グループウェア 29−1
（1998. 9. 17）

プレゼンテーションシステムにおけるリアルタイムアプリケーション共有方法の検討

青柳 慶光　　藤岡 秀樹
日立ソフトウェアエンジニアリング（株）

コンピュータ会議システムをの基本機能の一つであるリアルタイムアプリケーション共有機能の実現方式について報告する。既存のプレゼンテーションシステムのアプリケーション共有機能を利用した場合、(1)マウスポインタを共有するために個人作業が制限される。(2) アプリケーションを起動したコンピュータ上のファイルしか参照できない。(3) 画面描画に遅延が発生する。などの問題があった。そこで、この問題点を解決するため、OLE 技術を利用したアプリケーションデータのオブジェクト化、共有アプリケーションの分散実行、オブジェクトに対するアクセス権導入による遅延同期方式を採用したホワイトボードシステムを試作した。CPU 利用率、ネットワークトラフィックに関する性能測定の結果、本システムのアプリケーション共有機能の有効性を確認した。

Method Of Real-time Application Sharing on Presentation System

Yoshimitsu Aoyagi　　Hideki Fujioka
Hitachi Software Engineering Co., Ltd.

This paper reports the method of real-time application sharing, one of the most important functions of Computer Meeting System. Application sharing of current real-time presentation system has some problems. (1) Personal operation is restricted by mouse pointer sharing. (2) User can access only local files of the computer where the shared application is running. (3) Screen depiction is delayed. We solved these problems adopting following methods, expressing application data as an object by using OLE technology, running shared application on every distributed computer, and adopting delayed synchronization with privilege to access object. The result of measuring CPU performance and network traffic shows the effectiveness of our system.

1. はじめに

近年コンピュータを利用して人間の共同作業を支援するグループウェアが注目されている[1]。中でもコンピュータ会議システムでは、実際の机上の会議[2]と同様に参考資料に基づいた議論を実現するための機能として、プレゼンテーションデータの表示や編集を実行するためのアプリケーション共有機能[3]を有するリアルタイムプレゼンテーションシステムとして実現されている[4]。しかし多くのアプリケーション共有機能は、複数ユーザが共同でプレゼンテーションデータを編集するといった機能面に重点が置かれており、情報共有を主眼においた利用方法では操作性や性能面で制限事項が発生することがわかった。我々は操作性や性能面に重点を置いたユーザインタフェースを検討し、プレゼンテーションシステム(以下ホワイトボードシステムと呼ぶ)の試作を実施した。

2. アプリケーション共有時の問題点

アプリケーション共有機能の利用時に発生する操作性や性能面の制限事項を以下に示す。

(1)マウスポインタ共有による個人作業の制限

一般にアプリケーション共有機能には、共同作業状態と単独作業状態の 2 種類の状態が存在する。

共同作業状態とは共有アプリケーションデータの編集を複数ユーザが共同で実行する状態である。共同作業状態ではマウスポインタが共有される。その結果、実際にマウスを操作できるユーザはマウスの操作権を獲得したユーザに限られる。マウスの操作権を保持していない他のユーザは自分の個人作業を実行することができない。

単独作業状態とはアプリケーションを共有したユーザだけが共有アプリケーションへの操作が可能な状態である。単独作業状態では共有アプリケーションのウインドウを操作できるユーザがアプリケーション共有者に限定されるため、それ以外のユーザは表示する必要がない共有アプリケーションをアイコン化するような操作を実行することができない。

このように共同作業状態ではマウスポインタが共有される点、単独作業状態では共有アプリケーションに対する操作がすべて禁止されるという点で操作性の制限が発生する。

(2)ディスクアクセスの制限

多くのプレゼンテーションシステムでは、アプリケーション共有機能の実現方法に、共有アプリケーションが1台のコンピュータだけに稼動する集中方式[4]を採用している。共有アプリケーションが稼動しているコンピュータで表示されるウインドウと同一のウインドウがすべてのコンピュータ上で表示される。そのため、ファイルデータを参照する場合でも、共有アプリケーションが稼動しているコンピュータのディスクだけしか参照することができないという操作性の制限が発生する。

(3)共有アプリケーションウインドウ描画時の遅延

共有アプリケーションへのキーボード入力による編集、マウスによる共有アプリケーションウインドウの移動など、共有アプリケーションウインドウにイベントが発生するたびにウインドウの再描画が発生する。この時、ウインドウ描画のレスポンスが遅延するという性能上の制限が発生する。リアルタイム同期処理により、断続的にコンピュータのCPUを使用することが原因として挙げられる。

3. ホワイトボードシステムの実現

3.1 制限を回避するための方針

ホワイトボードシステムはアプリケーション共有機能の一部に制限を加えることで、操作性や性能面の制限回避を実施した。以下にその方針を示す。

(1)共同作業空間と個人作業空間の分離

共同作業空間で実行される作業と個人作業空間で実行される作業を独立に動作させることで、個々の作業を実行中は別のユーザから割込が発生しない。

(2)アプリケーションプロセスの分散化

すべてのコンピュータ上で共有アプリケーションを起動することで、個々の共有アプリケーションからはそのコンピュータのディスクをアクセスできる。

(3)遅延同期処理による描画性能改善

同期通信をリアルタイムに行わず、入力データをある程度収集し、一括して同期することで、同期通信の回数を減らすことができ、共有アプリケーションウインドウ再描画時の遅延を回避できる。

3.2 ホワイトボードシステムの実現方式

ホワイトボードシステムの実現方式を表1に示す。

表1 ホワイトボードシステムの実現方式

項番	制限事項	制限回避の方針	実現方式
1	マウスポインタ共有による個人作業の制限	共同作業空間と個人作業空間の分離	データのオブジェクト化による編集状態/参照状態の分離
2	ディスクアクセスの制限	アプリケーションプロセスの分散化	分散方式の採用
3	共有アプリケーション描画時の遅延	一括同期処理による描画性能改善	アクセス権による同時編集機能の禁止

(1)データのオブジェクト化

共有アプリケーションのデータをオブジェクト化し、個々のオブジェクトに編集状態/参照状態という2つの状態を定義する方式を導入した。オブジェクトの状態はデータ編集時だけ編集状態に遷移し、それ以外は参照状態にする。ユーザは個々の作業の間、自分が編集すべきデータのオブジェクトの状態だけを編集状態にすれば良くなる。ホワイトボードシステムは MS-Windows をプラットフォームとしたため、オブジェクトの実現に OLE 技術を適用した。

(2)分散方式[4]の採用

共有アプリケーションをすべてのコンピュータで起動する分散方式を採用した。OLE オブジェクトは同一コンピュータ上のアプリケーションと関連づけられており、OLE オブジェクトの編集時には同一コンピュータ上の共有アプリケーションが起動され、入力データが処理される。

(3)アクセス権[5][6]による同時編集機能の禁止

ホワイトボードシステムでは、データを編集できるユーザは1人とし、編集終了時に一括して更新データを転送する方式を採用した。複数ユーザによる同時編集を禁止するため、オブジェクトにはアクセス権を導入した。同時編集を禁止し、データを編集できるユーザを1人とすることで、同期通信はデータ編集開始時とデータ編集終了時の2回に減らすことができ、断続的な同期通信の発生を回避できる。

3.3 ホワイトボードシステムの構成

ホワイトボードシステムの構成を図1に示す。クライアントコンピュータでは、OLE オブジェクトという共通のフォーマットで作成された手書き文字データやアプリケーションデータを表示することができるホワイトボードクライアントが稼動している。個々の OLE オブジェクトは同一コンピュータ上で稼動する共有アプリケーションに関連づけられている。サーバコンピュータでは、OLE オブジェクトのアクセス制御/同期制御/バージョン管理を実行するホワイトボー

ドサーバが稼動している。

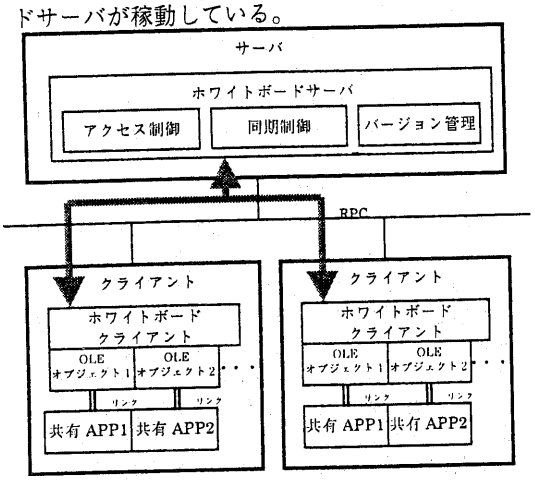

図1 ホワイトボードシステムの構成

(1)ホワイトボードサーバ

アクセス制御は同一の OLE オブジェクトを複数のユーザが編集できないように制御する機能である。同期制御は OLE オブジェクトに対する変更データを収集し、すべてのホワイトボードクライアントに転送する機能である。バージョン管理はクライアントの OLE オブジェクトの整合性が壊れた場合(3.4 節で説明する)、クライアント間の整合性を回復する機能である。ホワイトボードサーバは上記の3つの機能を実現したプログラムである。

(2)ホワイトボードクライアント

ホワイトボードクライアントは OLE オブジェクトの表示/ホワイトボードサーバとの同期通信という機能を実現したプログラムである。

3.4 ホワイトボードサーバの実現方法

3.4.1 アクセス制御

3.2 節で述べたように、ホワイトボードサーバがオブジェクトのアクセス権を付与する方式を採用した。図 2 にホワイトボードサーバのアクセス制御の処理を示す。

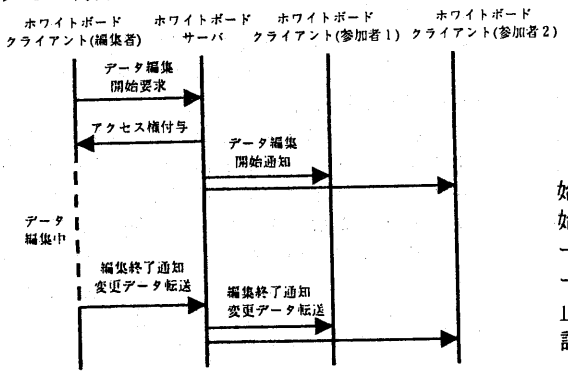

図2 ホワイトボードサーバのアクセス制御の処理

サーバ/クライアント間でトラフィックが発生するのは編集者がデータをホワイトボードサーバに要求し、アクセス権を取得する場合とデータ編集が終了して変更データをホワイトボードサーバに転送する場合の2回で済み、断続的な同期を回避している。ホワイトボードサーバと参加者が利用するホワイトボードクライアント間の同期通信も同様に2回となる。

3.4.2 同期制御

オブジェクトに対するユーザ操作イベントは以下の4通りを想定した。

(a)オブジェクト移動/リサイズ
(b)オブジェクト編集状態へ移行(アクセス権取得)
(c)オブジェクト参照状態へ移行(変更データ転送)
(d)オブジェクト削除

同期通信の回数を減らすため、オブジェクトに対するユーザ操作イベントによる同期制御方法を以下のように分類した。

(1)クリティカルな同期

クリティカルな同期とはホワイトボードサーバとの同期通信終了までホワイトボードクライアントはユーザインタフェースを停止する同期方法である。上記(b)、(c)のユーザ操作イベントが発生した場合、クリティカルな同期を実行する。クリティカルな同期処理を図3に示す。

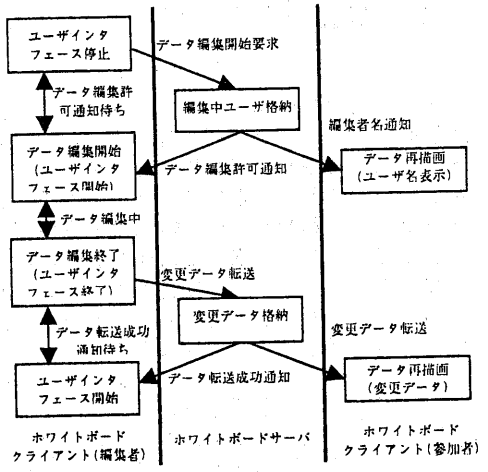

図3 クリティカルな同期処理

ホワイトボードクライアントはデータ編集開始時にホワイトボードサーバからデータ編集開始許可を受ける。データ編集開始要求時からデータ編集許可通知受信時までの間、ホワイトボードクライアントはユーザインタフェースを停止してホワイトボードサーバからのデータ編集許可通知を待つ。同様にデータ編集終了時にホ

ワイトボードサーバに変更データ転送を実行し、データ転送成功通知の受信を待つ。変更データ転送開始時からデータ転送成功通知受信時まで、ホワイトボードクライアントはユーザインタフェースを停止する。他の参加者にはホワイトボードサーバから、データ編集開始時に編集中ユーザ名が通知され、データ編集終了時に変更データが転送される。その都度、ホワイトボードクライアントはデータ再描画を実行する。

(2)あいまいな同期

あいまいな同期とはホワイトボードサーバとの同期通信終了を待たず、ユーザインタフェースと同期通信を並行して実行する同期方法である。あいまいな同期処理を図4に示す。

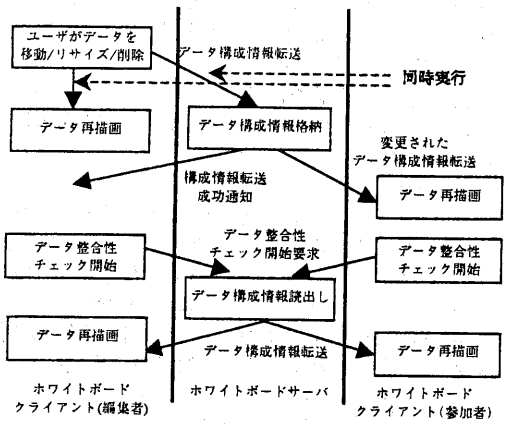

図4 あいまいな同期処理

上記(a)、(d)のユーザ操作イベントが発生した場合、あいまいな同期を実行する。すべてのデータは位置やサイズを格納したデータ構成情報を保持しており、ホワイトボードクライアント上でデータ移動/リサイズ/削除の操作が実行された場合、データ構成情報が変更された後、ホワイトボードサーバに転送される。データ構成情報転送と並行して変更されたデータの再描画が同時実行される。これによりホワイトボードクライアントのユーザインタフェースを停止しないで済むという利点があるが、あいまいな同期通信が失敗した場合、クライアント間でデータの整合性が壊れるという欠点も生じる。ホワイトボードシステムではデータの整合性崩壊時の対策としてデータ整合性チェック機能を追加した。ホワイトボードサーバは、ホワイトボードクライアントからデータ整合性チェック開始要求が転送された場合、格納してあるデータ構成情報を読出し、ホワイトボードクライアントに転送する。ホワイトボードクライアントはホワイトボードサーバから転送されたデータ構成情報を元にデータの再描画を実行する。データ整合性チェックにより、すべてのホワイトボードクライアント上のデータは、ホワイトボードサーバに格納されているデータ構成情報と同期するため、データの整合性が確保される。

3.4.3 バージョン管理

3.4.2 項で述べたデータ整合性チェックではデータの位置やサイズの整合性をチェックする機能を提供した。バージョン管理では、データ自体の整合性確保の機能を提供する。ホワイトボードクライアント上に表示されているデータとホワイトボードサーバに格納されているデータと同一であるか比較するチェック方法では、頻繁なデータ転送が発生し、トラフィックが増大する問題が発生する。またデータを比較することにより、ホワイトボードサーバの負荷が増大する問題が発生する。ホワイトボードシステムではデータにバージョン番号を付与し、データ比較時にバージョン番号を比較する方法を採用することで上記の問題を回避した。データのバージョンチェックはデータ整合性チェック時に同時に実行する。ホワイトボードクライアント上に表示されているデータのバージョン番号とホワイトボードサーバに格納されているデータのバージョン番号が不一致の場合、ホワイトボードサーバ上に格納されているデータがホワイトボードクライアント上にダウンロードされる。

4. ホワイトボードシステム試作結果の検討

現在多くの企業内で OA 用に利用されているリアルタイムプレゼンテーションシステムに Microsoft 社の NetMeeting がある。ホワイトボードシステムの評価を実施するため、今回は NetMeeting との比較結果を用いることにした。

4.1 ホワイトボードシステムの機能評価

ホワイトボードシステムと NetMeeting の機能面での比較を表 2 に示す。ホワイトボードシステムはリアルタイムプレゼンテーションシステムのアプリケーション共有機能に一部制限を加えることで、2 章で述べた操作性や性能面での制限を回避することができた。

(1)複数ユーザによるデータ同時編集機能の制限

ホワイトボードシステムはマウスポインタを共有することで実現できる複数ユーザによるデータの同時編集機能を制限した。その結果、ユーザが別のユーザからの割込を受け、個人作業が制限されるという操作性の制限を回避できた。

(2)アプリケーション共有機能の制限

ホワイトボードシステムはディスクアクセス

の制限を回避するために分散方式を適用した。その結果、ホワイトボードシステムでは OLE 対応のアプリケーションだけしか共有できないという制限が発生した。しかし、プレゼンテーションに利用する Windows アプリケーションは OLE 対応のものが多いため問題はない。むしろ、OLE という標準形式を扱うことで分散方式を実現が容易になる。

(3)リアルタイムデータ更新

共有アプリケーションウインドウや共有データの再描画の回数を低減することで描画遅延を回避することができることから、ホワイトボードシステムはデータ編集終了時のみデータ更新を実行する方式を採用した。それと引き換えに、データ編集中は他のコンピュータに変更が反映されないという制限が発生した。

他にもホワイトボードシステムは共有アプリケーションをすべてのコンピュータにインストールしなくてはならないという制限がある。これに対しては運用で対応することになる。

表2 NetMeeting とホワイトボードシステムの機能比較

機能	ホワイトボードシステム	NetMeeting
複数ユーザによるデータ同時編集	△(1人だけに編集を許可)	△(書き込みができるのは1人だけ)
アプリケーション共有	△(OLE 対応のアプリケーションだけ)	○
リアルタイムデータ更新	△(データ編集終了時に更新)	○
データ格納場所	○	×(ローカルディスクのデータのみ)
APP データと手書き文字の同一ウインドウへの表示	○	×
データへのアクセス制御	○	×
会議の状態の保存	○	×
会議中の個人作業	○	△(同時編集モードでマウスも共有)
アプリケーションインストール	×(すべてのコンピュータにインストール)	○(不要)
ネットワークサポート	△(RPC 通信のみサポート)	○

4.2 ホワイトボードシステムの性能評価

ホワイトボードシステムの描画性能を評価するため、NetMeeting 利用時とホワイトボードシステム利用時のコンピュータの CPU 使用率やネットワークトラフィックを測定した。測定時の環境を表3に示す。

表3 性能測定時の環境

項目	CPU	RAM	OS	Network
サーバ	Pentium300	192MB	Windows NT Server 4.0	10Mbps LAN
クライアント	Pentium120	128MB	Windows NT Workstation 4.0	

4.2.1 コンピュータの CPU 使用率の評価

描画遅延が発生する原因として、描画頻度が CPU の描画処理能力を超えていることが考えられる。NetMeeting 利用時に、クライアント上で共有アプリケーションを起動した場合のサーバ/クライアントの CPU 使用率を図5に、サーバ上で共有アプリケーションを起動した場合のサーバ/クライアントの CPU 使用率を図6に示す。

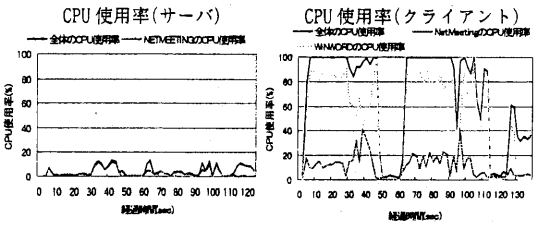

図5 クライアント上で共有アプリケーションを起動した場合の CPU 使用率

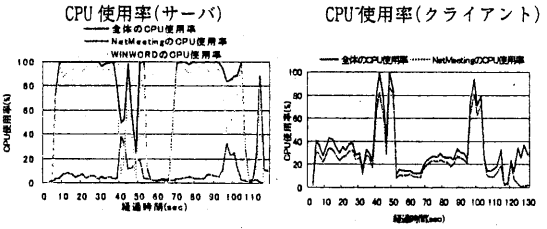

図6 サーバ上で共有アプリケーションを起動した場合の CPU 使用率

共有アプリケーションを起動しているコンピュータには入力データが集中するため、CPU 使用率が上昇する。図5、図6では共有アプリケーションを起動していないコンピュータの CPU 使用率も上昇している。これは NetMeeting が Windows メッセージをリアルタイムに送受信し、受信したメッセージの変換処理/共有アプリケーションウインドウへメッセージポスト処理のためである。特に図6では、クライアントの CPU のスペックが低いため、CPU 使用率がコンスタントに 20~40%を維持しており、CPU 使用率の上昇が画面再描画時に遅延が発生する原因の1つと考えられる。

ホワイトボードシステムで共有アプリケーションを起動した場合の CPU 使用率を図7に示す。

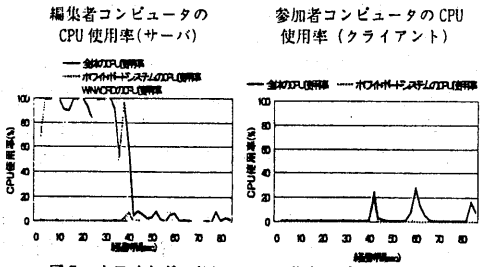

図7 ホワイトボードシステムで共有アプリケーションを起動した場合の CPU 使用率

サーバ上で 0～40 秒までデータ編集が実行されたため、サーバの CPU 使用率が上昇しているが、クライアントの CPU 使用率には影響がない。40 秒後のデータ編集終了時に変更データ転送が発生するため、一時的にクライアントの CPU 使用率が上昇するが、20～30%の CPU 使用率が数秒続くだけであり、レスポンスの悪化にはつながらない。

4.2.2 ネットワークトラフィックの評価

図 8 にホワイトボードシステムと NetMeeting で共有アプリケーション利用時の IP データグラム数の時間的推移を示す。

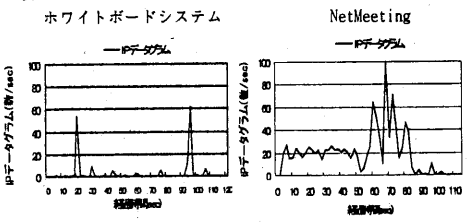

図 8　IP データグラム数の時間的推移

両システムともピーク時の IP データグラム数はそれ程多くない。しかし、ホワイトボードシステムでは 30、95 秒前後だけ通信が発生するのに対し、NetMeeting では断続的に通信が発生しているという違いがある。2 回だけ発生する同期通信は、データ編集開始/終了時やデータ整合性チェック時に発生する同期通信である。断続的な同期通信は画面描画頻度の上昇も引き起こすことからも、同期頻度が画面再描画時に遅延が発生する原因の 1 つと考えられる。ホワイトボードシステムはデータ編集開始/終了時だけホワイトボードサーバ/ホワイトボードクライアント間で同期通信を実行するため、断続的な同期通信は起こらない。

5.　まとめ

多くのリアルタイムプレゼンテーションシステムのアプリケーション共有機能には、マウスポインタ共有による個人作業の制限、ディスクアクセスの制限、共有アプリケーションウインドウ描画時の遅延、という操作性や性能面の制限がある。我々は共同作業空間と個人作業空間の分離、アプリケーションプロセス分散化、遅延同期処理による描画性能改善という方針での制限回避を実施した。

OLE 技術を利用し、データをオブジェクト化することで、共同作業空間と個人作業空間の分離を実現した。共有アプリケーションを各コンピュータ上で起動するという分散方式を利用することで、アプリケーションプロセス分散化を実現した。アクセス権を導入し、複数ユーザによるデータ同時編集を禁止することで、遅延同期処理が可能となり、描画性能改善を改善することができた。

今回はプレゼンテーションシステムの機能の中でアプリケーション共有機能に絞って検討を行い、ホワイトボードシステムの試作を行った。また、ホワイトボードシステムを評価するために NetMeeting との比較を実施した。NetMeeting 利用時とホワイトボードシステム利用時のコンピュータ CPU 使用率やネットワークトラフィックの測定結果を比較することで、ホワイトボードシステムの有効性を明らかにすることができた。

参考文献

[1]松下 温：グループウェア実現のために：情報処理, Vol. 34, No 8, pp. 984-993 (Jun. 1993)
[2]中村 達也, 横田 裕介, 垂水 浩幸, 上林彦：協調ハイパーメディアシステム VIEW Media におけるアクセス権を考慮した会議支援機能の提案：情報処理学会研究報告(97-GW-25), 25-5, pp.25-30(Apr. 1997)
[3]阿部 豊子, 前野 和俊, 阪田 史郎, 福岡 秀幸：マルチメディア分散在籍会議システム(MERMAID)を利用したグループアプリケーションの分散協調制御方式の提案：情報処理学会論文誌, Vol.34, No 6, pp. 1406-1415(Jun. 1993)
[4]中島 周 他5名:共有ウインドウと動画を用いた遠隔マルチメディアプレゼンテーションシステム:情報処理学会論文誌, Vol. 34, No 6, pp.1385-1394 (Jun.1993)
[5]田淵 仁浩, 大泉 俊雄：分散マルチメディア文書同時共有ミドルウェアのグループ学習への応用：情報処理学会研究報告(96-GW-17), 17-4, pp.19-24(Apr. 1996)
[6]角田 潤, 岡田 壮一, 渡辺 理, 浅見 俊宏：対面型会議の電子的支援(その 2)－グループオーサリングサービスの開発－：情報処理学会第52回全国大会, 6, 283-284(Mar. 1996)
[7]田頭 繁, 倉島 顕尚, 市村 重博, 水野 浩三, 前野 和俊：マルチメディア分散在籍会議システムにおける IP マルチキャストの適用：情報処理学会第 52 回全国大会, 6, 265-266(Mar. 1996)