処理向けマイクロカーネルにおける排他制御方式，コンピュータシステム・シンポジウム論文集，pp.71–78 (Nov. 1998).
7) 川田ほか：連続メディア処理向けカーネルの性能評価，第 57 回情報処理学会全国大会論文集（1），pp.52–53 (Oct. 1998).
8) 竹内ほか：連続メディア処理向き OS の周期駆動保証機構の設計と実装，情報処理学会論文誌，Vol.40, No.3, pp.1204–1215 (1999).
9) Digital Semiconductor 21140A PCI Fast Ethernet LAN Controller Hardware Reference Manual, Digital Equipment Corp. (Nov. 1996).
10) Schulzrinne, H., et al.: RFC-1889: RTP: A Transport Protocol for Real-time Applications (Jan. 1996).
11) Barden, R., et al.: RFC-2205: Resource ReSerVation Protocol (RSVP) – Version 1 Functional Specification, RFC-2205 (Sep. 1997).
12) Yavatkar, R., et al.: SBM (Subnet Bandwidth Manager): A Protocol for RSVP-based Admission Control over IEEE 802-style networks, Internet Draft, draft-ietf-issll-is802-sbm-07.txt (Nov. 1998).
13) Venkatramani, C. and Chiueh, T.: Design, Implementation, and Evaluation of a Software-based Real-Time Ethernet Protocol, *SIGCOMM '95* Cambridge, pp.27–37 (1995).

(平成 11 年 4 月 26 日受付)
(平成 11 年 10 月 7 日採録)

中野 隆裕（正会員）

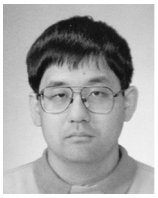
昭和 44 年生．平成 5 年電気通信大学電気通信学部情報工学科卒業．平成 7 年同大学大学院電気通信学研究科情報工学専攻修士課程修了．同年（株）日立製作所システム開発研究所入社．連続メディア処理向きマイクロカーネルの研究，特に入出力方式やリアルタイム通信方式に関する研究・開発に従事．

岩嵜 正明（正会員）

昭和 33 年生．昭和 56 年九州工業大学工学部電子工学科卒業．昭和 58 年九州大学大学院総合理工学研究科情報システム学専攻修士課程修了．同年（株）日立製作所中央研究所入所，平成 5 年より同社システム開発研究所勤務．並列推論マシン，金融 OLTP システム，並列計算機 SR2201 の OS 研究開発を経て，HiTactix の研究を開始，現在に至る．

中原　雅彦（正会員）

昭和 40 年生．昭和 63 年東京農工大学工学部数理情報工学科卒業．平成 2 年同大学大学院工学研究科修士課程修了．同年（株）日立製作所システム開発研究所入社．ワークステーションの性能評価，並列計算機用オペレーティングシステム，連続メディア処理向きマイクロカーネル等の研究・開発に従事．



竹内　　理（正会員）

昭和 44 年生．平成 4 年東京大学理学部情報科学科卒業．平成 6 年同大学大学院理学系研究科情報科学専攻修士課程修了．同年（株）日立製作所システム開発研究所入社．連続メディア処理向きマイクロカーネルの研究，特にリアルタイムスケジューリング方式，リアルタイム通信方式，ヘテロジニアス OS アーキテクチャに関する研究に従事．