マルチキャストに基づく分散仮想環境管理プロトコル

江 崎 美 仁† 佐 藤 文 明††

近年のインターネットの普及によって，個人同士がネットワークを通じて直接インタラクションを行うようになってきた．現在そのような個人向けに開発されているサービスの中にはネットワークを介して複数のユーザが同一の三次元仮想空間を共有するシステムがある．これを分散仮想空間システムと呼ぶ．本論文では分散仮想空間システムにおける送信処理と再送処理に焦点を当てている．通信処理における負荷を分散させるために，マルチキャストを利用して送信処理の軽量化することと，動的な木構造によってメンバーリストを管理するプロトコルと，それに基づくメッセージ再送管理を提案する．

A Mangamement Protocol for Distributed Virtual Environment Based on a Multicasting

Yoshihito Esaki† and Fumiaki Sato††

By deployment of the Internet in recent years, individuals have come to perform a direct interaction through a network. In the service developed for such individuals, 3-dimensional virtual space systems shared by two or more users through network have been developed. This is called distributed virtual space system. In this paper, the focus is put to the transmission and retransmission in a distributed virtual space system. In order to balance the load for communication, this paper proposes the lightweight transmition process using a multicasting, and a protocol which manages the member list according to dynamic tree structure, and the message retransmission management based on the protocol.

1. はじめに

近年のインターネットの普及によって，個人同士がネットワークを通じて直接インタラクションを行うようになってきた．その動きに合わせるようにコンピュータネットワークを利用した様々なサービスが開発されてきている．そのようなサービスの中にはネットワークを介して複数のユーザが同一の三次元仮想空間を共有するシステムがある．例をあげると，バーチャルモール，3D チャットなどが存在する．これらのように仮想空間に入り，その中で様々なインタラクションを行うシステムは分散仮想空間（DVS）と呼ばれている．

分散仮想空間の各ホストは仮想空間データの複製を所有している．ユーザはその複製を使用することで仮想空間データに高速にアクセスできるが，メンバーそれぞれが持つ仮想空間データの一貫性を保つ必要がある．そのためにユーザは自分が保持している複製に対して行った操作内容を，他のメンバーに送信しなくてはならない．しかしこれはユーザ数が増加したときに通信処理が重くなったり，ネットワークに大きな負荷をかけるなどの問題を引き起こし，システム全体の応答性に支障をきたす．本研究では，過去の研究において実装された分散仮想環境システムの通信処理に焦点を当て，通信処理における負荷の集中を分散させるために，マルチキャストを利用した送信処理の軽量化と動的な木構造によるメンバーリストの管理プロトコルとそれに基づくメッセージ再送管理を提案する．

本論文の構成を以下にあげる．2 章では分散仮想環境システムについて説明し，我々の研究で実装されたシステムの動作を説明し，その問題点をあげる．3 章では分散仮想環境システムの問題点を解消しうる方式を提案し，その動作を説明する．4 章では，提案した方式について考察する．5 章は本論文のまとめである．

2. 分散仮想環境システム

2.1 分散仮想空間の概要

分散仮想空間（Distributed Virtual Space）とは

† 静岡大学理工学研究科
Graduate School of Science and Technology, Shizuoka University
†† 静岡大学情報学部
Faculty of Infomation, Shizuoka University

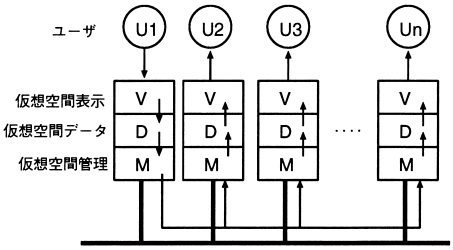


図 1 システムモデル
Fig. 1 System model.

ネットワークに接続された複数のホスト間で仮想的な三次元空間を共有し，各種のインタラクションを実現するシステムのことである．代表的な例としては，バーチャルモールシステム，3D チャットシステム，PAW などがあげられる．分散仮想空間では参加しているユーザは同一の仮想空間を共有する．そのため，同時に同じ仮想空間に参加しているユーザは，同じ世界を異なる視点から見ることができる．仮想空間の中では，参加しているユーザはアヴァタと呼ばれる CG の人形として表され，他のユーザの動作を見ることができる．また，空間内で起こったことや他のユーザが行った動作は，参加ユーザのそれぞれの視点から同じように見ることができる．

2.1.1 システムの動作

分散仮想環境システムの実際の動作を**図 1** に示す．ユーザ（Ui）はそれぞれ分散仮想環境システムのアプリケーションを持っている．アプリケーションは仮想空間データ（D），空間の描画モジュール（V），仮想空間の管理モジュール（M）で構成される．ユーザは仮想空間の管理モジュールを使って空間データを操作する．ユーザの行った操作は空間の更新情報としてシステムの参加ユーザに送信される．更新情報を受け取ったユーザはそのデータに基づいて自分が持つ仮想空間データを変更し，空間描画モジュールで画面に表示する．

2.1.2 システムのアーキテクチャ

分散仮想環境システムには，システムの目的やネットワーク環境に応じたいくつかのアーキテクチャが存在する．以下にアーキテクチャの例をあげる．

(1) **サーバレス方式**

専用のサーバを置かない方式で，アーキテクチャ上最もシンプルなアーキテクチャである．後述する DIVE が採用している方式で，一貫性制御のために各ホストは他のホストへの更新情報をブロードキャストする機能を持たせている．このアーキテクチャは各ホストやネットワークに高い性能を求めるもので，スケーラビリティにも問題がある．

(2) **機能集中型サーバ方式**

仮想空間のデータを保持する専用のサーバを 1 つ設置し，クライアントがサーバと通信をして仮想空間内部を動き回るアーキテクチャである．サーバとクライアント間の通信処理部分が比較的シンプルであり，機能に応じたサーバを選択する必要がないためクライアント側の負荷が少ない．しかしサーバ側に負荷が集中し，負荷分散の形態についてバリエーションが少ないという欠点がある．

(3) **一部ピア・ツー・ピア方式**

(2) に加えて，必要な機能のみクライアント間のピア・ツー・ピア接続方式で対処しようとするアーキテクチャである．多人数参加型のサービスを行う場合には (1) と同様の課題が存在する．またサーバが 1 つなので (2) であげられた問題も含む．

(4) **機能分散型サーバ方式**

機能やサービスに応じてサーバを分ける方式で，機能の追加や負荷の分散形態に柔軟に対応できる．クライアントはその時々に必要なサーバとのみ通信をすればよく，クライアントの処理能力に応じて選択することができる．

2.1.3 システムに求められる課題

分散仮想空間システムには，一貫性の保証とスケーラビリティ，そしてユーザの自由な途中参加が求められる．

(1) **一貫性の保証**

ユーザが仮想環境システムを使って空間を共有するには各ユーザが持つ空間データがまったく等しいものでなくてはならない．システムはそれらの空間データの一貫性保証を行う必要がある．空間データの一貫性は空間データの更新が複数のユーザで異なる順序で行われる場合に崩れる．一貫性保証を行うためには更新処理の順序保証が必要となる．サーバレス方式ではシステム全体で一意に決められた順序番号を付加し，メンバー全員へ送信する方法を用い，その他の方式では順序番号の発行を行うサーバをもうけ，順序番号の発行を受けてから更新情報を送信する．受信者は順序番号に基づいて空間データを更新するという方法が用いられる．

(2) **スケーラビリティ**

分散仮想空間システムには，参加しているユーザが増加しても少人数の場合と同様のサービスを提供できる性能が求められる．そのような性能を持つシステムは，スケーラビリティがあるシステムと呼ばれる．スケーラビリティは，ある処理に対するシステムの応答

性が低下する場合に損なわれる．その原因としてユーザの増加により，ある処理が完了するまでの時間が増加することがあげられる．この問題を解消するには何らかの方法で負荷の軽減，または分散を行わなくてはならない．一部ピア・ツー・ピア接続方式や機能分散型サーバ方式は他の 2 つの方式よりも，負荷分散が行いやすいのでスケーラビリティがあるといえる．

(3) **ユーザの自由な途中参加**

仮想空間データはユーザの更新処理によって随時書き換えられている．新たなユーザの途中参加を実現するためには，その時点での仮想空間データを新しいユーザに送信する状態転送プロトコルが必要になる．

2.1.4 従来の研究

(1) **DIVE**

DIVE（Distributed Interactive Virtual Enveronment）[1]はサーバレス方式のシステムでそれぞれのクライアントが仮想空間データのコピーを持つことにより仮想空間を共有する．DIVE では厳密な一貫性保証を行うために一貫性保証プロトコルを使用して，このプロトコルの上に三次元オブジェクトに対する様々な操作を実装している．また途中から参加するユーザに現在の仮想空間の状態を転送するための状態転送プロトコルも実装されている．このシステムは当初 LAN と高性能なワークステーションを対象としていたため，インターネットのような低帯域，高遅延のネットワークではスケーラビリティに問題があり，最大参加者は 10 人が限界である．

(2) **NPSNET**

NPSNET [2]は多数のユーザのアドレスへの IP マルチキャスティンググループを使った最初の分散仮想環境システムである．DIVE とは異なり，ユーザ全員に対して厳密な一貫性保証を行っていないため，1000 人以上のユーザが同時に利用しても問題なく動作する．しかし基本となっているプロトコルが軍事シミュレーションに大きく依存した内容となっていることと，MBONE（virtual Multicast Backbone On the interNEt）[7]に対応していないネットワークでは利用できないという問題がある．

(3) **Community Place**

Community Place（以下 CP）[3]は SONY で研究開発されているシステムで，複数のユーザが同時にアクセスできる三次元の共有仮想社会（Virtual Society）をインターネット上に構築するためのソフトウェア群である．VRML [4]ブラウザである CP ブラウザ，サーバプログラムである CP ビューロ，そして仮想空間内の物体とそれに関連づけられたアプリケーションである CP アプリケーションオブジェクトの 3 つからなる．現在 Community Place システムをベースとした PAW（Personal Agent World）が実装されている．

2.2 我々の研究によるシステム

2.2.1 システムの構成

本研究は，我々の研究で実装された分散仮想環境システムの拡張を行うという形で進めるので，ここでそのシステムについて説明する[5),6)]．システムは前述した一部ピア・ツー・ピア方式で構成されており，仮想空間データサーバと順序番号発行サーバとクライアントが存在する．

- 仮想空間データサーバ
  - 仮想空間データのオリジナルを保持する．
  - 一貫性管理モジュール，グループ通信プロトコル，参加メンバー管理モジュールで構成されている．
  - 新しいユーザからの参加要求を受け取ったとき，新ユーザに対し，現在の空間データとメンバーリストの引き渡しを行う．メンバーリストはメンバーのソケットアドレスとポート番号を 1 項目とした線形リストとして構成されている．
- 順序番号発行サーバ
  - 一貫性を管理するための順序番号を発行するサーバ．
  - このシステムでは，システムに参加しているユーザのだれもがこのサーバになりうるが，同時に 2 つ以上のユーザが順序番号発行サーバになることはない．
- クライアント
  - クライアントはシステムに参加するユーザが利用する．
  - 一貫性管理モジュール，グループ通信プロトコル，仮想空間の描画モジュールで構成されている．
  - ユーザに仮想空間を閲覧，または操作するインタフェースを提供する．
  - ユーザが仮想空間の更新を行ったとき，参加メンバーへの更新情報の送信や再送管理をする．

2.2.2 システムの機能

ここではシステムが実際にどう動作するのか説明する．

(1) **仮想空間への参加処理**

仮想空間への参加処理の流れを順に説明していく．

- サーバは起動すると初期状態の仮想空間データを

読み込んで，クライアントの接続を待つ．

- 新クライアントはサーバの位置をユーザから得て，システムへの参加要求を出す．
- 参加要求を受けたサーバは，現在のメンバーリストを新クライアントに送信する．
- メンバーリストを受け取った新クライアントは，自分自身をメンバーリストに加える．新クライアントはそのメンバーリストを参照し，順序保証通信プロトコルを使用してメンバー全員にシステムへ参加したことを通知する．
- メンバーは受け取った参加通知から，新クライアントのアドレスを読み込み，自分が持つメンバーリストに加える．
- 次に新クライアントは順序保証通信プロトコルを使ってサーバに空間データの要求を出し，サーバから現在の仮想空間データを受信する．この処理を行っている間に他のメンバーから空間データの更新情報が送られてきている可能性があるが，順序保証通信プロトコルを使用しているので空間データを受信し終わってから更新を行い，他のユーザが持つ空間データとの一貫性を保つことができる．

(2) 仮想空間の記述と表示

仮想空間は，空間内のオブジェクトの形を表す形状情報ファイルと，オブジェクトの種類や位置や回転などの情報を持つ仮想空間データファイルによって形成される．形状情報ファイルは三次元空間記述言語であるVRMLを独自拡張した言語で記述される．仮想空間データはシステムの実行にともなって変化するが，オブジェクトの形状情報は不変である．クライアントは形状情報ファイルを構文解析して3D描画のためのオブジェクトシーングラフを作成する．クライアントは仮想空間データが更新されると，仮想空間データとオブジェクトシーングラフから3D画像を描画する．3D画像の描画には3Dグラフィックライブラリ OpenGL[8]を使用している（**図2**）．

(3) 順序保証グループ通信

分散仮想環境システムでは，複数のユーザが空間内の1つのオブジェクトに同時に操作を行う状況が発生する．このような場合，それらの更新情報がどのような順序で届くかによって，ユーザの間で空間データに食い違いが起こる．順序保証グループ通信は，操作が重複し，衝突を起こす可能性のある更新メッセージの管理をして空間データの一貫性保証を行う．具体的な動作を**図3**として示す．順序保証グループ通信の要求が発せられると，まずメンバーの中にいる順序番号発行権を持つユーザに対して順序番号の発行を要求する通信を送る．順序番号発行権を持つユーザは番号を発行して，ユーザに送り返す．番号を受け取ったユーザは，送信する情報に順序番号を付加してグループに通信する．

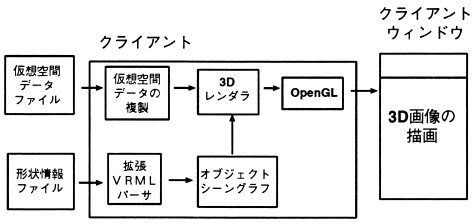


図2 仮想空間描画部のシステム
Fig. 2 The system of virtual space drawing module.

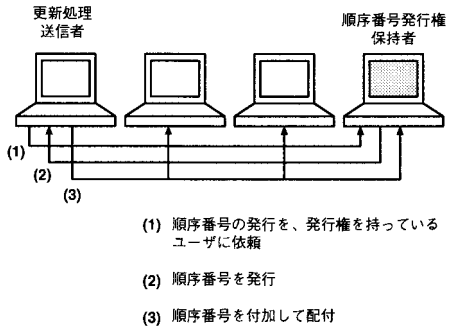


図3 順序番号の付加
Fig. 3 Issuance of sequencial number.

順序保証グループ通信に対して，非順序保証グループ通信がある．この通信はユーザの間で操作の衝突が起こらない状況で使用され，順序保証が行われない．

DVSでは順序番号を発行する権利がメンバー内を移動する方式がとられているが，本研究の内容とは関係が薄いので詳細な説明は割愛する．

(4) 仮想空間の共有

(3)であげたような操作は，それぞれのユーザが持つ複製された仮想空間データ内だけで行われる．だが分散仮想環境システムは空間の共有を目的としているので各ユーザが自分の仮想空間データに対して行った更新情報を，サーバを含む参加メンバー全体に送信する必要がある．ここでそれぞれの操作を行ったときクライアントがどのように送信処理を行うか説明する．

DVSにおいて最も頻繁に発生する更新処理はユーザの移動である．ユーザの移動は，それぞれのユーザにしか行えないので非順序保証グループ通信を使用する．ユーザがマウスをドラッグすると，クライアントはマウスの軌道に沿ったオブジェクト移動通知をメンバー全員に連続送信する．アヴァタ以外の一般オブ

ジェクトはすべてのユーザが更新を行うので非順序保証グループ通信だけでなく順序保証グループ通信も利用する．

ユーザがオブジェクトを操作するには，まずオブジェクトの所有権を得る必要がある．所有権を得るというのは，指定したオブジェクトが所有権を持つユーザからの操作しか受け付けないようにロックすることである．ユーザは一般オブジェクトを操作するために，順序保証グループ通信でオブジェクトの所有権を取得する．ユーザがドラッグすると，非順序保証グループ通信を使用してオブジェクト移動通知をメンバー全員に連続送信する．マウスが放されるとクライアントは順序保証グループ通信でオブジェクトの所有権を開放する．オブジェクトに対する移動要求を受け取ったクライアントは仮想空間データを更新し，新しい 3D 画像を描画する．更新要求は連続的に到着するので，参加メンバーは，他のユーザが行っているオブジェクトの移動をなめらかに見ることができる．

ユーザがオブジェクトの新規作成や削除，固有動作の起動を行うと，クライアントは順序保証グループ通信を使用して要求を送信する．クライアントは新規作成と削除を受信すると仮想空間に処理を行い，新しい 3D 画像を描画する．固有動作の起動を行うと，クライアントはタイマを起動する．そして一定時間間隔でオブジェクトの固有動作に従った処理を実行し，新しい 3D 画像を描画する．つまりオブジェクトの固有動作に関しては，動作を行うかどうかをそれぞれのユーザが指定することができ，実際の動作処理は各ユーザに分散されている．

(5) **通信処理**

従来のシステムでの通信は基本的に ACK を使って再送管理を行う 1 対 1 通信を利用している（**図 4**）．送信者はメッセージを送信すると，メッセージを ACK 待ちリストに格納し，タイマを起動する．メッセージの受信者は受信すると即座に送信者に ACK を返す．ネットワークの異常などで ACK が帰らずタイマがタイムアウトを起こした場合，送信者はメッセージの再送を実行する．

メンバー全員へメッセージを送信するグループ通信は，この 1 対 1 通信をメンバー全員に対して行うことで実現している．

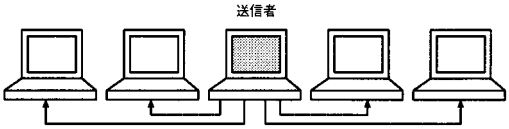


図 4 グループ通信処理
Fig. 4 Group communication processing.

2.2.3 問 題 点

このシステムではグループ通信のために，送信者は 1 対 1 通信をメンバー 1 人 1 人に対して行っている．しかしシステムへの参加メンバー数が増大すると送信処理に時間が必要となる．またメッセージ受信者からの ACK の集中や，タイムアウトによる再送処理のために送信者周辺のネットワークに過負荷がかかるようになる．

このような理由でこのシステムにはスケーラビリティにおける問題が残されていた．

3. 提 案 方 式

本章では，我々の研究による分散仮想環境システムの通信処理に関する問題点を解消するための方式を提案し，その実装方法を説明する．

3.1 提案システム

前述した我々の研究によるシステムの問題点を解消するには，まず送信処理を軽量化すること，そして再送管理のための負荷を軽減させることが必要であると考えられる．そこでグループ通信に関しては送信処理をマルチキャストで行う．そして参加ユーザ 1 人 1 人に，再送管理を受けるユーザと再送管理をするユーザを割り当て，メッセージを受けられなかったユーザに対する再送処理を送信者以外に分散させて行う方式を提案する．

システムに参加するユーザは事前にマルチキャスト IP アドレスを知っており，起動時にマルチキャストのためのソケットを作り，マルチキャストグループに入る．メンバー全体への送信はマルチキャストを利用して送信する．

ユーザ間で相互に再送管理を行うためにメンバーを木構造で管理する．木構造上で，上位ノードは下位ノードに対して再送管理をする．ユーザは不規則に参加，脱退を行うので，メンバーの木構造を動的に変更できるメンバー管理が必要となる．

3.2 メンバー管理

メンバー管理には木構造を用いる．仮想空間に参加するユーザは不規則に参加，脱退を行うので，動的に変更できる構造が必要である．ここではメンバーリストの構造と参加，脱退処理におけるメンバーリストの操作ついて説明する．

(1) メンバーリストの構造

ユーザ同士の再送管理の関係を表すために，メンバーリストの項目を木構造に基づいて構成する（**図 5**）．

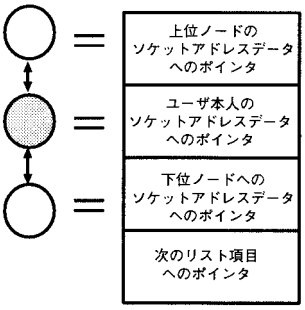


図 5 メンバーズノードリストの 1 項目
Fig. 5 An item of the members node list.

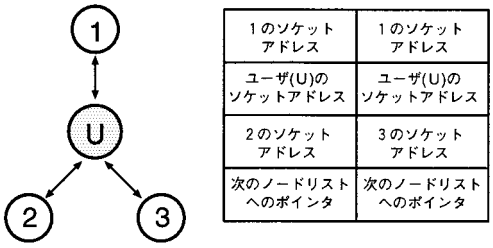


図 6 ノードリストによる二分木表現
Fig. 6 The tree structure based on the node list.

リストの 1 項目はさらに 4 項目からなり，

1. ユーザ本人のアドレス．
2. 上位ノードにあたるユーザのソケットアドレス．
3. 下位ノードにあたるユーザのソケットアドレス．
4. そして次のリスト項目へのポインタ．

で構成されている．

ここで上位ノードとは，自分が再送管理を受けるユーザのことで，下位ノードとは自分が再送管理を行うユーザのことである．このリストを複数用意し，下位ノードを示す内容を変えることによってユーザを中心とした木構造を表現することができる．こうして構成されたメンバーリストをメンバーズノードリスト，そしてメンバーズノードリストの 1 項目をノードリストと呼ぶこととする．

(2) 仮想空間への参加処理

図 6 のように，ノードリストは基本的に二分木を形成することとし，この方式ではサーバを root とした木構造を作るよう設計した．そのために参加処理を行うとき同一ユーザのためのノードリストは 2 つ作成され，メンバーズノードリストに追加される．

システムはこれまでと同じようにサーバを起動することから始まる．

- 起動したサーバは自分をユーザ本人としたノードリストを 2 項目作成し，メンバーズノードリストに組み入れる．このときサーバのノードリストにおける上位ノード，下位ノードはともに NULL である．
- 新クライアントも起動と同時に自分自身が持つノードリストを 2 項目作成し，サーバへ参加要求メッセージを送信する．
- 参加要求を受け取ったサーバはメンバーズノードリストを先頭から読み，下位ノードが NULL であるノードリストを検索する．
- サーバはシステムへの参加許可メッセージとともに，その項目のユーザ（今の状態ではサーバ）のソケットアドレスを新クライアントに返す．
- その後，サーバは検出したユーザを上位ノードとした新クライアントのノードリストを 2 項目作成してメンバーズノードリストの最後尾に追加する．

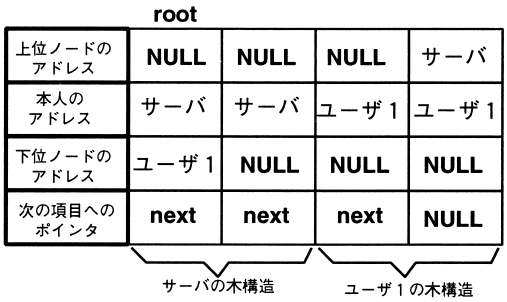


図 7 ユーザが 1 人いる状態のメンバーズノードリスト
Fig. 7 The members node list for one user entry.

ユーザが 1 人参加した状態のメンバーズノードリストを**図 7** に示す．

なお，上で説明したのは上位ノードとなるユーザがサーバであるときの動作である．ユーザが 3 人目以降の場合，新クライアントは別のクライアントの下位ノードとなるので処理が変わる．

- サーバは新クライアントの上位ノードとなるユーザを発見したとき，上位ノードのユーザに下位ノード追加メッセージを送信し，新クライアントを新しい下位ノードに登録させる．
- その後，新クライアントに上位ノードのソケットアドレスを送信する．

以上の操作を行うことで，サーバを root とした木構造が形成される（**図 8**）．また木構造全体を把握するためのデータはサーバのみが持っており，各ユーザは自分の上位ノード，下位ノードの情報だけを持つシステム構造が形成できる．

(3) 仮想空間からの脱退処理

システムから脱退するときは，ノード構造を変更する必要がある．本システムでは脱退するユーザの下位

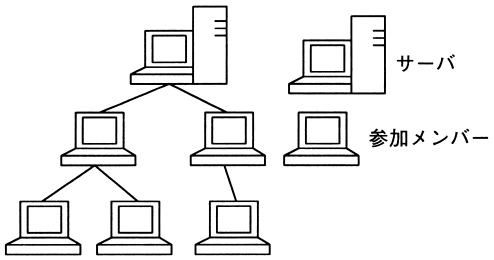


図 8 メンバーの木構造
Fig. 8 The tree structure of members.

ノードを上位ノードの下位ノードとして再構成する．

- 脱退するユーザはサーバに脱退要求メッセージを送信する．
- 要求を受けたサーバは脱退するユーザに適合するノードリストをメンバーズノードリストから検索する．検出したノードリストに従い，上位ノードには下位ノード追加メッセージ，下位ノードには上位ノード変更メッセージを送信する．それぞれのメッセージには追加，または変更するノードのソケットアドレスが付加されている．
- 下位ノード追加メッセージを受け取った上位ノードは新しいノードリストを作成して自分が持つノードリストに加える．
- 上位ノード変更メッセージをを受け取った下位ノードは自分の上位ノードを設定し直す．
- 新しいノード構造を送信したサーバは，自分が持つメンバーズリストも同様に書き換え，脱退するユーザに脱退許可メッセージを送信する．

メンバーズノードリストは1項目で1本のノードを表しているので，1項目を参照するだけで処理に必要なノード構造を得ることができるという利点がある．また，項目を増やすだけで二分木に限らず様々な分岐をする木構造を表すことが可能である．

3.3 再送管理

送信されたグループメッセージに対し，どのような処理を行うか説明する．

ユーザは受信したメッセージを再送処理のために一時的に ACK 待ちリストとして保存しておく．ACK 待ちリストの項目は，下位ノードそれぞれに対して1つずつ作成される．また，下位ノードからの ACK がメッセージよりも先に届く場合があるので，下位ノードからの ACK テーブルとして一時的に保存しておく．

(1) グループ通信メッセージが下位ノードからの ACK より先に届いた場合（**図 9**）

1. ユーザがグループ通信メッセージを受信すると，ユーザはただちに ACK を上位ノードに返す．

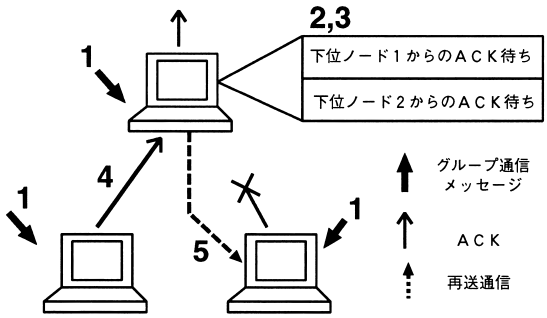


図 9 メッセージが先に届いた場合
Fig. 9 The message arrived earlier than ACK from lower node.

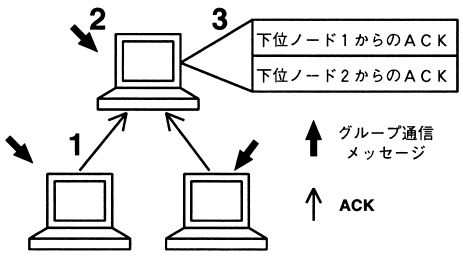


図 10 ACK が先に届いた場合
Fig. 10 ACK from lower node arrived earlier than the message.

2. ユーザは下位ノードからの ACK テーブルを検索し，適合する下位ノードからの ACK があった場合リストから削除し，受け取ったグループ通信メッセージを実行する．

3. 適合する下位ノードからの ACK がなかった場合，メッセージを下位ノードそれぞれに対する ACK 待ちリストに加え，タイマをスタートさせる．

4. 下位ノードからの ACK が届き次第，適合する ACK 待ちリストを削除し，タイマをリセットする．

5. もしタイマがタイムアウトを起こしたら，ユーザはその下位ノードにメッセージが届いていないと見なして再送処理を行う．

(2) グループ通信メッセージよりも下位ノードからの ACK が先に届いた場合（**図 10**）

1. 下位ノードからの ACK が届いた場合，ユーザは ACK 待ちリストを検索して適合する ACK 待ちメッセージを探す．

2. 適合するメッセージがあった場合，ユーザは再送のためのタイマを止め，ACK と ACK 待ちリストを消去する．

3. 適合するメッセージがなかった場合，ユーザには ACK に対応するメッセージが届いていないとい

うことなので下位ノードからの ACK テーブルに加える．

3.4 メッセージの流れ

本方式でのグループ通信処理は ACK の送受信や再送処理がサーバを root とした木構造に沿って，各クライアントで独立して実行される．そのためサーバに対する再送管理はメッセージの送信者が受け持つよう設計した．メッセージの送信者は，マルチキャスト IP アドレスへのメッセージ送信処理が終わり次第，上位ノードに ACK を送り，下位ノードとサーバに対する ACK 待ちリストを作成する．

最初のグループ通信でメンバーのだれにもメッセージが届かなかったような場合，送信者が行う再送管理により，グループメッセージはノードの最上部に伝わり，上から順にメッセージが行き渡ることとなる．送信者は必ず送信メッセージを持っているため，送信者から下のすべてのノードには送信者自身による再送処理でメッセージが伝えられる．それによって送信者より下のノードには再送メッセージが早く伝わることになる．

4. 考　　察

本研究のシステムの特徴として，再送処理の分散管理とメンバー管理があげられる．この章ではそれらについて考察する．

- 再送処理の分散管理
  本研究のシステムでは，グループ通信メッセージが到着し次第上位ノードに ACK を送ることで処理する方法をとっているため，再送管理がノード間だけで独立している．つまり再送管理がメンバー管理の木構造にあまり依存しない．そのため再送処理を高速に行うことができるといえる．
- メンバー管理
  メンバー全員の管理はメンバー管理を行うサーバを置いて集中管理をしている．メンバー管理を分散管理で行った場合，入退出時の通信コスト，スケーラビリティ，システムの信頼性において分散管理の方がわずかに上回っている．しかし分散仮想環境システムではメンバーの入退出が頻繁に起こるわけではないので，入退出時の通信コストがシステムに与える影響は小さい．またメンバーが多くなったとき，メンバーズノードリストが大きくなるがこれもスケーラビリティに与える影響は小さい．システムの信頼性はバックアップするサーバを置けば解決が可能である．

  集中管理の場合の大きな利点として，各ノード間の物理的な距離による通信速度などを基に，木構造を最適な形に組み替えられるという点があげられる．分散仮想環境システムでは再送処理がより速く，確実に行うことが求められるため木構造の最適化が容易にできることは大きい．

これらのほかに，本システムでは物理的なデータ配送経路や手段と，再送のためのネットワーク構成が独立しているという特徴を持つ．これは配送経路の変更や媒体の変更が容易であり，端末の移動にも柔軟に対応できる．

5. ま と め

本研究では，分散仮想環境システムの送信処理の軽量化と，再送処理を分散させるための新しいメンバー管理プロトコルを設計，提案した．

送信処理の軽量化にはマルチキャストを用いることで軽量化した．再送処理は送信者以外のメンバーを木構造で集中管理させる入退出プロトコルを設計し，上位ノードにあたるユーザが，下位ノードのユーザの再送を管理することでメッセージ送信者以外のユーザに再送処理をを分散させることとした．提案した木構造の構成方式は再送のためのメンバー管理が独立しており，木構造の最適化を行うのが容易である．

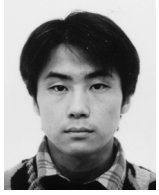
本論文で提案したプロトコルを利用することで，分散仮想環境システムがネットワークにかける負荷を分散させ，軽減することができると思われる．さらにメンバー管理のための木構造を最適化するプロトコルを追加することが課題として残されている．

参 考 文 献

1) Hagsand, O.: Interractive Multiuser Ves in the DIVE System, *IEEE Multimedia Magazine*, Vol.3, No.1 (1996).
2) Macedonia, M.R., Zyda, M.J., Paratt, D.R., Bartham, P.T. and Zeswitz, S.: NPSNET: A Network Software Architecture for Lage Scale Virtual Environment, *Presence*, Vol.3, No.4 (1994).
3) 日本バーチャルリアリティ学会，仮想都市研究会第 1 回シンポジウム，VR 学研報，Vol.1, No.1 (1997).
4) 山本精一：VRML2.0 パーフェクトガイド，技術評論社 (1996).
5) 岡田忠義，福岡久雄，佐藤文明，水野忠則：順序発行権移動方式による分散仮想空間の一貫性管理，第 56 回情報処理学会全国大会論文集（4），pp.217–218 (1998).
6) 福岡久雄，岡田忠義，佐藤文明，水野忠則：分散仮想環境システムにおけるイベント順序保証機構

の部分適用方式の評価，電子情報通信学会論文誌（B-I），Vol.J82-B-I，No.5，pp.928–938（1999）. http://riki.cs.inf.shizuoka.ac.jp/jp/papers/dve/dve.html
7) Kumer，V.（著），楠本博之（監訳）：インターネットマルチキャスト MBone，インプレス (1996).
8) Neider, J., Davis, T. and Woo, M.: *OpenGL Programing Guide*, Addison-Wesley Publishing Company (1993).

(平成 11 年 4 月 30 日受付)
(平成 11 年 12 月 2 日採録)



江崎　美仁

昭和 51 年生．平成 11 年静岡大学工学部知能情報工学科卒業．同年静岡大学大学院理工学研究科計算機工学専攻入学，現在に至る．通信ソフトウェア，分散共有システムに興味を持つ．

佐藤　文明（正会員）

昭和 37 年生．昭和 61 年東北大学大学院工学研究科及通信工学専攻博士前期課程修了．同年三菱電機（株）入社．平成 7 年 1 月より静岡大学工学部助教授．現在，静岡大学情報学部助教授．通信ソフトウェア，形式言語，分散処理システムに関する研究に興味を持つ．電子情報通信学会，IEEE Computer Society 各会員．工学博士．