音楽情報科学 21－3
（1997. 7. 20）

RMCP: Remote Media Control Protocol

— 時間管理機能の拡張と遅延を考慮した遠隔地間の合奏 —

後藤 真孝　　根山 亮　　菊地 淑晃　　村岡 洋一

早稲田大学 理工学部

{goto, ryo, kiku, muraoka}@muraoka.info.waseda.ac.jp

あらまし　本稿では，シンボル化された音楽情報をネットワークを介して共有するための通信プロトコル RMCP について述べる．従来の関連プロトコルの多くはコネクション型であり，複数プロセス間での効率的な情報共有は十分考慮されていなかった．RMCP はコネクションレス型であり，各プロセスへ個別に送信するオーバヘッドがないブロードキャストを用いて通信するため，情報共有の効率が良い．本研究では従来の RMCP で未対応だった，リアルタイム音楽情報処理のためのタイムスタンプを用いた時間管理と，遠隔地間の合奏のための信頼性を確保した双方向パケット中継を実現する．また，遅延を考慮した新たな形態の遠隔地間の合奏も提案する．RMCP は既に様々なアプリケーションに応用され，有効性が確認されている．

RMCP: Remote Media Control Protocol

— Time Scheduling Extension and Remote Session with Delay —

Masataka Goto　Ryo Neyama　Yoshiaki Kikuchi　Yoichi Muraoka

School of Science and Engineering, Waseda University

3-4-1 Ohkubo Shinjuku-ku, Tokyo 169, JAPAN.

Abstract　This paper describes a communication protocol, called RMCP (Remote Media Control Protocol), which is designed for sharing symbolic musical information through computer networks. Most previous related protocols were connection-oriented and did not emphasize efficient information sharing among multiple processes. Since the RMCP is a connection-less protocol, it supports broadcast-based information sharing without the overhead of multiple transmission. It also supports time scheduling using time stamps for real-time musical information processing and reliable bidirectional packet relay for remote sessions. This paper also proposes an innovative remote session over the Internet that has a long delay. RMCP has been utilized in various applications and we found that RMCP facilitated system implementation and expansion.

1　はじめに

MIDI[1]情報などのシンボル化された音楽情報を通信するためのネットワークプロトコルは，音楽情報処理システムを分散実装するためだけでなく，ネットワークを活用した新たな音楽アプリケーションを実現するためにも重要になってきている．例えば，LAN(local area network) や WAN(wide area network) などの計算機ネットワーク上で MIDI 情報を通信することで，ネットワーク経由のライブ MIDI 中継や遠隔地間の合奏，ネットワーク上の多様な計算機によって支援(演奏に関連した情報のコンピュータグラフィックスによる視覚化など) された合奏，人間とネットワーク上に分散した複数の計算機間の即興演奏といった様々なアプリケーションが可能になる．さらに，音楽アプリケーションを分散実装することは，負荷分散以外にも，異なる計算機に接続されている多様な機器を同時に活用できるといった利点を持つ．これらのアプリケーションのためには，ネットワークを経由した効率の良い音楽情報の共有が重要となる．

従来の MIDI に基づいたネットワークプロトコル[2)~6)]の多くはコネクション型(connection-oriented) であり，複数の分散したプロセス間で情報共有する際の遅延時間を小さく抑えることは，十分考慮されていなかった．また，MIDI だけを用いた場合，同じ情報を共有するために MIDI 機器間で全対全通信を実

現するのは効率が悪かった．さらに，MIDIは通信のバンド幅が低く制限されており，局所的な通信のために設計されていた．MIDIに基づかない演奏情報の通信プロトコル[7]も提案されているが，その多くは特殊な機器の使用を前提としていた．

我々は，計算機ネットワークを介してシンボル化された音楽情報を共有するために，音楽用通信プロトコルRMCP (Remote Media Control Protocol)☆1を設計し提案してきた[8)~10)]．RMCPは，Ethernet[11)]などの計算機ネットワークとMIDIを融合した分散協調システムにおける，コネクションレス型(connectionless)の通信プロトコルである．RMCPでは，再送(各プロセスへの転送)のオーバヘッドがないブロードキャストを利用して通信するため，複数プロセス間での情報共有の効率が良い．しかし，文献8)の時点では時間管理の機能は提供していなく，受信されたパケットはすべてその直後に処理されていた．そのため，演奏情報等を事前に通信しておくことはできなかった．また，十分高速で信頼性の高い通信回線を使用することを前提に，RMCPでは通信の信頼性は確保していなかった．そのため，パケットロスや大きな遅延が発生するInternetのようなWANで運用するのは，実用上は難しかった．

そこで本研究ではこれらを解決するために，タイムスタンプを用いた時間管理と，遠隔地間の信頼性を確保した双方向パケット中継を実現する．前者により，タイムスタンプの時刻以前に到着したパケットは，伝送中の遅延の揺らぎ等の影響を受けずに，受信側で時間管理されて時刻通りに処理可能になる．また後者により，Ethernetなどの LANだけでなく，Internetのような WANでも運用可能になる．さらに，遅延を考慮した新たな形態の遠隔地間の合奏も実現できる．

以下，**2**においてRMCPがサーバ・クライアント・モデルに基づいていることを述べ，サーバ・クライアントの設計指針と基本的な機能を紹介する．そして，**3**においてRMCPの具体的な実装を説明し，時間管理の実現方法と信頼性を確保した遅延付きパケット中継の方法を提案する．我々はRMCPプログラミングライブラリをC言語上とJava言語上に実装し，既に様々なOS上でRMCPを運用してきた．そこで**4**では具体的な運用例として，音楽に踊らされる仮想のダンサー，仮想ジャズセッションシステム，遠隔地間の合奏などのRMCPを用いた様々なアプリケーションを紹介する．最後に**5**でまとめを述べる．

☆1 文献8)の時点ではRemote Music Control Protocolであったが，その後コンピュータグラフィックス等の他のメディアにも対応できるように拡張したため，現在はRemote Media Control Protocolと呼んでいる．

2 RMCPの概要

RMCPでは，様々な音楽情報はRMCPパケットという単位で伝送され，分散した計算機上の複数のプロセスによって共有される．例えば，MIDI note on/offなどの各MIDIメッセージは，一つのRMCPパケットの中に包まれた後に，LANを経由して他のプロセスへと送られる．

RMCPはマルチサーバ・マルチクライアント・モデルに基づいている．複数のサーバが様々なクライアントからの要求を受信して処理するモデルである．そのため，RMCPに関するすべての計算機プロセスは，**RMCPクライアント**と**RMCPサーバ**に分類される．

各RMCPクライアントは，MIDI機器(MIDI IN)から得たMIDIメッセージや，ユーザ(演奏者等)のインタラクション内容に対応するメッセージなどを含むRMCPパケットを生成する．そして，これをすべてのRMCPサーバへ向けてブロードキャストする．このブロードキャストはコネクションレス型の片方向の送信であるため，サーバからクライアントへの返答(受取通知パケット)はない．

一方，各RMCPサーバは，ブロードキャストされたすべてのパケットを受信し，様々な方法で利用する．例えば，MIDI機器(MIDI OUT)を制御して音を鳴らしたり，音楽情報を視覚化したり，音楽に反応するコンピュータグラフィックスを生成したりする．このように，クライアントから一度ブロードキャストされたパケットの内容は，すべてのサーバが共有して同時に活用できるため，複数回送信するオーバヘッドがない．さらに，通信量を増やすことなく，演奏の視覚化などの多様なサーバを追加することもできる．

2.1 設計指針

RMCPサーバ・クライアントを設計する際の指針を述べる．RMCPでは，実現したい機能を，再利用が可能なようにできるだけ小さなプロセスに分けて実装する．これにより，各サーバ・クライアントはそれぞれに特化した処理に専念すればよいため実装が容易になり，サーバを単に追加するだけで新たな機能を実現できるので拡張性が高くなる．さらに，効果的に負荷分散できるように，各サーバ・クライアントを異なる計算機上に割り当てることが容易になる．

2.2 基本的なサーバ・クライアント

2.1の指針に基づいて設計された，基本的なRMCPサーバとRMCPクライアントの一覧を表1に示し，これらの運用例を図1に示す．図中には各サーバ・ク

表 1: 基本的な RMCP サーバと RMCP クライアント

RMCP サーバ	機能
RMCP Sound Server	受信パケット中の MIDI メッセージを MIDI 機器へ送信 (主に聴覚化)
RMCP Display Server	受信パケット中の MIDI メッセージを画面上の鍵盤の色を変えて視覚化
RMCP Animation Server	受信パケットを用いて音楽に反応するコンピュータグラフィックスを生成
RMCP Recorder	全受信パケットを受信時刻と共に「RMCP パケット記録ファイル」へ記録
RMCP クライアント	**機能**
RMCP MIDI Receiver	MIDI 楽器を用いて演奏 (MIDI メッセージを MIDI 機器から受信)
RMCP MIDI Station	MIDI 楽器の代わりに計算機のキーボードとマウスを用いて演奏
RMCP SMF Player	標準 MIDI ファイル (SMF) を再生
RMCP Player	「RMCP パケット記録ファイル」を再生

ライアントを一つずつしか示していないが，実際にはそれぞれが複数あってもよい．

ユーザは，MIDI 楽器と RMCP MIDI Receiver か，計算機のキーボードと RMCP MIDI Station を用いて演奏する．演奏手段として MIDI 楽器を用いる場合には，MIDI 楽器から送られてくる MIDI メッセージを RMCP MIDI Receiver が受信し，これを RMCP パケットとしてブロードキャストする．一方，演奏手段として計算機のキーボードとマウスを用いる場合には，RMCP MIDI Station が MIDI メッセージを生成し，RMCP パケットとしてブロードキャストする．この場合，ユーザは各キーが鍵盤に割り当てられたキーボードを押すか，画面上の鍵盤をマウスでクリックして演奏する．また，RMCP SMF Player により，標準 MIDI ファイル (SMF: Standard MIDI File) を再生することもできる．

こうしてブロードキャストされたパケットが RMCP Sound Server に受信されると，パケットに含まれる MIDI メッセージが MIDI 楽器へ送られて，実際の演奏音として出力される．これらのパケットは RMCP Display Server にも受信され，画面上の鍵盤の色を変えることで視覚化される．さらに RMCP Animation Server にも同時に受信され，その内容に反応して動作する仮想のダンサーや演奏者などのリアルタイムコンピュータグラフィックスが生成される．

RMCP Recorder と RMCP Player は，ネットワーク上の RMCP パケットの状況を記録して再現するために用いる．RMCP Recorder は全 RMCP パケットを，その受信時刻と共に「RMCP パケット記録ファイル」へ記録する．一方，RMCP Player は，そのファイル中のパケットを，受信時刻通りの適切なタイミングでブロードキャストする．これらは，RMCP に基づいた音楽情報処理システムにおいて，ユーザの演奏やインタラクションの状況，システムの出力等を保存したいときに便利である．

3 RMCP の実装

我々は，通信プロトコル RMCP を設計し，それに基づいて RMCP プログラミングライブラリと様々なサーバ・クライアントを実装した．RMCP は，下位レイヤーとして UDP/IP を用いたコネクションレス型のプロトコルである．Ethernet LAN などの十分高速で信頼性の高いネットワークで運用されることを前提として，RMCP レイヤーでは基本的にブロードキャストで通信し，その信頼性は確保しない．これは，情報共有する際の遅延時間を小さく抑えるためである．ただし **3.3** で述べるように，Internet のような WAN で運用する際には，そもそも遅延時間が大きいので，信頼性を確保した通信を提供する．

3.1 RMCP パケット

RMCP レイヤーのパケットを RMCP パケットと呼び，これは RMCP クライアントから RMCP サーバ

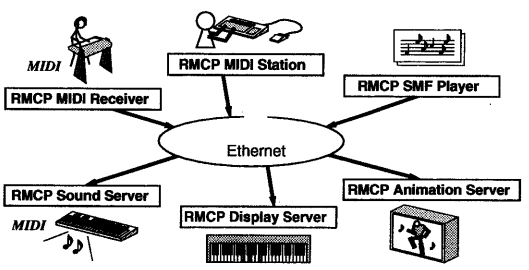

図 1: RMCP サーバと RMCP クライアントの運用例

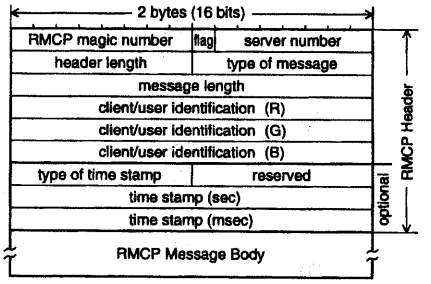

図 2: RMCP パケットの構造

へ片方向に送信(ブロードキャスト)される．RMCPパケットはヘッダーとそれに続くメッセージ本体から成る(図2)．ヘッダーには，メッセージの種類，サーバ番号，クライアント識別子(ユーザ識別子)，タイムスタンプ情報，中継禁止フラグ(flag)，ヘッダー長，メッセージ長などが含まれている．

主要なメッセージの種類には，以下のものがある．

- MIDI情報
  MIDIメッセージを伝送するために用いる．メッセージ本体がそのままMIDIメッセージ(複数のMIDIメッセージでもよい)となっている．
- ビート情報
  テンポの同期のためにビートの時刻を伝送する際に用いる．
- コード情報
  コード名やボイシングなどを伝送するために用いる．曲の調の指定も伝送することができる．
- アニメーション情報
  コンピュータグラフィックスを生成・制御するために用いる．
- ジェスチャー情報
  ジェスチャー認識の結果などを伝送するために用いる．

他にもサーバを終了させるメッセージや，3.2で述べる時刻同期のためのメッセージ，メッセージ本体が規定されていない拡張用メッセージなどがある．

サーバ番号はサーバを区別するための7ビットの識別子である．RMCPサーバは，起動時に自分のサーバ番号を設定でき，同じサーバ番号を持つもの同士でグループを作ることができる．RMCPクライアントは，ブロードキャストするときにどのサーバ番号(グループ)に対する要求かを指定する．このとき，全サーバという指定もできる．複数の音楽アプリケーションが同一ネットワーク上でRMCPを利用するときに，それぞれがサーバ番号を変えて運用すれば，お互いに独立に実行できる．

クライアント識別子は，RGBそれぞれ2バイトづつの計6バイトのカラーコードである．特にRMCP Display ServerがMIDI情報を画面表示する際に，このカラーコードを用いて鍵盤の色を変えることで，その演奏がどのクライアントによるものかをユーザが判断できる．また，これはユーザ識別子として用いることも可能であり，各ユーザが自分固有のカラーコードを定義すれば，複数のユーザが合奏しているときなどにユーザ間の区別ができる．

タイムスタンプ情報については3.2において，中継禁止フラグについては3.3において説明する．ヘッダー長，メッセージ長は，それぞれの長さをバイト単位で示したものである．

3.2 時間管理

RMCPサーバがクライアントの指定する時刻にパケットを処理できるように，RMCPではタイムスタンプ☆2を用いた時間管理機能を提供する．RMCPパケットには，それが処理されるべき時刻を示すタイムスタンプを付与されたパケットと，タイムスタンプなしパケットの二種類がある．これによりRMCPサーバは，タイムスタンプの時刻前に到着した前者のパケットを，その時刻が来るまでバッファリングして時刻通りに処理することができ，それらの時間順も保証できるようになった．タイムスタンプの時刻後に到着したパケットやタイムスタンプなしパケットは，基本的にサーバに到着した直後にその場で処理される．

例えば，RMCP SMF PlayerがSMFを再生する際には，SMF中のMIDIメッセージをタイムスタンプ付きパケットとして，数秒前にブロードキャストしておく．これにより，仮にネットワークの遅延時間が揺らいでも，RMCP Sound Serverによる演奏音は影響を受けずに適切なタイミングとなる．また，事前に演奏情報がわかることで，RMCP Animation Serverによる仮想の演奏者が，「予備動作をおこなってから実際の発音タイミングで楽器を弾く」といった動作をすることも可能になる．

3.2.1 時刻同期サーバ

タイムスタンプによる時間管理は，複数の計算機の間で時刻が同期していることを前提としている．同じタイムスタンプのパケットは，異なる計算機上のRMCPサーバによって，同一時刻に処理される必要があるからである．

そこで，同期を必要とする各計算機上に一つのRMCP Time Synchronization Server(時刻同期サーバ)を導入する．時刻同期サーバによって，あたかも異なる計算機の内部時計が同期しているかのように，RMCPサーバが受信パケットのタイムスタンプを扱うことができる☆3．この時刻同期に関する処理はすべてプログラミングライブラリ内に隠蔽され，RMCPの上位レイヤーからは既に同期した時刻上のタイムスタンプしか見えないため，RMCPサーバの開発者は時刻同期を意識する必要がない．

各時刻同期サーバは，自分の計算機の内部時計と他

☆2 現在の実装では，1 ms単位の絶対時刻表現のみに対応している．

☆3 時刻同期サーバを用いる代わりに，NTP (Network Time Protocol)[12]等の他の方法によって，事前に計算機間で内部時刻を合わせておいてもよい．ただし，一般に内部時計の補正は管理者特権が必要である．一方，我々が提案する手法は内部時計を直接補正するわけではないので，管理者特権は必要ない．

のそれぞれの計算機の内部時計との時刻差の表(オフセットテーブル)を作成する．そして，このオフセットテーブルに送信時刻(現在の内部時計の時刻)のタイムスタンプを付けて，定期的にブロードキャストする☆4．一方，時刻同期サーバが他の計算機上の時刻同期サーバからこのテーブルを受信すると，自分が所持するテーブル中の関連する時刻差の項目を更新する．具体的には，自分の内部時計で計測したテーブルの受信時刻と，そのテーブルのタイムスタンプに書かれた送信時刻との差を計算し，テーブル送信者に対応する項目を更新する．ただし，LANの遅延時間は十分小さく無視できると仮定している．

各RMCPサーバは，同じ計算機上の時刻同期サーバからオフセットテーブルを受信する．そして，タイムスタンプ付きのパケットを受信する度に，その送信者の時刻差の項目をテーブル中から見つけ，その時刻差を加えてタイムスタンプを補正する．こうすることで，実際には内部時計がずれているにも関わらず，見かけ上はすべてが同期しているかのようにタイムスタンプを扱うことができる．また，RMCPクライアントの側は時刻差について知る必要がないため，実装が簡潔になる利点も持つ．

3.2.2 時刻同期サーバの運用例

時刻同期サーバの運用例を図3に示す．ある時点における3台の計算機Host A, B, Cの内部時計が，右下に示された1000, 1050, 980 (時間単位は省略する)であり，時刻同期サーバによって図中に示したオフセットテーブルが作成されたとする．Host AのRMCPクライアントが，時刻1200のタイムスタンプを付けたパケットをブロードキャストすると，各計算機上のRMCPサーバは，オフセットテーブルからHost AのIPアドレスを探し，タイムスタンプに時刻差を加えて補正する．その後，Host Bがこの時刻から100後の時刻1350のタイムスタンプを付けたパケットをブロードキャストすると，同様に他の計算機上で適切に補正される．

実際にHost Aの時刻が1200になったときには，すべての計算機の時刻が一つ目のパケットのタイムスタンプの時刻と等しくなり，同時に処理される．それから100後にも同様に二つ目が同時に処理される．このように，内部時計があたかも同期しているかのように，各RMCPサーバはタイムスタンプを処理できる．

☆4 ただし，RMCPパケットのトラフィックが多いときはブロードキャストしない．つまり極力アプリケーションが利用されていないときに時刻差を測定する．これは，アプリケーション動作中に余計なトラフィックが発生するのを防ぐだけでなく，より適切な時刻差を得ることが期待できるという効果もある．

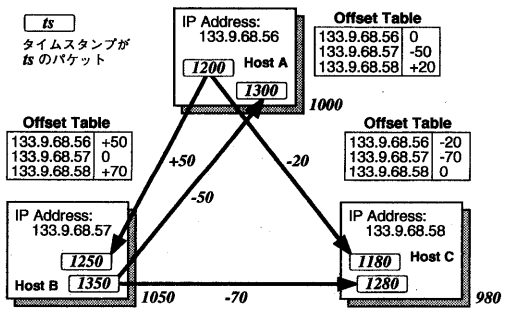

図3: オフセットテーブルによるタイムスタンプの補正

3.3 信頼性を確保したパケット中継

RMCPをInternetなどのWAN上で運用するために，WANで結ばれた二つのLAN間で双方向にパケットを中継する(共有する)RMCP Gatewayを導入する．RMCP Gatewayは，下位レイヤーとしてTCP/IPを用いたコネクション型の通信をおこない，信頼性を確保してパケットを中継する．

遠隔地にある二つのLANを結ぶときに，まず双方でRMCP Gatewayを実行してコネクションを確立する☆5(図4)．各RMCP Gatewayは，自分のLAN上にブロードキャストされたRMCPパケットを，他方のRMCP Gatewayへ送信する．逆に，他方から中継されたパケットを受信すると，自分のLAN上にブロードキャストする．ただし，一度中継したパケットは再び中継されることがないように，ヘッダー中の中継禁止フラグをセットしておく．これにより中継の無限ループを回避する．

RMCPサーバ・クライアントは，RMCP Gatewayによって結ばれた二つのLANが，あたかも同一のネットワークであるかのように通信できる．遠隔地間用に新たに開発し直す必要はない．ただし，ユーザはWAN経由の通信の遅延時間が，LAN上の通信よりも長いことを考慮する必要がある．

遠隔地間で通信する際には，ネットワークの遅延時間を回避することは不可能である．光速でさえ地球を半周するのに約66 msもかかってしまう．したがってRMCP Gatewayによる中継では，遅延時間を小さくするよりも，変動(ジッタ)が非常に小さい一定の遅延時間を提供する方が重要だと我々は考える．そこで，ネットワークの遅延時間よりも十分に大きい遅延時間 D を中継用に設定し，受信側でパケットをバッファリングすることで，ネットワークに起因する変動の影響を受けないようにする．

☆5 複数のLAN間でRMCPパケットを共有する場合，各LAN間でRMCP Gatewayによるコネクションを確立すればよい．

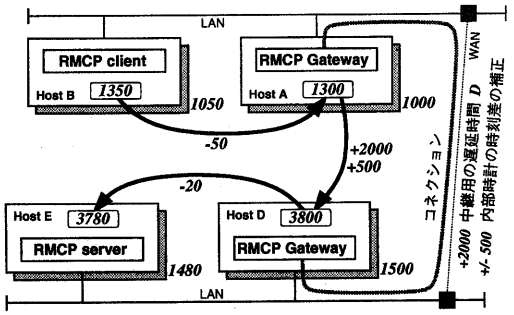

図 4: RMCP Gateway による遅延付きパケット中継

RMCP Gateway は，LAN から受信したパケットを中継する際に，パケット中のタイムスタンプとは別に，その受信時刻もタイムスタンプとして付加する．他方の RMCP Gateway は，両者のタイムスタンプに D を加えた上に，両端の計算機の内部時計の時刻差も補正する (図 4)．そして，それを送信すべき時刻が来るまでバッファリングして，時刻通りにブロードキャストする．適切な D を設定し，内部時計の時刻差を補正するために，RMCP Gateway はコネクション確立直後に，ネットワークの遅延時間と内部時計の時刻差をパケットを往復させて計測する．計測方法は NTP[12] の手法に基づいている．

以上の遅延付き双方向パケット中継により，Internet を介したライブ MIDI 中継だけでなく，4.5 で述べる新たな形態の遠隔地間の合奏 RemoteGIG も可能になる．

3.4 実験結果

RMCP プログラミングライブラリを C 言語上と Java 言語上☆6にそれぞれ実装し，IRIX-5.3, IRIX-6.2, Solaris-2.5, SunOS-4.1.3, HP-UX, Linux-2.0, Windows-95, Windows-NT などの様々な OS 上で RMCP を運用した☆7．現在の実装では，IRIX OS 上の RMCP サーバは 2 ms，他の OS 上の RMCP サーバは 10 ms の分解能でタイムスタンプを処理する．

我々は，実際に RMCP に基づいた様々な分散音楽情報処理システムを実現してきた．その経験から，RMCP を用いることで必要な機能が再利用できて実装が容易になり，拡張性も高くなることを確認した．特に，リアルタイムシステムやインタラクティブシステムの実現に効果的であった．

LAN 上の RMCP パケットの遅延時間の測定結果を示して考察する．異なる計算機上にあるクライアントからサーバへ RMCP パケットが到達するまでの遅延時間を，MIDI note on メッセージを一万回伝送して測定した．実験には C 言語上の RMCP ライブラリを用い，10 Mbps Ethernet 上の 2 台の SGI Indigo2 (IRIX-5.3) 間で測定した．ただし，測定時の LAN と計算機の負荷は小さかった．その結果，遅延時間は平均値 0.30 ms，標準偏差 0.06 ms，最小値 0.28 ms，最大値 1.24 ms であった．これから，RMCP は速度的に MIDI (31.25 Kbits/sec) に遜色なく情報伝送できることがわかった☆8．また，パケットロスや到着順の入れ換え等の問題も発生しなかった．

4 アプリケーション

1992 年以来，RMCP に基づいた様々なアプリケーションが実現されてきた．以下，これらの中から主要なものを紹介する．なお，表 1 以外の RMCP サーバ・クライアントは，基本的にアプリケーション固有のものである．

4.1 仮想のダンサー “Cindy”

我々は RMCP を応用して，音楽に踊らされる CG (コンピュータグラフィックス) ダンサーによるインタラクティブパフォーマンスシステムを実現した[14),15)]．本システムでは，二人の演奏者が自分たちの即興演奏する音楽によって，仮想のダンサー “Cindy”☆9の踊りをリアルタイムに振付けできる (図 5)．二人は通常のジャムセッションのように即興演奏するだけでなく，異なる役割で振付けを分担しながら，協調してダンサーを踊らせようとする．つまり，音楽的にも映像的にも協調して演奏しようとする．こうして聴覚と視覚によるマルチモーダルインタラクションを達成することで，CG ダンサーは単に音楽に付随したものでなく，二人の演奏者にとっての新たな表現手段となる．

本システムは，Cindy を表示する RMCP Animation Server，演奏を踊りに反映させるために分析する RMCP Music Analyzer，RMCP MIDI Receiver，RMCP Sound Server で構成される．

4.2 VirJa Session

VirJa Session (Virtual Jazz Session System) とは，人間の演奏者と計算機演奏者が，お互いの演奏を聞

☆6 Java アプレット上で安全性を確保ながら RMCP を利用する方法も提案されている[13)]．

☆7 OS によっては，複数のプロセスが同一 UDP ポートからパケットを受信できないため，パケット分配用プロセスを用いる必要がある．

☆8 MIDI で一音分の演奏情報 (3 bytes) を伝送するのに 0.96 ms かかり，MIDI 楽器内部では 10 ms 以上も遅延する場合があることを考慮すると，RMCP の遅延時間は十分小さいと言える．

☆9 ここで紹介したものとは別の応用例になるが，Cindy は受動的に踊らされるだけでなく，ビートトラッキングシステム[16),17)]により音楽に合わせて踊ることもできる．

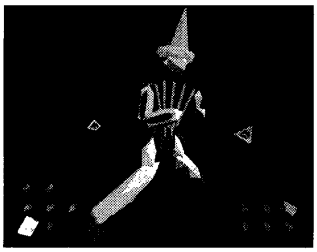

図 5: 仮想のダンサー "Cindy"

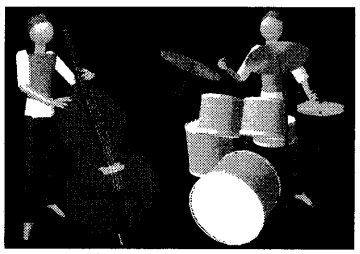

図 6: VirJa Session

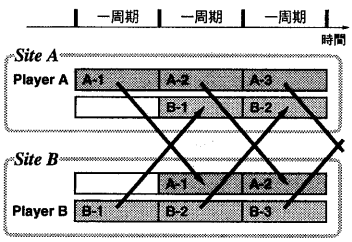

図 7: RemoteGIG

き合いながら対等な立場で即興演奏するだけでなく，お互いの姿やジェスチャーも見ながらインタラクションできるシステムである[18),19)]．現実のセッションにより近い全演奏者間のマルチモーダルインタラクションを実現したことで，従来の音だけのセッションシステムに比べ，より臨場感のあるセッションが達成できた．

現在の実装では，ジャズのピアノトリオを対象とし，人間がピアニスト，計算機がベーシストとドラマーを担当する (図 6)．本システムは，RMCP MIDI Receiver，RMCP Sound Server，カメラ画像からジェスチャーを認識する RMCP Camera Analyzer，各計算機演奏者の演奏を担当する RMCP Player Server，その姿を表示する RMCP Animation Server，テンポを管理する RMCP Beat Provider から成る．

4.3 ネットワークセッション

RMCP は設計当初，複数の演奏者が LAN を介して合奏するために用いられた[8),9)]．各演奏者は，単に他の演奏者の音を聞けるだけでなく，より協調して演奏しやすいように，視覚化された他の演奏者の演奏状態も見ることができる．

本アプリケーションでは，RMCP MIDI Receiver あるいは RMCP MIDI Station，RMCP Sound Server，RMCP Display Server を利用する．RMCP SMF Player による伴奏に合わせて合奏してもよい．また，合奏を録音・再生するために RMCP Recorder と RMCP Player を用いることもできる．

4.4 Improvision

Improvision とは，鍵盤演奏ができない人でも計算機のマウスをクリック・ドラッグするだけで即興演奏できる，音楽演奏用インタフェースである[8)]．ただし，無調性でリズムのない音楽を対象としている．各演奏者は，RMCP MIDI Station の代わりに RMCP Improvision を用いて合奏でき，RMCP Display Server の代わりに RMCP Improvision Display Server を用いて，他の演奏者の演奏状態を見ることができる．

4.5 RemoteGIG

Internet のような遅延の大きい WAN を使ってリアルタイムに合奏するのは困難なため，従来は演奏を片方向へ送信するライブ MIDI 中継が主流であった．遠隔地間の合奏に関しては，WAN(ISDN) を用いた実施例[20)]も報告されているが，**3.3** で述べたように遠隔地間では遅延時間の影響は不可避である．そこで我々は，従来の一箇所で合奏をするモデルを遠隔地間に持ち込むのではなく，遅延を前提とした遠隔地間の合奏用のモデルを提案する．

RemoteGIG とは，Internet 等で不可避な遅延を積極的に利用した，新たな形態の遠隔地間の合奏である．RemoteGIG では，同一のコード進行 (12 小節のブルース進行など) の繰り返しを，テンポ一定で演奏することを前提とする．遠隔地にいる二人の演奏者が合奏する様子を図 7 に示す．演奏者はお互いの演奏を，コード進行のちょうど一周期分の時間だけ遅れて聞き合うことで合奏する．コード進行は繰り返すため，演奏が一周期分遅れれば再び同じコードとなり調和することが期待できる．このとき，演奏者は一箇所で合奏するときと同様なインタラクションを試みるのではなく，新たなインタラクションを探求するとよい．なお，この合奏のモデルは無調性音楽等にも適用できる．

RemoteGIG では，**4.3** の RMCP サーバ・クライアントに加え，一組の RMCP Gateway を必要とする．その際，**3.3** の D を一周期分の時間の倍数に設定する．特に，一定のテンポを維持するために，RMCP SMF Player で伴奏用のドラムスの演奏を流すとよい．

4.6 その他

CG 上の仮想犬と，音楽も含めた 3 種類のインタラクションができるシステム[21)]を実現する際にも，RMCP が用いられた．また，仮想空間中の仮想キャラクタの動きを複数の仮想カメラで同時に撮影するように CG 画像が生成できる，分散 CG アニメーションシステム VirStA System (Virtual Stage and Actors System)[22)~24)]の実装においても，RMCP は有

効であった．これらのアプリケーションにおいては，RMCPが音楽情報だけでなくアニメーション情報やジェスチャー情報も扱えることが不可欠であった．

さらに，ジャズらしいコードにリハーモナイズするシステム「ハービー君」[25]や，ネットワーク経由で相互作用するアルゴリズム作曲システム[26]の実装にもRMCPが用いられた．

5 おわりに

本稿では，複数の分散したプロセス間で，MIDIのようなシンボル化された音楽情報を共有するためのネットワークプロトコルRMCPについて述べた．RMCPでは，タイムスタンプを用いた時間管理とそのための時刻同期の機能により，音楽やCGに関連した情報をリアルタイムに扱うことを可能にした．また，運用するネットワークの性質・信頼性に応じて下位レイヤーを選択して通信することで，分散音楽情報処理に適した効率が良く実用的な情報共有を実現した．具体的には，LAN内の通信にはUDP/IPを用いてブロードキャストによるオーバヘッドの小さい情報共有を実現する一方，Internet経由などの遠隔地間の通信にはTCP/IPを用いることで，信頼性を確保した情報共有を実現した．RMCPは既に複数のOS上に実装され，様々な目的のために運用されている．

我々は，RMCPソフトウェアパッケージをWWWページ (http://www.info.waseda.ac.jp/muraoka/members/goto/RMCP/) 上で公開する予定である．また，Javaアプレット上でRMCPとMIDIを利用するためのAPIも提供していく予定である．

謝辞

本プロジェクトをはじめるきっかけを与えて頂いたSALA (Science Art Laboratory) に感謝する．特に研究の初期の段階において多大な御協力・御指導を頂いた，元SALAメンバーの，橋本 裕司 氏，千葉 滋 氏，宮川 晋 氏に感謝する．また，本研究に対し有益な御助言・御協力を頂いた，日高 伊佐夫 氏，阿部 哲也 氏，松本 英明 氏，黒田 洋介 氏，鷲坂 光一 氏，平田 圭二 氏，長嶋 洋一 氏に感謝する．

参考文献

[1] MIDI規格協議会規格検討委員会: MIDI 1.0 規格 (Document Ver.4.1 日本語版), MIDI規格協議会 (1989).

[2] Aoyagi, T. and Hirata, K.: Music Server System — Distributed Music System on Local Area Network —, *Journal of Information Processing*, Vol. 15, No. 1, pp. 1–9 (1992).

[3] 森光彦: 音楽情報処理システムの構築とネットワーク技術の活用, 日本音響学会音楽音響研究会 MA92-3, Vol. 11, No. 1, pp. 21–26 (1992).

[4] Nielsen, O.: MIDI and Audio via ISDN, *Proc. of the 1994 ICMC*, pp. 451–454 (1994).

[5] Fober, D.: Real-time Midi data flow on Ethernet and the software architecture of MidiShare, *Proc. of the 1994 ICMC*, pp. 447–450 (1994).

[6] Fober, D., Letz, S. and Orlarey, Y.: Recent Developments of MidiShare, *Proc. of the 1996 ICMC*, pp. 40–42 (1996).

[7] McMillen, K., Simon, D. and Wright, M.: A Summary of the ZIPI Network, *Computer Music Journal*, Vol. 18, No. 4, pp. 74–80 (1994).

[8] 後藤真孝, 橋本裕司: MIDI制御のための分散協調システム — 遠隔地間の合奏を目指して —, 情処研報 音楽情報科学 93-MUS-4-1, Vol. 93, No. 109, pp. 1–8 (1993).

[9] 後藤真孝: MIDI制御のための分散協調システム, jus 設立10周年記念UNIX国際シンポジウム 論文集, pp. 161–171 (1992).

[10] 後藤真孝: 分散協調インタラクティブシステム — 音とCGによる遠隔地間のインタラクション —, NICOGRAPH'93 CG教育シンポジウム プロシーディングス, pp. 44–49 (1993).

[11] Metcalfe, R. M. and Boggs, D. R.: Ethernet: Distributed Packet Switching for Local Computer Networks, *Communications of the ACM*, Vol. 19, No. 7, pp. 395–404 (1976).

[12] Mills, D. L.: Network Time Protocol (Version 3) Specification, Implementation and Analysis, RFC-1305 (1992).

[13] 根山亮, 後藤真孝, 村岡洋一: WWW上の異なる計算機環境で動作する音楽セッションシステム — VirJa Session on WWWへ向けて —, 第54回情処全大 7J-05 (1997).

[14] Goto, M. and Muraoka, Y.: A Virtual Dancer "Cindy" — Interactive Performance of a Music-controlled CG Dancer —, *Proc. of the Lifelike Computer Characters '96*, p. 65 (1996).

[15] 後藤真孝, 村岡洋一: 音楽に踊らされるCGダンサーによるインタラクティブパフォーマンス, コンピュータソフトウェア, Vol. 14, No. 3, pp. 20–29 (1997).

[16] Goto, M. and Muraoka, Y.: A Beat Tracking System for Acoustic Signals of Music, *Proc. of the Second ACM Intl. Conf. on Multimedia*, pp. 365–372 (1994).

[17] Goto, M. and Muraoka, Y.: Beat Tracking based on Multiple-agent Architecture — A Real-time Beat Tracking System for Audio Signals —, *Proc. of the Second Intl. Conf. on Multiagent Systems*, pp. 103–110 (1996).

[18] 後藤真孝, 日高伊佐夫, 松本英明, 黒田洋介, 村岡洋一: すべてのプレーヤーが対等なジャズセッションシステム I. システムの全体構想と分散環境での実装, 情処研報 音楽情報科学 96-MUS-14-4, Vol. 96, No. 19, pp. 21–28 (1996).

[19] Goto, M., Hidaka, I., Matsumoto, H., Kuroda, Y. and Muraoka, Y.: A Jazz Session System for Interplay among All Players — VirJa Session —, *Proc. of the 1996 ICMC*, pp. 346–349 (1996).

[20] 高山明, 千葉雅之, 三木恵造, 首藤一彦: 遠隔同時演奏システム, 日本音響学会音楽音響研究会 MA89-10, Vol. 8, No. 3, pp. 23–28 (1989).

[21] 阿部哲也, 後藤真孝, 黒田洋介, 村岡洋一: インタラクティブドッグ～仮想犬との3種類のインタラクション～, インタラクティブシステムとソフトウェア III, 近代科学社, pp. 19–28 (1995).

[22] 後藤真孝, 阿部哲也, 松本英明, 村岡洋一: VirStA System: 仮想ステージと仮想アクターによる分散CGアニメーションシステム I. システムの全体構想, 第53回情処全大 1P-05 (1996).

[23] 阿部哲也, 後藤真孝, 松本英明, 村岡洋一: VirStA System: 仮想ステージと仮想アクターによる分散CGアニメーションシステム II. 分散環境でのリアルタイム実装, 第53回情処全大 1P-06 (1996).

[24] 松本英明, 後藤真孝, 阿部哲也, 村岡洋一: VirStA System: 仮想ステージと仮想アクターによる分散CGアニメーションシステム III. ジャズセッションプレーヤーの実現, 第53回情処全大 1P-07 (1996).

[25] 後藤真孝, 平田圭二: ハービー君: 演繹オブジェクト指向に基づいてジャズらしいコードにリハーモナイズするシステム, 情処研報 音楽情報科学 96-MUS-16-6, Vol. 96, No. 75, pp. 33–38 (1996).

[26] 長嶋洋一, 中村文隆, 後藤真孝, 片寄晴弘, 井口征士: ネットワーク上で相互作用するアルゴリズム作曲系を用いた音楽教育システム, 第54回情処全大 7J-04 (1997).