社団法人　情報処理学会　研究報告
IPSJ SIG Technical Reports

2005－QAI－14（17）
2005／1／19

利用者環境を考慮した相互通信セッションの実現

橋本浩二　　柴田義孝

岩手県立大学ソフトウェア情報学部
〒020-0193 岩手県岩手郡滝沢村巣子152-52
tel. : 019-694-2552, e-mail : {hashi, shibata}@iwate-pu.ac.jp

ブロードバンドネットワークのサービスとして，テレビ電話やビデオ会議などの通信サービスは実用的なものとなった．しかし，通信端末の性能や利用可能な帯域は，利用者の通信環境によって異なるので，適切なフォーマットによる相互通信を実現するための仕組みが必要となる．RTP で定義されているトランスコーディング機能は，利用者の通信環境を考慮した相互通信を実現するために有効な機能であるが，通信環境の異なる複数の参加者を想定したトランスコーディング機能の動的な利用方法やシステムは確立していない．そこで本稿では，利用者環境に応じてトランスコーディング機能をネットワーク上へ動的に配置し，適切なフォーマットによる相互通信セッションを実現するためのミドルウェアを提案する．

キーワード : 相互通信セッション，ビデオ会議，トランスコーディング機能

Adaptive Intercommunication Session for Users' Environment

Koji Hashimoto　　Yoshitaka Shibata

Faculty of Software and Information Science, Iwate Prefectural University
152-52 Sugo, Takizawa, Iwate 020-0193, Japan
tel. : 019-694-2552, e-mail : {hashi, shibata}@iwate-pu.ac.jp

On broad band networks, we can communicate with each other by intercommunication services such as video phone, teleconference system and so on. However, our computer's capabilities and available network bandwidths are different respectively. Therefore, it is important for the communication system to use suitable formats according to users' communication environments. Although the transcoding function defined by RTP is a useful function to realize intercommunication adapted users' communication environments, dynamic transcoding mechanisms have not established for multi-user environments. In this paper, we propose a middleware system to realize intercommunication services with suitable media format according to users' environments. The middleware system integrates multicast communication sessions by using relocatable transcoding functions.

Keyword : Intercommunication Session, Video Conference, Transcoding Functions

1．はじめに

近年におけるネットワーク技術の進歩により，インターネットの末端において利用可能な帯域幅も急速に増加し，映像や音声を扱う大容量のデータ転送を必要とするアプリケーションが日常的に利用されるようになった．蓄積型ビデオストリームの配信に加え，テレビ電話やビデオ会議など相互通信を実現するシステムも多数存在し，遠隔地間でのコミュニケーションが容易に行えるようになった．

しかしながら既存の相互通信システムは，H.323 やSIP[1]によるシグナリングプロトコル上でH.263/264, MPEG4などのビデオフォーマットを用いるものが多く，利用者の通信環境やQoS要求に応じて複数のビデオフォーマットを用いた相互通信を実現するための十分な仕組みは実現されていない．例えば既存の相互通信システムを利用して，DV による通信セッションと MPEG4 による通信セッションを動的に統合し，1つの相互通信を実現することは困難である．

ここで，RTP[2]で定義されているトランスコーディングやミキシングの機能が利用できれば，例えば DV ストリームを MPEG4 ビデオストリームにトランスコーディングすることにより，利用者の通信環境やQoS要求に応じて適切なフォーマットを用いた相互通信が可能となる．しかしながら，複数の参加者を想定した相互通信におけるトランスコーディングやミキシング機能[3,4]の動的な利用に関する方法論やシステムは確立していない．そこで本稿では，利用者環境に応じてトランスコーディング機能をネットワーク上へ動的に配置し，適切なフォーマットによる相互通信セッションを実現するためのミドルウェアを提案する．

これまで筆者らは，柔軟な相互通信環境構築のために必要とされる機能をミドルウェアとして実現するアーキテクチャを考案し，トランスコーディングエージェントの動的な配置に関する報告[5]をしてきた．本稿では，柔軟な相互通信機能を実現するために移動エージェントベースのトランスコーディング機能を利用し，複数のマルチキャストセッションを動的に接続する仕組みを実現する．

2．MidField システム概要

筆者らの提案する **MidField** (**Mid**dleware for **F**lexible **i**ntercommunication **e**nvironment by **r**elocatable **d**ecision) システムの機能モジュール構成を図1に示す．本システムは，ネットワーク接続されたコンピュータシステムのトランスポート層より上位に位置し，アプリケーション

プログラムに対してマルチメディア通信機能を提供する．Stream Plane はマルチメディアストリーム転送処理を行い，Session Plane は通信セッションの管理を行う．また，System Plane ではネットワークトラフィックや CPU 利用状況の監視を行い，システム利用者が要求する QoS に対するアドミッションテストを実行する．そして Event Process Plane では，システム内部で発生する各種のイベントを処理する．

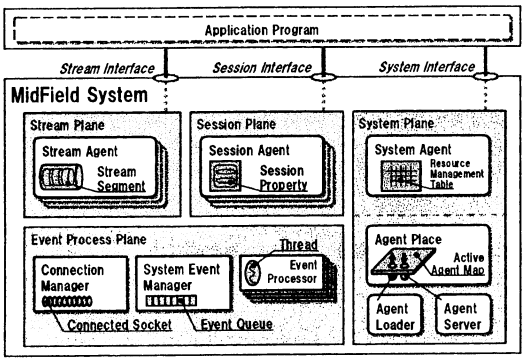


図1：MidField システム機能モジュール構成

Stream Agent は，RTP ストリーム処理を実現するために Stream Segment を利用し RTP ストリームの送受信およびトランスコーディングを行う．Session Agent は Session Property を所有し，複数の RTP セッションを統合して MidField セッションを実現するための管理機能を実現する．また，System Agent はローカルコンピュータシステムの CPU 資源やネットワークトラフィックを監視し，システム利用者が MidField セッションへ参加する際にはアドミッションテストを実行する．一方，Agent Place はエージェントの生成・起動，移動，終了処理を行い，ローカルシステム内のエージェントを管理する．そして，エージェントの移動を実現するための Agent Loader と Agent Server を利用する．Agent Place も MidField システムにおけるエージェントの 1 つである．

これらのエージェントはそれぞれ，コンピュータネットワーク上で稼動する MidField システム間でメッセージの送受信が可能である．Event Process Plane の Connection Manager は，MidField システム間を接続したソケットインターフェースを保持し，エージェント間メッセージ通信の排他的制御を行う．また，System Event Manager は，各エージェントが発行するシステム内部イベントを Event Queue に格納し，Event Processor の Thread でイベントを処理する仕組みを実現する．

ここで，MidField システムにおける相互通信セッションを MidField セッションと呼ぶ．1 つの MidField セッションは複数のマルチキャストセッションで構成され，各マルチキャストセッションで利用するメディアストリームを，動的に配置されるトランスコーディング機能を用いて中継することで，利用者の通信環境とサービス品質要求に応じた相互通信を実現する．

図 2 では，2 つのマルチキャストセッションで構成される MidField セッションに 3 つの利用者端末（MF_{U1}，MF_{U2}，MF_{U3}）が参加している．MF_{U1} と MF_{U2} は共に，DV ストリームによる相互通信が可能であるが，MF_{U3} は，帯域不足または DV 送受信処理に対する CPU 資源不足のため，DV ストリームを用いた通信が不可能であるとする．そこで，MF_{U1} と MF_{U2} の通信セッションに参加するために MF_{U3} は，MPEG4 ストリームによる送受信を要求したとする．ここで MidField システムは，必要となるトランスコーディング処理モジュールを，適切なトランスコーディングノード（MF_{T1}，MF_{T2}）に動的に配置する．図 2 では，MF_{T1}，MF_{T2} が DV ストリームを MPEG4 ビデオストリームへトランスコーディングするとともに，MF_{U3} が送信する MPEG4 ビデオストリームを MF_{T2} が中継することにより，MF_{U3} も相互通信が可能となる．

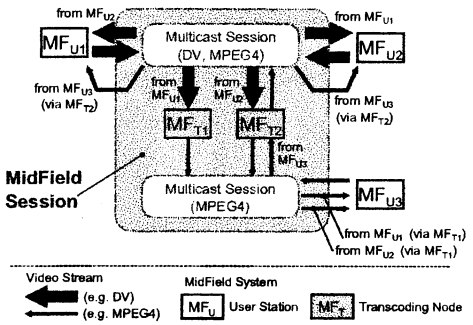


図2：MidField Session 概要

3．MidField セッション

上述した MidField セッションを実現するために，セッションエージェントが利用するセッションプロパティのデータ構造を図 3 に示す．

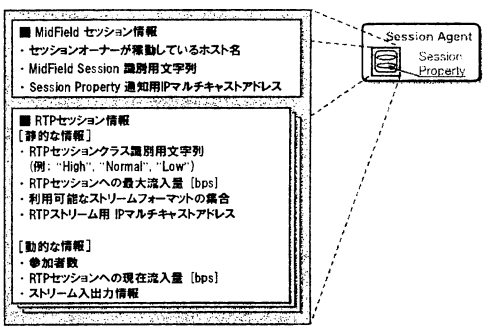


図3：セッションプロパティのデータ構造

セッションプロパティは MidField セッション毎に 1 つ存在し，1 つの MidField セッション情報と複数の RTP セッション情報により構成される．MidField セッションを生成するセッションエージェントをセッションオーナーと呼び，セッションオーナーがセッションプロパティを管理する．

MidField セッション情報は，セッションオーナーが稼動しているホスト名と MidField セッション識別用文字

列,そしてセッションプロパティ通知用 IP マルチキャストアドレスで構成される. これらのデータは MidField セッション生成時にセッションオーナーが決定し, MidField セッション終了時まで変更されない静的な情報である.

一方, RTP セッション情報は静的な情報と動的な情報により構成される. 静的な情報には, RTP セッションクラスを識別する文字列, RTP セッションへのストリームの最大流入量[bps]と利用可能なストリームフォーマットの集合, そして利用する IP マルチキャストアドレスが含まれる. 各 RTP セッションにおけるメディアデータ転送は, この IP マルチキャストアドレスを用いて行われる. 参加者は, 利用可能なストリームフォーマットと最大流入量を参照することにより, 各自の相互通信環境に応じて適切な RTP セッションを選択することが可能となる. 動的な情報には, RTP セッションへの参加者数, RTP セッションで用いられているストリームの現在流入量[bps]とストリームの入出力情報が含まれる. これらの動的な情報は, セッションオーナーにより適宜更新され, 更新されたセッションプロパティはセッションプロパティ通知用 IP マルチキャストアドレスを用いて参加者全てに通知される.

3．1 セッションへの参加とストリームの追加

MidField セッションを開始するためには, まずアプリケーションがセッションプロパティを用意する. 次に, セッションオーナーへセッションプロパティを与えることによって, セッションオーナーは MidField セッションを生成し, 相互通信が可能となる. 一方, MidField セッションへ参加するためには, セッションオーナーが稼動している MidField システムよりセッションプロパティを取得し, 適切な RTP セッションへの参加要求をセッションオーナーへ送信する. MidField セッションへ参加する際のメッセージフローを図 4 に示す. 参加者からの要求に対し, セッションオーナーはセッションプロパティを更新し, その参加者へ応答を返す. 一方, 更新されたセッションプロパティは全ての参加者へマルチキャストにて通知される. セッションからの退出とストリームの削除に関するセッションオーナーと参加者間のメッセージフローもほぼ同じ流れとなる.

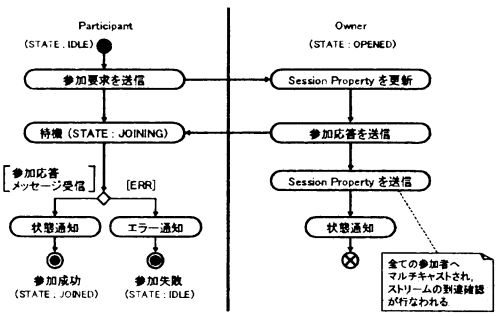


図4：セッションへの参加

ストリーム追加に関するメッセージフローを図 5 に示す. 参加者がストリームを追加する場合は, 追加する RTP セッションにおける最大流入量を超えぬよう, セッションオーナーがアドミッションテストを実行する. アドミッションテストの結果ストリームが追加可能である場合, それ以降のフローは図 4 と同様のものとなる.

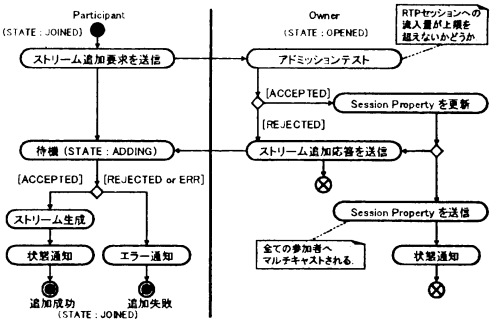
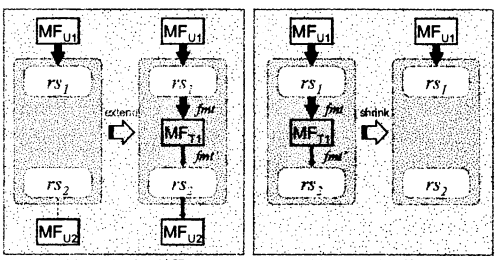


図5：ストリームの追加

このように, MidField システムにおける相互通信は, セッションオーナーが管理するセッションプロパティをもとに行われる. セッションへの参加/退出, ストリームの追加/削除要求を参加者から受信する毎に, セッションオーナーはセッションプロパティを更新して, MidField セッションの参加者全てにその更新内容を通知する. 更新されたセッションプロパティを受信した参加者はストリームの到達確認を行い, 必要に応じてストリームの拡張と縮退を実行する.

3．2 ストリームの拡張と縮退

更新されたセッションプロパティを受信した各参加者はストリームの到達確認をし, 必要に応じてストリームの拡張と縮退を行う. その概略を図 6 に示す.



(a) ストリームの拡張　(b) ストリームの縮退

図6：ストリームの拡張と縮退

図 6(a)に示す通り, 参加者のいる RTP セッションへストリームが到達していない場合, ストリームを拡張する. 一方, ストリーム到達確認の結果, 参加者のいない RTP セッションへストリームを拡張している場合, 図 6(b)に示す通りストリームを縮退させる.

ストリームを拡張する際は, 各 RTP セッションへ追加すべき出力フォーマットを, セッションプロパティをもとにセッションエージェント(参加者)が決定し, 適切なトランスコーディングノードをストリームエージェントが選択する. そしてストリームエージェントは, 自身の出力を入力とするトランスコーディングエージェントを生成し, 選択したトランスコーディングノードへトランスコーディングエージェントが移動する.

図7は，ストリーム拡張時のストリームエージェントの処理フローを示している．まず初めに，資源利用状況を収集するための要求メッセージを稼働中のトランスコーディングノードへマルチキャストする．これに対して，稼働中のトランスコーディングノードにおける System Agent は現在の資源利用状況を返信する．資源利用状況には，各トランスコーディングノードにおいて利用可能な入出力帯域幅の上限値と利用状況[bps] および CPU 利用率[%]が含まれる．資源利用状況を受信したストリームエージェントは，各トランスコーディングノードの資源利用状況を要素とする稼動システムリストを生成し，資源利用状況を受信する毎に追加更新する．

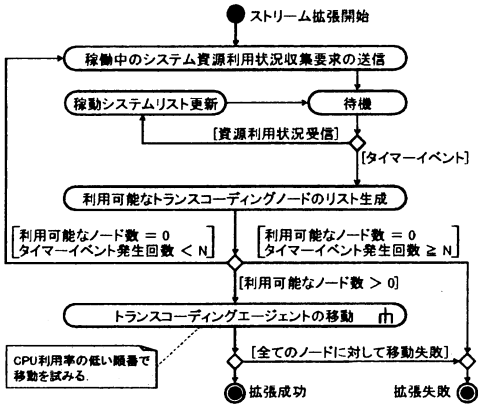


図7：ストリームの拡張

一方，ストリーム拡張開始時にストリームエージェントはタイマーを起動し，一定間隔で発生するタイマーイベントを受け取る．タイマーイベントを受け取ったストリームエージェントは，利用可能なトランスコーディングノードのリストを稼動システムリストをもとに生成する．リストの要素は，必要な送受信帯域幅と十分な CPU 資源をもつノードの資源利用状況である．そして，CPU 利用率の低いトランスコーディングノードから順番に移動を試みる．

もし，タイマーイベント発生回数が一定の値(N)未満で，かつ，利用可能なトランスコーディングノードが存在しない場合，ストリームエージェントは資源利用状況を収集するための要求メッセージを再度マルチキャストする．また，利用可能なトランスコーディングノードが存在せず，タイマーイベント発生回数が一定の値を超えた場合，必要な資源が確保できないと判断し，ストリームの拡張は失敗となる．次章で述べるプロトタイプシステムでは，タイマーイベントの発生間隔を1秒とし，発生回数の上限は3回としている．

トランスコーディングエージェントがストリーム拡張のために移動する際のフローは図8に示す通りである．まず，移動先のトランスコーディングノードにおける System Plane のエンティティに対し，ストリームエージェントはトランスコーディング処理のための入出力情報を送信する．次にトランスコーディングノードでは，トランスコーディング処理に必要となる CPU 資源と送受信帯域幅に対してアドミッションテストを実行し，確保可能であれば資源の予約を行う．そして，トランスコーディングエージェントがトランスコーディングノードへ移動し，処理を開始する．

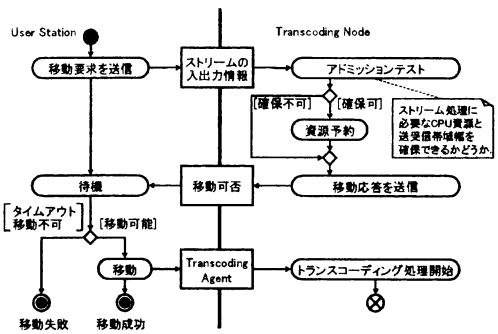


図8：トランスコーディングエージェントの移動

一方，各 RTP セッションで利用可能なストリームフォーマット全てに対してストリームの拡張が失敗した場合，MidField システムを利用するアプリケーションへ失敗の原因を含むイベントが通知され，ストリーム拡張処理は終了する．

3．3 CPU 利用率の計測

ストリーム拡張時におけるトランスコーディングノードの選択には，稼働中のシステムにおける利用可能な入出力帯域幅と CPU 利用率を用いる．ストリームの入出力情報から必要な帯域幅を見積もることは可能であるが，必要となる CPU 資源はトランスコーディングノードの性能によって異なる．そこで，MidField システムにおけるトランスコーディングノードは，トランスコーディング処理に必要となる CPU 資源を予め測定しておき，テーブルとして利用する．

CPU 利用率は，ノードの負荷を測る1つの値となる一方，一定の負荷状態においてもその値にばらつきが見られるので，平滑化した値を用いる必要がある．そこで，MidField システムでは，対象となるメディア処理のみを実行させたノードで CPU 利用率の1分間の移動平均をとり，移動平均の標準偏差が1未満となった時刻の移動平均値を必要な CPU 資源とする簡単な方法を導入した．

4．プロトタイプシステム

MidField システムのプロトタイプと簡単なビデオ会議アプリケーションを，Windows XP 上に Java，C，C++ 言語を用いて実装した．プロトタイプシステムの実装に利用した PC は，Dell WORKSTATION PWS360，Intel(R) Pentium(R) 4 3.20Ghz，2.00GB RAM のマシンで，ネットワークアダプタは Intel(R) PRO/1000 MT Network Connection を使用している．

MidField システムはトランスコーディング処理に必要となる CPU 資源を予め計測し，その結果を保持する．表1はプロトタイプシステムにおける計測結果の一部を示している．ここで，ビデオのフレームサイズは DV(720×480 [pixel])を基準とし，MJPEG と MPEG4 のフレームサイズは DV の 1/4(360×240 [pixel]) としている．PCM のフォーマットは，量子化ビット数 16[bit]，サンプリング周波数 32[kHz]，チャンネル数を2(ステレオ)とした．

表1：トランスコーディング処理とCPU利用率

入力フォーマット	出力フォーマット	CPU利用率	必要な帯域幅
中継処理			
DV/RTP	DV/RTP	17.4 %	入出力： 約28.8 Mbps
MJPEG/RTP	MJPEG/RTP	10.5 %	入出力： 約20.0 Mbps
MPEG4/RTP	MPEG4/RTP	1.0 %	入出力： 約0.5 Mbps
PCM/RTP	PCM/RTP	0.9 %	入出力： 1.024 Mbps
トランスコーディング処理			
DV/RTP	MJPEG/RTP + PCM/RTP	38.1 %	入力： 約28.8 Mbps 出力： 約20.0 Mbps
DV/RTP	MPEG4/RTP + PCM/RTP	20.5 %	入力： 約28.8 Mbps 出力： 約1.5 Mbps

表1より，プロトタイプシステムの実装で用いたPCでは，DVストリームをMJPEG+PCMストリームへトランスコーディングするために約38.1%のCPU資源を必要とし，DVストリームをMPEG4+PCMストリームにトランスコーディングするためには約20.5%のCPU資源を必要とする．これより，CPU資源のみを考慮するなら，DVストリームをMJPEG+PCMストリームにトランスコーディングする場合，1台のPCで2本分，MPEG4+PCMストリームにトランスコーディングするのであれば，1台のPCでも4本分の処理を受け持つことが可能であると期待できる．

そこで，MidFieldセッションの動的構成機能を評価するために，DVストリームとMPEG4+PCMストリームを用いた相互通信実験を行った．実験環境を図9(a)に示す．1台のギガビットスイッチングハブに6台の利用者端末(MF_{U1}〜MF_{U6})と2台のトランスコーディングノード(MF_{T1}, MF_{T2})を接続した環境で，図9(b)に示すMidFieldセッションを構成した．また，このMidFieldセッションは2つのRTPセッション(rs_1, rs_2)から構成される．rs_1で利用可能なフォーマットはDV，MPEG4，PCMとし，rs_2で利用可能なフォーマットはMPEG4，PCMとした．

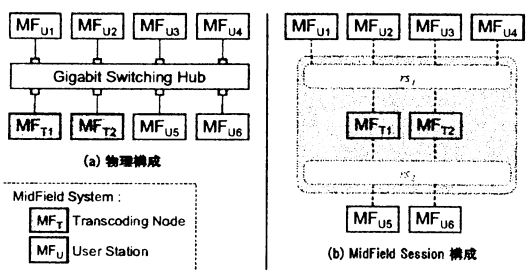


図9：評価実験環境

ここで，MF_{U1}〜MF_{U4}はRTPセッションrs_1を利用して相互通信を行い，MF_{U5}とMF_{U6}はRTPセッションrs_2を利用して相互通信を行う．必要に応じてトランスコーディングエージェントがトランスコーディングノード(MF_{T1}, MF_{T2})へ移動し，rs_1とrs_2を動的に接続することにより，DVストリームとMPEG4+PCMストリームによる6者間通信を行う．

以下(S1)〜(S8)は実験のシナリオである．

(S1) MF_{U1}〜MF_{U4}がrs_1で相互通信を開始する．
(S2) MF_{U5}がrs_2へ参加する．
(S3) MF_{U5}がストリームを追加し送信を開始する．
(S4) MF_{U6}がrs_2へ参加する．
(S5) MF_{U6}がストリームを追加し送信を開始する．
(S6) MF_{U6}がrs_2から退出する．
(S7) MF_{U5}がrs_2から退出する．
(S8) MF_{U1}〜MF_{U4}がrs_1から退出する．

実験結果として，MF_{U1}とMF_{T1}およびMF_{U5}とMF_{T2}におけるCPU利用率と入出力ビットレートを，図10から図13に示す．

まず，シナリオ(S1)の結果，4本のDVマルチキャストストリームがrs_1に流れる．図10の(S1)〜(S2)にかけて，MF_{U1}はDVストリームを1本送信し，その送信したストリームを含めて4本分のDVストリームの受信を開始したことがわかる．

次にシナリオ(S2)の結果，図11より，MF_{T1}がrs_1から4本分のDVストリームを受信し，MPEG4+PCMストリームへトランスコーディングし，rs_2への送信を開始したことがわかる．これは，MF_{U5}がrs_2へ参加したことにより，MF_{U1}〜MF_{U4}がrs_2へストリームを拡張する際，トランスコーディングノードとしてMF_{T1}を選択したことを意味する．

MF_{T1}がトランスコーディング処理を始めたことにより，RTPセッションrs_2へMPEG4+PCMストリームが4本流れ始める．これは，図12(S2)でMF_{U5}がrs_2へ参加した後，入力ビットレートが約6Mbpsで推移していることから確認できる．本プロトタイプで用いているMPEG4+PCMストリームのビットレートは最大約1.5Mbps程度なので，MPEG4+PCMストリームを4本受信すると，約6Mbpsとなる．

次にシナリオ(S3)の結果，MF_{U5}がrs_1へストリームの拡張を行った．この場合トランスコーディング処理は必要ではなく，中継のみが行われる．図13の(S3)より，その中継処理をMF_{T2}が担当していることがわかる．ここで，MF_{T2}はMEPG4+PCMストリーム1本をrs_2からrs_1へ中継するが，図13の通りMF_{T2}は5本分のMPEG4+PCMストリームを受信し，1本分のMPEG4+PCMストリームを送信している．これは，MF_{T2}が中継処理を行うためにrs_2で利用されているIPマルチキャストセッションに参加したため，rs_2へ流入する全てのトラフィックを受けていることを意味する．一方，MF_{T2}の中継したMPEG4+PCMストリームをMF_{U1}が受信していることは，図10の(S3)から確認できる．図10と図11に対し，図12と図13の入出力ビットレートのスケールが異なることに注意し，図10の(S3)以降の入力ビットレートを見てみると，若干(約1.5Mbps)増加している．また，受信したMPEG4+PCMストリームを再生表示するためにCPU利用率も増加している．

シナリオ(S4)ではMF_{U6}がrs_2へ参加する．しかしながら，既にrs_2にはMF_{U5}が参加しており，rs_2に対してストリームの拡張が行われているので，各図とも(S4)の時点では変化が見られない．続くシナリオ(S5)の結果，MF_{U6}はMPEG4(360×240)+PCMストリームの送信を開始し，rs_1へストリームを拡張する．中継ノードにはMF_{T2}が選択され，その結果は各図の(S5)より確認できる．

次にMF_{U6}は，シナリオ(S6)で退出する．これにより，MF_{U6}が送信していたMPEG4+PCMストリームが削除される．そして，シナリオ(S7)でMF_{U5}が退出する．ここで，

rs_2 の参加者数は 0 となり，MF_{U1} から MF_{U4} はそれぞれ rs_2 へ拡張していたストリームを縮退させる．ストリームが縮退した結果，MF_{T1} におけるトランスコーディング処理は終了する．これは図 11 の(S7)より確認できる．

最後に，シナリオ(S8)で MF_{U1}～MF_{U4} が rs_1 から退出し，相互通信セッションも終了する．

今回の評価実験に用いた MidField セッションの構成では，RTP セッションが 2 つであり，利用可能なフォーマットも限定されたものである．しかしながら，トランスコーディングエージェントを必要に応じて動的に配置することにより，利用者の通信環境や QoS 要求に応じて適切なフォーマットを用いた相互通信サービスの基盤となる機能が実現されたことを確認した．

5．まとめ

本稿では，利用者環境に応じて，適切なフォーマットによる相互通信サービスを実現するためのミドルウェアを提案した．提案システムは，音声やビデオストリームを用いる相互通信アプリケーションに対して，通信機能と通信資源の管理を含む相互通信セッションを提供する．また，柔軟な相互通信セッションを実現するために，移動エージェントベースのトランスコーディング機能を利用し，トランスコーディングエージェントが複数のマルチキャストセッションを動的に接続する仕組みを設計した．そして，プロトタイプシステムを構築し，通信実験を通して，利用者環境を考慮した相互通信サービスの基盤となる機能が実現されたことを確認した．

今後，相互通信拠点数のスケーラビリティを上げるため，複数のストリームを 1 本のストリームへ統合するミキサーの機能を導入し，動的に配置する仕組みを考案する．一方，提案システムは IP マルチキャストの利用を前提としているが，ミキサーの機能と併せて，マルチキャストとユニキャストの変換機能の実現も今後の課題とする．

参考文献

[1] Rosenberg, J., Schulzrinne, H., Camarillo, G., Johnston, A., Peterson, J., Sparks, R., Handley, M., Schooler, E. : SIP: Session Initiation Protocol, RFC3261 (2002).

[2] Schulzrinne, H., Casner, S., Frederick, R., Jacobson, V. : RTP: A Transport Protocol for Real-Time Applications, RFC3550 (2003).

[3] Ooi, W.T. and Renesse, V.R. : Distributing Media Transformation Over Multiple Media Gateways, Proc. 9th ACM International Multimedia Conference (2001).

[4] Gharai, L., Perkins, C., Riley, R. and Mankin, A. : Large Scale Video Conferencing: A Digital Amphitheater, Proc. 8th International Conference on Distributed Multimedia Systema (2002).

[5] Hashimoto, K. and Shibata, Y. : Dynamic Transcoding Functions by Extended Media Stream, Proc. 18th International Conference on Advanced Information Networking and Applications, Vol. 1, pp. 334-339 (2004).

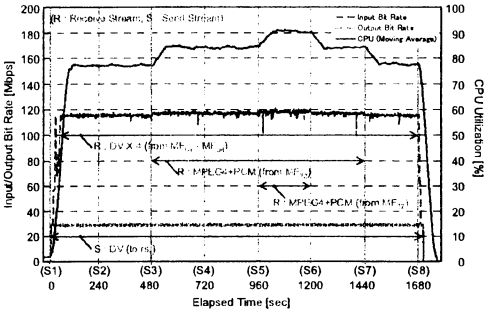


図１０：CPU 利用率と入出力ビットレート(MF_{U1})

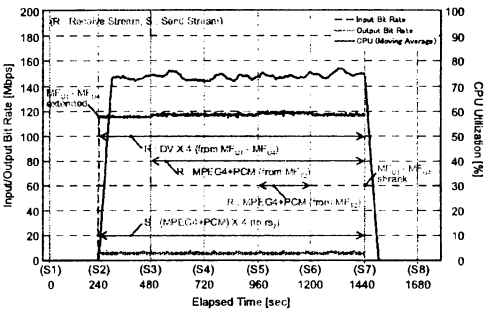


図１１：CPU 利用率と入出力ビットレート(MF_{T1})

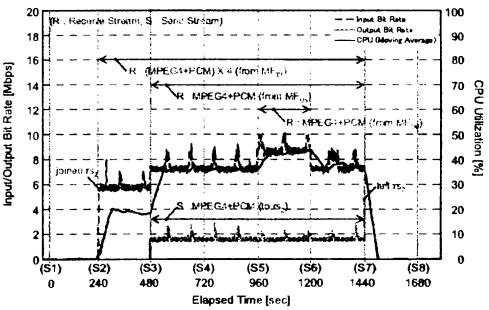


図１２：CPU 利用率と入出力ビットレート(MF_{U5})

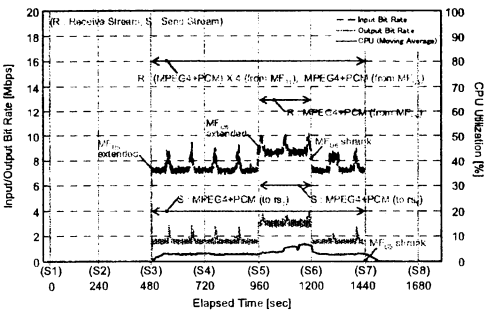


図１３：CPU 利用率と入出力ビットレート(MF_{T2})