マルチメディア通信と分散処理 60－8
（1993．5．14）

分散環境に適したマルチメディア制御モデル

米田 健　林 正薫 松下 温
慶応大学理工学部

さまざまな分散マルチメディアアプリケーションの共通基盤となるマルチメディア論理デバイスモデルの設計と実装について報告する.そのモデルでは,マイク,スピーカ,カメラ,ウインドウ,ファイル,ネットワークチャネル,オーディオミキサをすべて統一的に論理的なデバイスとして扱い,各デバイスを容易に生成,削除,結合することができる.デバイス同士は1対1,1対多,多対1,多対多と柔軟に結合することができる.またデバイス間の結合の際に必要となるマルチメディアネゴシエーションについても述べる.提案されるモデルを用いてプロトタイプ会議システムを構築し,マルチメディア論理デバイスモデルの有用性を確認した.

A Multimedia Control Model Suitable for Distributed Environment

Takeshi Yoneda Joung-Hoon Lim Yutaka Matsushita
Faculty of Science and Technology, Keio University
3-14-1 Hiyoshi, Kohoku-ku, Yokohama 223, Japan

To develop distributed multimedia applications easily and efficiently, a multimedia logical device model is proposed in which not only cameras, microphones and speakers but also files ,windows and network channels are uniformly treated as logical devices. According to the model ,a MMC(Multimedia Control) server which runs as a user-level process on a UNIX system is implemented. The MMC server provides logical device operation services such as creating, destroying, connecting, disconnecting, and using such device operation services, a prototype multimedia teleconferencing system is developed and it is proved that MMC can supports a wide range of distributed multimedia applications.

1. はじめに

分散マルチメディアアプリケーションを構築するためにはまずリアルタイムデータとノンリアルタイムデータを統合伝送できるマルチメディア統合LANが必要となる.そのようなLANとして,FDDI-I,II,ATM-LANが実用段階に入りつつある.また,複数のマルチメディア統合LANを経由してホスト-ホスト間で映像,音声通信を行うにはそれぞれの経由するネットワークであらかじめネットワークリソース(帯域,中継ノードのCPU,バッファ)を予約し,ホスト-ホスト間で帯域と遅延が保証される仕組みが必要となる.そのようなネットワーク資源の予約はSRP[1],ST-II[2]で扱われている.また広域網であるB-ISDNでもホスト-ホスト間で統合伝送が可能となるように設計されている.さらにリアルタイムデータ伝送に適したプロセス間通信のプロトコルとしてXTP[3], NETBLT[4], VMTP[5]が提案された.以上のように単一の伝送路によりリアルタイムデータをエンド-エンド間で伝送する仕組みは整いつつある.しかし,複数のメディアを関連付けて伝送したり,処理するといった分野の研究は最近になって行われるようになった.まず,時間的,空間的に関係のあるマルチメディア情報の構造としてはHyTime[6], MHEG[7], TDMD[8]などの提案がなされた.構造化されたマルチメディア情報を時間関係,空間関係を正しく反映させて表示するアプリケーションを構築するためにはスピーカや映像表示ウインドウ,音声,映像ファイルを統一的にソフトウエアで扱い制御する仕組が必要である.そのようなものとしてはDVI[9]があるがネットワーク上に分散した情報の制御までは考慮していない.そこで我々は統合ネットワークの使用を前提として構造化されたマルチメディア情報を表示,制御するシステム,テレビ電話,テレビ会議といった分散マルチメディアアプリケーションの基盤となるマルチメディア論理デバイスモデルの提案をする.提案されるモデルではマイク,スピーカ,カメラ,ウインドウ,ファイル,ネットワークチャネル,オーディオミキサをすべて統一的に論理的なデバイスとして扱い,それらを容易に生成,削除,結合することができる.デバイス間の結合は1対1,1対多,多対1,多対多という形態を柔軟にとることができる.

提案されるモデルに従いMMC(MultiMedia Control)サーバをワークステーション上に実装し,MMCサーバを用いてプロトタイプ会議システムを構築し,提案するマルチメディア論理デバイスモデルの有用性を確認した.

2.マルチメディア論理デバイスモデル

マルチメディア論理デバイスモデルは以下の点を目標として構築されている.

(A)1)物理デバイスであるマイク,スピーカ,カメラ,2)ウインドウシステムの提供する映像の表示に使用されるウインドウ,3)ファイルシステムの提供する音声,映像情報を内容に持つファイル,4)ソフトウエアにより実現されるオーディオミキサ,ビデオミキサ,5)情報の伝送路としてのチャネルをすべて論理的なデバイスとし,統一的なインタフェースによりそれらの論理的なデバイスを作成,削除,結合,制御できるようにする.

(B)作成したデバイスを結合する際に,デバイス間でやり取りされるメディアの品質のネゴシエーションを自動的に行えるようにする.

(C)柔軟なデバイスの結合を可能とする.つまり,1対1の結合だけでなく,多対1,1対多,多対多の結合を可能とする.

(D)蓄積型リアルタイムデータ処理における制御イベント(start, pause, resume, stop, etc)に対しての同期が可能であるようにする.

以下では(A),(C)に焦点をあてる.(B)については5節で述べる.(D)については現在検討中である.

2.1 マルチメディア論理デバイス

実際にデバイスがどのような処理を行うかを考えた場合,一般にマイク,カメラのような情報

の生成元(source)となるデバイスは生成した情報を書き込むバッファを所有し,スピーカ,ウインドウのように情報の表示先(sink)となるデバイスはどの情報生成元デバイスの所有するバッファの情報を読むかを示すポインタ(読み出しバッファポインタ)を持つ.そこで情報を書き込むバッファを所有するデバイスをsourceデバイス,読み出しバッファポインタを所有するデバイスをsinkデバイスとする.本モデルで規定されている12種類の論理デバイスは扱うメディア(音声/映像),情報の扱い方(source/sink)で表1のように分類できる.

表1 マルチメディア論理デバイスの分類

タイプ / メディア	source	sink	source &sink
音声	Mic Audio_FIile_Read	Speaker Audio_File_Write	Audio_Mixer
映像	Camera Video_File_Read	Window Video_File_Write	Video_Mixer
音声 or 映像	Channel_In	Channel_Out	

ファイルは情報が読み出される場合_Read,情報が書き込まれる場合_Writeが拡張子としてつけられている.複数の入力を受け付け,合成して一つの出力をするオーディオミキサ,ビデオミキサは複数読み出しバッファポインタを持ち,ミキシング後の情報を書き込むバッファを所有するのでsourceデバイスかつsinkデバイスである.またChannel_Outはローカルシステムの情報をネットワークに対して送出するデバイスで,Channel_Inはネットワークからの情報をローカルシステム内に取り込むデバイスである.したがって,Channel_Outは読み出しバッファポインタを持つという意味でsinkデバイスに分類され,Channel_Inはネットワークから受け取った情報を格納するバッファを所有するという意味でsourceデバイスに分類した.

2.2 論理デバイスに対する操作プリミティブ

2.1で示された全てのデバイスに対して以下の操作プリミティブが統一的に適用できる.

(1)CREATE	デバイスの作成
(2)DESTROY	デバイスの削除
(3)SET	デバイス属性の設定
(4)GET	デバイス属性の取得
(5)CONNECT	デバイスの結合
(6)DISCONNECT	デバイスの結合解除

2.3 デバイス間の柔軟な結合

CREATEプリミティブによって作成されたsource(source and sink)デバイスとsink(souce and sink)デバイスをCONNECTプリミティブによって結合することができる.

以下に結合の形態と具体例を述べる.

[1対1結合]

例はマイクからの音声がスピーカから出力されることを示している.

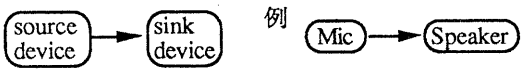


図1 デバイスの1対1結合

[1対多結合]

例はカメラからの映像がウインドウに表示され同時にファイルに保存されることを示している.

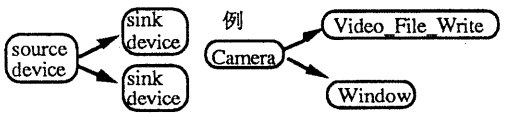


図2 デバイスの1対多接続

[多対1結合,多対多結合]

多対1,多対多結合の場合はsink and sourceデバイス(Audio Mixer,Video_Mixer)を介して,複数のsourceデバイスと,1つもしくは複数のsinkデバイスを結合する.

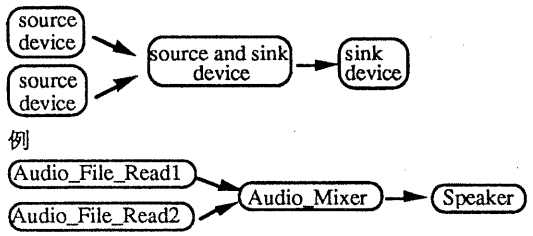
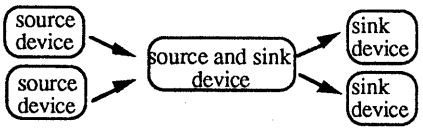


図3 デバイスの多対1結合



例

図4　デバイスの多対多結合

例では,複数の音声ファイルを合成してスピーカから出力する場合(図3)と,複数の映像ファイルを合成して,その結果をウインドウに表示しかつファイルに保存する場合(図4)を示している.

さらに上記の接続においてsinkデバイスにChannel_Inデバイスsourceデバイスに Channel_Outデバイスを用いることでネットワークを介したデバイスの接続が可能となる.

[ネットワークを介したデバイスの接続]

ホストAのsourceデバイスとホストBのsinkデバイスを結合する際には,図5のようにホストAでsourceデバイスとChannel_Outを,ホストBでChannel_Inとsinkデバイスを結合すればよい.

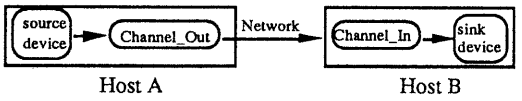


図5 ネットワークを介したデバイスの結合

3. 論理デバイスモデルの実装

3.1 MMC(MultiMedia Control)サーバ

実装形態として,マルチメディア論理デバイスの諸機能を提供するMMCサーバとその機能を利用するクライアントとから構成されるクライアント・サーバモデルを採用した(図6参照).

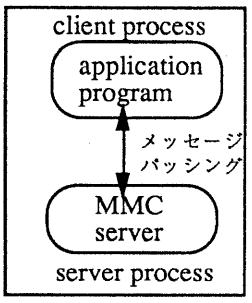


図6クライアント-サーバモデル

MMCサーバは3節(A),(C)を実現するserver processであり,論理デバイスの作成,削除,結合,制御といったサービスを提供する.Client processであるApplication programまたは遠隔ホストのMMCサーバはMMC libraryというC言語のインタフェースを介して,MMCサーバの提供するサービスを利用する.サーバ-クライアント間の通信はメッセージパッシングにより実現されている.

3.2 MMC library

MMC libraryはMMCサーバの提供するサービスをクライアントが利用するためのC言語によるインタフェースである.MMC libraryは,サーバとの接続関数とデバイス基本関数に大別される.

(1)MMCサーバとの接続関数

X-Windowシステムにおけるディスプレイ接続のように,MMCサービスを利用するApplication Programはあらかじめ MMCサーバと接続されなくてはならない.また,サービスの利用を終了した場合にはその接続を解除する.そのために以下の2つの関数が用意された

・int mmc_open(char *name)

nameはMMC serverの名前である.各ホストには1つのMMCサーバが存在することとし,そのホストの名前を MCサーバの名前とする.戻り値は接続識別子(session id)である.

・void mmc_close(int session_id)

MMCサーバとの接続を終了する.

(2)デバイス基本関数

デバイスの作成,削除,デバイス属性の設定,取得,デバイスの結合,結合解除といった基本的な操作のためにそれぞれに対応して

mmc_creata(),mmc_destroy() ,mmc_set() ,mmc_get(),mmc_connect(), mmc_disconnect()

が用意された.

・int mmc_create(dev_t device_type,0)

mmc_create()はデバイスの作成を行う.引き数device_typeにはビデオミキサを除く11種類の

デバイスを指定する.作成されたデバイスはデバイス管理テーブルに登録され,一意のデバイス識別が割り当てられる.戻り値はそのデバイス識別子である.複数の属性のリストを引き数として指定することができる.

・void mmc_destroy(int device_id)

mmc_create()で作成したデバイスを削除する.device_idには削除するデバイス識別子を指定する.

・int mmc_set(int device_id, mmc_attr attr_type, value, ...0)

作成したデバイスに対して,各種設定を行う.Attr_typeで属性名を指定し,valueで属性値を指定する.属性名と属性値の組は複数指定することができる.

・char *mmc_get(int device_id, mmc_attr attr_type)

生成したデバイスの属性値を得る.属性名はattr_typeで指定し,戻値は属性値のポインタである.属性によって属性値を格納する変数の型が決まっているので,対応する型に変換する必要がある.

・int mmc_accept(int session_id, int ch_num)

session_idはMMCサーバとの接続識別子であり,ch_numはChannel_Inと通信相手のChannel_Outとが共通に持つチャネル識別子である.Channel_Inは通信相手の存在するホストのMMCサーバからの作成要求によって自ホストに作られ,その際相手ホストのChannel_Outと共通のチャネル識別子(ch_num)が属性として設定される.mmc_accept()によりACCEPT要求を受けたサーバはデバイス管理テーブルを検索し,指定されたチャネル識別子を持つChannel_Inが存在する場合はそのChannel_Inのデバイス識別子を返す.

3.3プロトタイプ会議システム

実際にMMCサーバを用いてプロトタイプ会議システムを作成した.システムの構成を図7に示す.

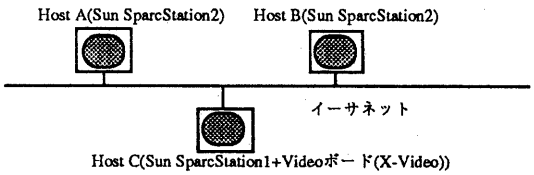


図7プロトタイプ会議システムの構成

イーサネットでつながれた3台のホスト上でMMCサーバを稼働させ,プロトタイプ会議アプリケーションをクライアントプロセスとしてそれぞれのホストで実行した.ホストA,BはそれぞれSun Sparc Station 2であり,ホストCはSun SparcStation 1にビデオボード(Parallax社X-video)を搭載したものである.ユーザインタフェースの構築には,X-Windowシステムのツールキット(XView)を用いた.図8にこのプロトタイプシステムで作成されたマルチメディア論理デバイスとそれらの結合を示す.ビデオボードを搭載しているのはホストCのみなので,ホストCのユーザの映像はホストA,Bに伝送することができるが, ホストA,Bのユーザは自分の映像を他のホストに送ることができない.そこで図8に示したようにホストA,Bのユーザの映像をあらかじめVideo_File_Read2, Video_file_Read1に保存しておき,ホストCでは,それら2つのファイルを, ホストAではVideo_File_Read1を,ホストBではVideo_File_Read2を再生してウインドウに表示した.

4. 問題点

プロトタイプシステム構築により,マルチメディア論理デバイスモデルを利用することで容易に分散マルチメディアアプリケーションが構築されることが確認された.また以下の様な問題点が生じた.

(1)CameraデバイスはX-Windowのウインドウに表示されている動画を定期的にキャプチャすることにより実現されている.その結果実用的な映像の転送レートを得ることができなかった.ビデオボードで圧縮した動画をそのままネットワークに送出し,受け側でその情報を伸長するように改良する必要がある.

(2)メディアごと通信相手ごとにChannel_Outデ

バイス,Channel_Inデバイスを作成しなくてはならず,LANの持つ同報機能,TCPの持つ全2重通信機能を有効に活用できていない.
(3)Unix上で実装したので厳密なリアルタイム性は保証されていない.
上記問題(1),(2)に関しては新たな同報機能,双方通信機能を持つネットワークチャネルデバイスの作成を検討している.また,(3)に関してはリアルタイムOSの利用を検討している.

5.マルチメディアネゴシエーション

テレビ電話やテレビ会議において音声,映像の情報が一般のアプリケーションと同様にユーザプロセスにより扱われることを考えると,映像音声の品質はシステムリソース(CPUやメモリ)とネットワークリソース(帯域,中継ノードのCPU,バッファ)によって制限されることになる.

したがって,エンドシステム内のシステムリソースレベルのネゴシエーションとネットワークリソースレベルでのネゴシエーションを分けて考える必要がある.また,複数のリアルタイム情報を同時に扱う場合には,それぞれのリアルタイム情報に要求される品質を満たすためにリアルタイム情報間(メディア間)でのネゴシエーションが必要になる.例えば,テレビ電話の場合,映像の品質を高めたために音声の品質が低くなるという状況が考えられる.そのような場合,映像の品質を許される範囲内で低くし,その分音声の品質を上げるということが考えられる.このように単一のメディアのネゴシエーションだけでなくメディア間のネゴシエーションも含めたネゴシエーションをマルチメディアネゴシエーションと呼ぶことにする.マルチメディアネゴシエーションでは,以下のことが必要となる.
(1)単一のリアルタイム情報に関して,システムリソースとネットワークのリソースの観点からネゴシエーションをする.
(2)ネゴシエーションに必要となる品質を定義する.
(3)定義された品質により品質の許容範囲をユーザに提示させる.
(3)複数のリアルタイム情報間でネゴシエーションする際に,どのようなポリシーでリアリタイム情報間の品質を変化させるか規定する.
(4)システム・ネットワークリソースの状況から自動的に決定される品質とユーザが指定することにより決定する品質音声のステレオ/モノラルなど)をわける.以下ではまず品質の定義について現在検討していることを述べる.

5.1 メディアの品質

(1)音声の品質
音声についてはネゴシエーションを効率的に行うために以下のような仮定を設け,図9のように1次元的に品質を表わせるようにする.
1)音声のbits/sampleは固定(ex. 16bits/sample)とする.サンプリングレートは離散的な値(ex.8kHz,22kHz,44KHz)をとる.
2)ステレオ、モノラルのどちらを希望するか,またどちらでもよいかはアプリケーションユーザが選択する.
3)単位時間あたりの生成情報量(圧縮される場合は圧縮前の情報量)を品質の基準とみなし、単位時間あたりの生成情報量が同一の場合のみ遅延の小さい方をより高品質と判断する.例では遅延は1,5,30msとしている.

ランク	rate(delay)
1	44kHz(1ms)
2	44kHz(5ms)
3	44kHz(30ms)
4	22kHz(1ms)
5	22kHz(5ms)
6	22kHz(30ms)
7	11kHz(1ms)
8	11kHz(5ms)
9	11kHz(30ms)

図9音声の品質

[映像の品質]
単一の映像品質に関しては[10]において時間的解像度と空間的解像度に分けて表現されている.我々はさらにピクセル当りのビット数も品質に含めることにした.つまり,映像の品質はframes/sec,framesize(width,hight),depth/pixselの3つのパラメータで決定するも

のとし,それぞれ離散的な値をとるものとする.つまり映像の品質は図10のように3次元で表現される.システムリソースに関してのネゴシエーションが終了後ネットワークとのネゴシエーションを行う場合,ネットワークの満たすことのできる遅延の条件からframes/secの上限が決定される.また帯域の要求にあわせるためにframes/sec, depth ,framesizeを変更する.

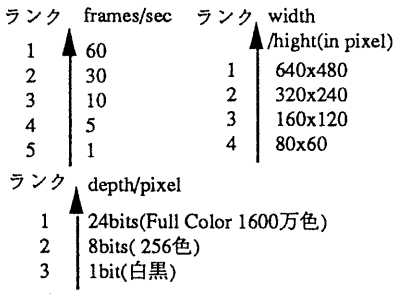


図10映像の品質

[複数の音声,複数の映像を利用する場合]

複数のメディアを利用する場合には,各メディアについての品質の1ランクの定義し,それぞれメディアの品質の許容範囲をユーザに提示させる.

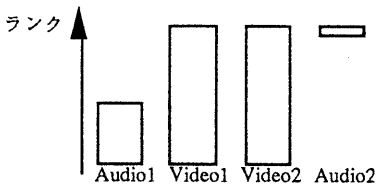


図11 複数メディアの許容範囲品質

例えば2つの音声,2つの映像を利用する場合,図11の様に品質の許容範囲をランクのmax値,mim値によりユーザに提示させる.min値として0が指定されている場合はその情報がなくてもよいことを示し,max値=min値の場合は品質はその値以外許されないことを示す.実際にはユーザにはもっと容易なインタフェースで品質を指定させそれを図11のように変換する必要がある.

例えば,図11のそれぞれの情報の品質を左側のものから満たしてゆくとする.その場合,Audio2, Video2, Video1とそれぞれmax値を満たすように品質が設定され,その結果Audio1の品質がmin値さえ満たせない状況になったとする.その場合,Audio1の品質がmin値を満たせるようにVideo1,Video2, Audio2の品質を下げるわけだがそのとき大きく2つの方針がある.

(1)既に品質の満たされているものを均等に1ランクずつさげてゆく.

(2)既に品質の満たされているものを順々に(例えば図の左側から)min値まで品質を下げてゆく.

以上の検討では,品質のランクを上げ下げすることでどの程度CPUの負荷,メモリの使用量,帯域の使用量が変るかを正確に判断できることが前提となっている.

6.むすび

マルチメディア分散アプリケーションの共通基盤としてマルチメディア論理デバイスモデルを提案しそのモデルに従いMMC(MultiMedia Control)サーバをワークステーション上に実装した.MMCサーバを用いてプロトタイプ会議システムを構築しMMCサーバの有用性を確認した.

5節で述べたマルチメディアネゴシエーションはまだ検討段階であるので今後そのメカニズムをより明確にし,マルチメディア論理デバイスモデルのデバイス間の結合の際に利用する予定である.また,マルチメディアネゴシエーションにより,複数の同期すべき音声,映像のframes/sec値が決定するが,その値を利用してマルチメディア同期のメカニズムを構築する予定である.さらに,大量の映像情報をトランスポート層のプロトコルを利用して転送した場合にどれくらいの時間で転送できるのかを正確に把握することはネゴシエーションの一つの判断材料になると思われるので(例えば,リモートにある映像情報をローカルに全て持ってきてから再生するのか,リアルタイムのチャネルをそのリモートとの間に確保してその映像を再生するのかなど)その評価をする予定である.

参考文献

[1]D.P.Anderson,"Meta-Scheduling for Distributed Continuous Media",Technical report UCB/CSD 90/599, University of California, Berkeley, (Oct.1990).

[2]C.Topolcic,"Experimental Internet Stream Protocol, Version2(ST-II)",RFC-1190, (Oct.1990).

[3]G.Chesson,"XTP/PE Overview",Proc.13th Conference of Local Computer Networks, (Oct.1988).

[4]D.Clark, et al. "NETBLT: A High Throughput Transport Protocol", ACM SIGCOMM'88, pp.353-359,(1988).

[5]D.Cheriton,"VMTP: A Transport Protocol for the Next Generation of Communication Systems"ACM SIGCOMM'86,(1986).

[6] ISO/IEC JTC1/SC18/WG3, HyTime Review, (Jan. 1991).

[7]ISO/IEC JTC1/SC2/WG12,Multimedia and Hypermedia Information Coded Representation, MHEG Working Document Version 1, (Jun. 1990).

[8]T.Yoneda,Y. Matsushita,"A New Communication Tool: Time Dependent Multimedia Document", Proceeding of the 12th IEEE Distributed Comuputing System, pp.90-97,(1992).

[9]J.L.Green, The Evolution of DVI System Software, Communication of the ACM, Vol.35,No.1, pp.53-67,(Jan.1992).

[10]T.Tokuda at al,"Continuous Media Communication with Dynamic QOS Control Using ARTS with an FDDI Network",ACM SIGCOMM'92,pp.88-pp.98,(Aug.1992).

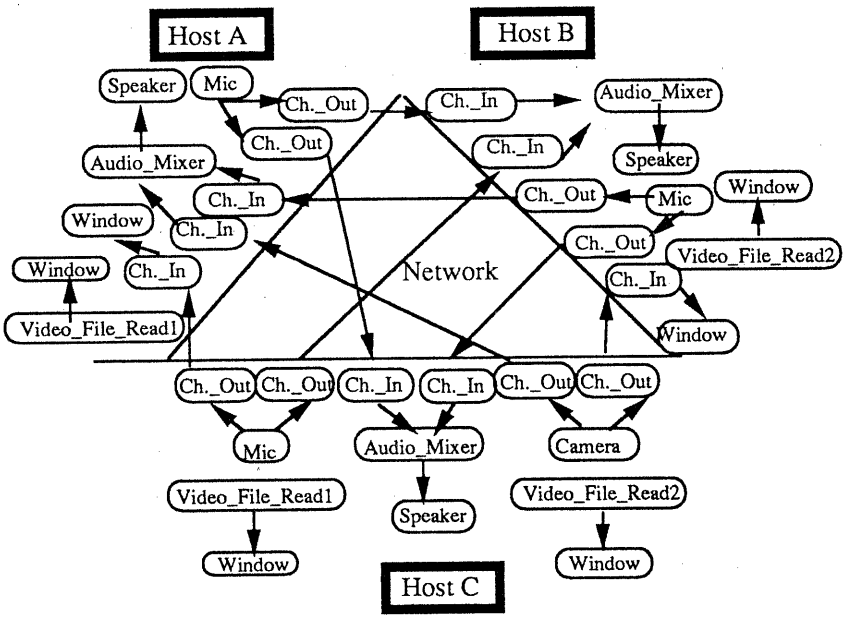


図8プロトタイプ会議システムで用いられたマルチメディア論理デバイス