修士論文　　2004年度 (平成16年度)

フォーマット非依存性を考慮した映像伝送支援技術の研究

慶應義塾大学大学院 政策・メディア研究科

入野 仁志

修士論文要旨 2004 年度 (平成 16 年度)

フォーマット非依存性を考慮した映像伝送支援技術の研究

本研究はインターネットを基盤とした映像転送システムに関する研究である．特に，映像機器が扱うデータフォーマットを中間的なフォーマットに変換することなく，映像転送を行うシステムに着目した．Windows Media Technology に代表される既存の映像転送システムは，中間的なフォーマットに変換を行い転送を行っている．しかしこれらの中間的なフォーマットの利用は，そのフォーマットの変換処理にかかる時間から，転送時に遅延が大きくなる問題がある．一方，映像機器で扱われるデータフォーマットは複数存在し，かつ個々の特徴が異なる．そのため中間的なフォーマットを利用しないシステムは，特定のフォーマットと特定の入出力や通信方法の組合せ毎に実現されてきた．そのため，それらの映像転送システムは，利用者に対して，複数の異なった入出力や通信方法の同時利用を提供できないといった機能的な制限があった．また，扱うフォーマットと通信方法の組合せ毎の個別システムは機能の拡張性に乏しかった．本研究は映像機器の扱うフォーマットの類似したデータ構造に着目し，そのフォーマット間の共通部分と非共通部分の分離を行い，さらにフォーマットが入出力に対して依存する部分と依存しない部分の分離を行うことで，これらの問題を解決した．本研究の成果により，中間的なフォーマットを利用しない映像転送システムの開発者に対して，機能拡張を簡易化した．また，これらの映像転送システムの利用者に対しては，複数の異なった入出力やその通信方法をデータフォーマットに依存なく同時に利用するといった自由度の高い通信を提供可能にした．

キーワード
1. 映像転送システム, 2. データフォーマット, 3. 異種入出力,
4. 拡張性, 5. フォーマット非依存性

慶應義塾大学大学院 政策・メディア研究科

入野 仁志

Abstract of Master's Thesis

Academic Year 2004

Research on Supporting Video Transport System with Format Independency

This research focuses on Internet-based video transmit systems. In particular, this video transmit system takes the data handled by video appliances and transmits the data without converting it into an interim format. Previous video transmit systems such as the Windows Media Technology convert video data into a specific interim format before transmitting it. However, such interim conversion process takes time, and causes a delay when transmitting data. The downside of not using such interim conversion format is that several different types of data format exist for video appliances, and compatibility between the different formats has been achieved by combining the specific format with the specific I/O. For that reason, a functional problem occurs where the user is not able to concurrently use multiple video transmit systems with conflicting I/O and transmission formats. Because the combined format and I/O each became an individual system, expand-ability and compatibility between the different systems have been scarce. This research approaches this problem by focusing on the similarity between the data structure formats of different video appliances. This research attempts to solve the problem by separating the common and uncommon characteristics in the different formats as well as separating the dependency and non-dependency for I/O. This research has provided developers that use the non-conversion video transmit system to easily develop additional enhancement features. In addition, this research allows users flexible communication between multiple conflicting I/O, without being tied to the data format.

Keywords :
1. Video Transport System, 2. Data Format, 3. Various I/O,
4. Extendability, 5. Format Independency

Keio University Graduate School of Media and Governance

Hitoshi Irino

目 次

図 目 次

表 目 次

第1章　序論

1.1　背景:デジタル映像機器とインターネットにおける映像転送技術の進化

通信基盤としてのインターネットの普及及び，インターネットに利用される回線の広帯域化により，大容量の映像，音声の転送が多く利用されるようになった．現在，Windows Media Technology[1] に代表される多くの映像転送技術が狭帯域環境下での配信を可能とするために，高圧縮な符号化を用いたデータフォーマットを利用している．その一方で広帯域環境下での利用を前提とした映像転送技術の中には，高品質映像転送を目的として，計算機と接続可能なデジタル映像機器から入力されるデータを他のフォーマットに変換することなく転送する映像転送技術も存在する．フォーマット変換を行わない映像転送システムとして，DVTS(Digital Video Transport System)[2][3], DV VoD[4], MPEG2-TS 転送システム [5] などが実装されている．その一部は標準化され，実用アプリケーションとして一般に公開されている．

これらの既存研究や実用化された映像転送システムは，映像機器間の映像転送をインターネットを介して実現している．しかし，映像機器で用いられるフォーマットは複数存在するために，既存研究や実用化された映像転送システムは，フォーマットと入出力機器や通信方法の組合せ毎に個別に存在している．個別に存在するこれらのシステムの利用者は入出力対象及び通信方法とフォーマットの組合せ毎に，個別のシステムを使い分ける必要がある．そのため複数の異なった入出力の同時利用が不可能であるなど利用時の機能面における問題がある．これらの特定の入出力の対象及び通信方法とフォーマットの組合せを前提にした個別的なシステムは，拡張性に乏しく，新たなフォーマットや新たな通信方法に追従できないなどの問題がある．

1.2　本研究の目的

本研究はフォーマット変換を行わない映像転送システムにおける，複数の異種入出力の同時利用の実現と，拡張性の確保の実現を目的とする．本研究では映像機器が扱う複数種のフォーマットの共通部分の定義，及び対象とすべき入出力における共通部分の定義を行う．個別実現がされているフォーマット変換を行わない映像転送システムの入出力に依存した処理とフォーマットに依存した処理の分離を行うことで，システムの拡張性を確保する．また複数の異なった入出力や通信方法を同時に利用することのできる自由度の高い通信を実現する．

1.3 本論文の構成

本論文は7章から構成される．本論文は第2章において，ストリーミングの分類，本研究が対象とするデジタル映像機器で利用されているデータフォーマットを他の中間的なフォーマットに変換することなく利用する映像転送技術のメリットを述べる．その上で現状のフォーマットを変換することなく利用する映像転送技術に基づいて作られている映像転送システムに関して述べる．第3章において，それらの映像転送システムが個別的に作られていることを問題として述べ，本研究の基本的アプローチとして，対象とする入出力や通信方法とフォーマットの機能分離の必要性を述べる．第4章において，中間的なフォーマットに変換することなく利用する映像転送技術における対象とする入出力や通信方法とフォーマットの機能分離と，それらの一律的な操作を提供する機構の設計を行う．第5章において，その設計に対する実装に関して述べる．第6章において本研究の実現する入出力処理とフォーマットを分離する機構の性能の定量評価と機能の定性評価を行い，最後に第7章で本研究のまとめを行う．

第2章　IPネットワーク上での映像転送技術の分類とその適応範囲

本章では本研究が前提とするIPネットワーク上の映像転送技術を，計算機上でのデータフォーマットの変換の有無で分類する．また通信路として用いられるIPネットワークにおけるプロトコルと用いられる技術に関して，その適応範囲を述べる．これらに基づいて本研究が対象とする映像機器で用いられるデータフォーマットを利用した映像転送の優位点を述べる．

2.1　IPネットワークの特徴と映像転送

通信路としての役割を果たすインターネットは，IP(Internet Protocol)[6] を利用した広域分散ネットワークである．IPはコネクションレスのデータグラム転送を行うプロトコルであり，IPネットワークではデータは，IPパケットにカプセル化され，パケット交換方式によって中継される．IPネットワークでは，帯域幅，IPパケットの伝播遅延，到達性，完全性は保証されない．帯域幅，伝搬遅延などの特性は下位層から影響を受ける．またIPで保証されない到達性や完全性は用途に応じて上位層でその機能を提供する必要がある．本節ではIPを用いた通信において，映像転送技術と関係のある，現在一般に広く利用される技術，及び新しい技術に関して述べる．

2.1.1　映像転送技術と関係のあるIPを用いた通信における一般に利用される技術

本項では映像転送技術と関係のあるIPを用いた通信における一般に利用される技術に関して，トランスポート層におけるプロトコル，セッション層におけるプロトコル，アプリケーションで用いられる技術を述べる．

トランスポート層で用いられるプロトコルと映像転送技術との関係

トランスポート層にはUDP(User Datagram Protocol)[7] とTCP(Transmission Control Protocl)[8] がある．TCPは信頼性のある通信を実現するために輻輳制御・再送制御を自律的に行う．一方のUDPはIPの特徴を引き継ぎ，到達性やデータの完全性を保証しない．TCPでは輻輳制御による意図しないスループットの低下と，確認応答を用いた再送

制御による遅延が起こる．TCP は輻輳制御・再送制御を自律的に行うが，それが上位層のアプリケーションで実現される映像転送において悪影響を与える．映像転送用途で特に遅延を重要視する場合は TCP より UDP が用いられる．UDP を用いた場合は，さらに上位層で必要に応じて到着順序などの情報の検知を行う機構が必要とされる．

RTP/RTCP と映像転送における適応範囲

UDP は TCP と異なりトランスポート層におけるパケットの到着順序は保証しない．パケットの到着順序や欠損の状況を判別するためにはより上位層での対応が必要となる．RTP (Real-time Transport Protocol)・RTCP(Real-time Transport Control Protocol)[9][10] が到着順序の把握と欠損の状況を始めとした品質情報の交換を行うためのプロトコルとして用いられる．

RTP は映像・音声を始めとした実時間性のあるデータの転送に利用する目的で，下位のトランスポートプロトコルとは独立に設計されているプロトコルである．RTP は順序番号やタイムスタンプ等の情報を，通信する映像音声情報のデータに付加する．この情報を用いることで受信側は正しいパケットの順序及びパケットの欠損状態を検知出来る．RTCP は RTP と同じく標準化されており，RTP にさらに付加的な情報を与え，RTP を用いた通信の管理を行うためのプロトコルである．

RTP 及び RTCP は多くの映像転送アプリケーション，特に UDP を用いる必要があるアプリケーションにおいて利用されている．

しかし，RTP 及び RTCP は状態の検知に用いられるのみで，パケットの到達性や到着順序自体は保証しない．従って，実際の通信においては検知した情報をもとに，別途アプリケーションで通信を制御する必要がある．

映像転送アプリケーションでの技術

映像転送アプリケーションでは，通信を効率的に行うことと，通信品質のばらつきを吸収することが行われる．

前者の一例としては，環境に応じてデータサイズを変更可能な機能が挙げられる．利用可能な帯域を越えた通信やデータリンクの MTU を越えた通信は非効率となる．インターネット上には様々な計算機がつながっており，その通信環境及びその通信品質は多様性に富むため，これらの制御に必要な情報は，映像音声転送先毎に保持する必要がある．

後者の一例がバッファリングである．バッファリングは一定量のデータを一時的に受信側で蓄積し，蓄積されたデータから消費する方法である．バッファリングはジッタと呼ばれるパケットの到着時間の揺らぎの吸収と，パケット順序の不整合性の解消や欠損の回復に用いられる．

バッファリングは一時的にデータを蓄積するため遅延と大きく関係する．バッファリングに用いるデータ蓄積量を増加させると，遅延時間が大きくなる．遅延を重視するアプリケーションではバッファリングを最小限に留める必要がある．

2.1.2 映像転送技術と関係のある IP を用いた通信における新しい技術

本節では，標準化され普及段階にある新しいトランスポート層プロトコルとセッション層プロトコルに関して述べる．また新しい一斉同報配信技術に関して述べる．

新しいトランスポート層プロトコル

現在一般的に IP ネットワーク上で用いられるトランスポートプロトコルは TCP と UDP である．TCP は既に述べた通り自律的な輻輳制御や再送制御が転送している映像音声情報に対して悪影響を与える．そのため，映像音声情報の転送用途には限定的に利用されている．一方の UDP は輻輳制御を持たないため，データリンクを共有する IP を用いた通信において，通信の不平等が問題となる．これらの問題を解決する可能性を持つ新しいトランスポートプロトコルが SCTP(Stream Control Transmission Protocol)[11] である．SCTP は TCP と同様に輻輳制御を行うプロトコルであるが，TCP にはない実時間情報の通信を考慮に入れた以下の特徴をもつ．

- message oriented framing
  データをメッセージの境界で分割できる．TCP はバイト単位であるため構造を持たない．

- multi streaming
  データを複数のストリームに分割することができる．あるストリームのデータ欠損や遅延は他のストリームに影響を与えない．

SCTP は既に Linux Kernel 2.6 でサポートされているなど，普及する初期段階にある．今後の映像転送技術において利用想定環境や状況に応じて利用されるプロトコルである．

新しいセッション層プロトコル

IP ネットワーク上における実時間情報の通信において，RTP が一般的であることは述べた．その通信状態を通知する為のプロトコルとして RTCP が用いられる．RTCP はこれまで SR(Sender Report), RR(Receiver Report) の 2 種類のメッセージの交換による品質通知を行っていた．VoIP(Voice over IP) の普及からその品質管理用途により細かい品質制御が求められ，その要求に答えるために新たに RTCP XR(Extended Report)[12] が標準化された．RTCP XR は VoIP の品質管理がその目的の一つであるが，交換される情報は一般的な品質を表す情報であり，実時間情報を送受するアプリケーション一般に利用できる．このプロトコルも今後の映像転送技術において利用想定や状況に応じては利用される．

新しい一斉同報配信技術

IP ネットワークにおける一斉同報配信技術では IP のプロトコルスペックとして IP マルチキャストがある．しかし，IP マルチキャストは，マルチキャストアドレスの予約の難しさ，IP マルチキャスト未対応機器の存在などの問題から，ISP 内での特定のサービスの利用に使用される程度に留まっている．

IP マルチキャストが一般的に十分に利用できない [13] ことから配送をアプリケーション層で行うオーバレイマルチキャスト技術に関する研究が盛んに行われている [14][15][16]．オーバレイマルチキャストは IP マルチキャストにおいてルータが果たしていた機能を各計算機が果たすことで実現される．これまで受信側として機能してきた計算機が同時に送信側の役割を果たし，中継の役目を果たす必要がある．

この技術も今後の映像転送技術において利用想定や状況に応じては利用される．ただし，データの送信・受信に特化し，中継の概念を持たない既存のアプリケーションはこの技術を利用できない可能性がある．

2.2 フォーマット変換の有無による映像転送技術の分類

映像・音声のデジタル化は対象の情報をある一定の時間間隔で標本化し，その情報を複数の階調に分解して量子化が行われる．より細かい標本間隔，量子間隔でデジタル符号化するとデジタル化された情報の品質は高くなる．しかし，非圧縮データの場合，データ量は標本間隔と量子間隔の積となるため，標本間隔，量子間隔が細かいほどデータ量は増大する．そのため，映像・音声情報用途の圧縮符号化技術がこれまでに多く研究され，実用化されてきた．

本論ではこの圧縮技術の利用と計算機上の変換の有無を軸として IP ネットワーク上の映像転送技術を分類する．本論では映像転送技術大きく分けて以下の 2 つに分類する．

- 独自の中間的なフォーマットを利用した映像転送技術
  映像機器からのデータを，別のフォーマットに変換し転送を行う．Windows Media Technology[1] や Real Media[17] など，ネットワーク配信に特化し変換を行うことで映像機器などと互換性を保つフォーマットを本論では中間的なフォーマットと表現する．

- デジタル映像機器で用いられるフォーマットを変換することなく利用する映像転送技術
  計算機をデータ転送機器として利用し，デジタル映像機器を撮影及び視聴機器として利用することで実現する．

上述の 2 種類の映像転送技術においてその概要をそれぞれ，図 2.1，図 2.2 に示す．

図 2.1 における送信側の計算機は映像機器から入力されたデータをあるデータ量の単位毎に中間的なフォーマットに変換・圧縮し，その変換後に，受信側の計算機に対して送信する．受信側は計算機で復号化する．図 2.2 における送信側の計算機は映像機器で入力さ

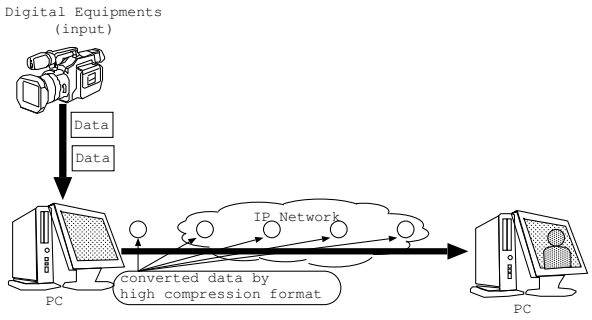

図 2.1: 独自の中間的なフォーマットを利用した映像転送技術

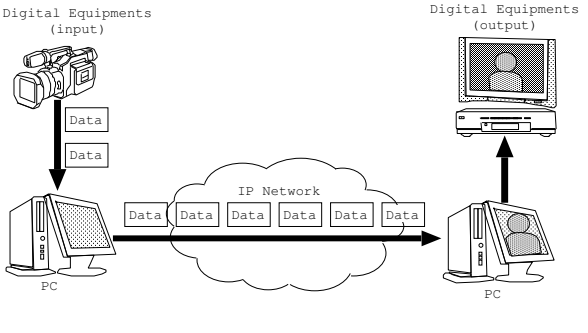

図 2.2: デジタル映像機器で用いられるフォーマットを変換することなく利用する映像転送技術

れたデータを無変換で受信側の計算機に送信する．受信側は計算機で受信データの復号化を行うか，受信データを映像機器に送信して映像機器でデータの復号化を行う．受信側の利用者は計算機及び映像機器で試聴が可能である．図 2.1 で示した対象とした独自の中間的なフォーマットを利用した映像転送技術でも受信側において映像機器へのデータ送信は可能である．しかし中間的なフォーマットから映像機器が用いるフォーマットへの変換を必要とする．

本論で 2 種類に分類した映像転送技術はその特徴から利用される通信形態が異なる．

映像転送の通信形態を分類すると以下の 2 種類に分類される．

- 双方向通信
- 単方向通信
  - リアルタイム型配信
  - オンデマンド型配信

中間的なフォーマットを利用した映像転送技術は，狭帯域での利用を可能とするために高圧縮な CODEC を用いたフォーマットを利用する傾向にある．それらのフォーマットへの変換処理に時間を要するため，計算機上での符号化・復号化遅延が必ず存在する．そのためこの方法に基づいたアプリケーションはオンデマンド型配信など，映像生成から映像消費までの時間差に対して寛容な用途に適している．

それとは対照的にデジタル映像機器で用いられるデータフォーマットを無変換で転送する映像転送技術は一般的に大容量のデータ転送になり，広帯域なネットワークでの利用が対象となる．一般にデジタル映像機器は大容量の記憶メディアの利用を前提としている．デジタル映像機器に用いられるフォーマットは比較的低圧縮なデータ量の大きいフォーマットとなる．あらかじめ広帯域環境を前提にして，これらのフォーマットを利用すると，変換を原因とした遅延がなく，また映像機器を用いた符号化・復号化が可能になるために符号化・復号化遅延も小さく抑えることが出来る．テレビ会議を始めとした双方向通信，

またはリアルタイム配信の利用に非常に適している．利用例はその低遅延を活かした遠隔医療，遠隔での映像・音声を用いた共同作業，共同演奏など多様である．

表 2.1 に上述の 2 種類の映像転送技術の分類の特徴の違いを示す．

表 2.1: 計算機上での変換を要する映像転送技術と要さない映像転送技術の比較

分類	必要帯域	遅延	画質	符号化/復号化
フォーマット変換有	小さい	大きい	QCIF〜HDTV	ソフトウェア
フォーマット変換無	大きい	小さい	SDTV〜HDTV	ハードウェア

表 2.1 はそれぞれの技術の特徴の違いを示している．上段は，中間的フォーマットを利用した既存の映像転送技術を示している．下段は映像機器のデータフォーマットを変換することなく利用する映像転送技術を示している．両者の大きな相違点は符号化・復号化を行う部分である．

伝送路となる IP ネットワークに使われるデータリンクは広帯域化しているため，今後映像機器のデータフォーマットを変換することなく利用する映像転送技術の利用頻度と重要性が増す．

2.3 映像機器のフォーマットを無変換でデータ転送する映像転送の既存技術及び研究

本節では広帯域環境下で用いられる，デジタル映像機器のフォーマットを無変換でデータ転送する既存技術及び研究に関して述べる．フォーマットを無変換でデータを転送する映像転送システムでは，映像機器が扱うデータフォーマットが複数あり，さらにそれらのデータフォーマットが入出力に依存する場合があるため，特定の入出力及び通信方式とフォーマットの組合せ毎に個別の設計に基づいて実現している．そのためそれぞれおいて機能差異がある．

2.3.1 DV over IP

本項では一般化したデジタル映像機器である DV で用いられる DV フォーマットの映像音声情報を IP を介して転送する技術に関して述べる．

DVTS

DVTS はデジタル民生用映像機器で用いられているデータフォーマットを変換することなく利用する映像転送技術である．DVTS は 1998 年から研究開発がなされており，IEEE1394[18] インタフェースを用いた民生用として利用されている SD(Standard Definition) 仕様の

DV[19] 機器を映像入出力の対象としている．従って DVTS が利用するフォーマットは SD 仕様の DV フォーマットである．送信側計算機は DV 機器から IEEE1394 インタフェース経由で DV フォーマットのデータ入力を受け，そのデータに対し IP データグラム化及び RTP 化を行い，転送を行う．その RTP ペイロードフォーマットは RFC3189[20], RFC3190[21] において標準化されている．現在，DVTS の実装は一般に公開されている．

標準化されていない独自の機能としてはフレーム内圧縮の特徴を利用したフレーム棄却による使用帯域の削減がある．図 2.3 に DVTS における画像フレーム棄却率と利用帯域の関係を示す.

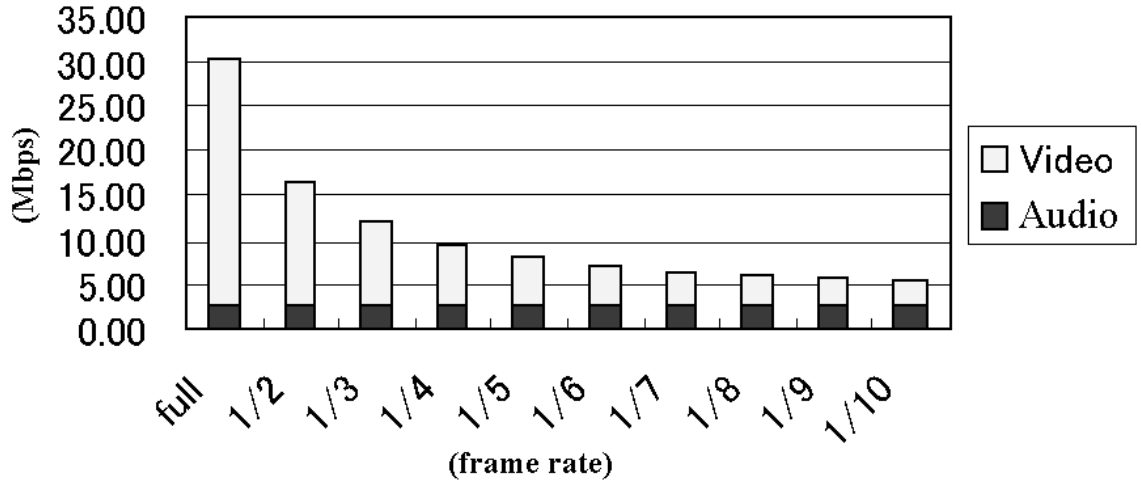

図 2.3: DVTS における画像フレーム棄却と利用帯域の関係

図 2.3 は縦軸に帯域，横軸にフルレート時のフレーム数に対して利用したフレーム数の割合を表している．映像フレームの棄却により使用帯域が抑えられている．このフレーム棄却の機能を用いて，TCP との親和性を高めた通信を行う研究が行われている [22].

計算機内においての符号化，復合化が一切なく，それらの処理を映像機器で行うため，非常に低遅延な通信が可能である．そのため特に実時間性が特に重要視される遠隔 TV 会議，映像中継に用いられる．遠隔 TV 会議等に用いた場合にコミュニケーションにおいて映像情報よりも音声情報が重用度が高い．DVTS は映像と音声の分離機能を持つ．この機能と優先制御技術を利用して分離した音声情報だけを優先させることができる．また DVTS 自体にも，同一の音声のデータを複数回送信することで音声情報の冗長化を可能にする機能がある [23].

DVBS:DV VoD

DVBS[4] は二次記憶装置に蓄積された DV データを転送するシステムで，放送品質の映像の一斉配信を目的としている．この研究では，以下の 2 つの制限事項を前提条件にしている．

- 一斉配信の実現のための IP マルチキャストの採用
  IP マルチキャストが前提のため，個々のクライアント毎の送信制御が難しい．
- DV の採用
  大容量フォーマットである DV を用いた場合，受信側でのバッファの確保が十分に期待できない．

この前提条件の下で，送信側での二次記憶装置からのデータ読み取りと受信側でのデータ消費量の速度差から発生する，有限バッファ問題 [24] に対してデータ送信側駆動のデータ送信間隔と送信データ量の制御を行うことで解決を計っている．

2.3.2 MPEG2 over IP

本項ではデジタル放送などで用いられる MPEG2-TS の映像音声情報を IP を介して転送する技術に関して述べる．

FEC を用いた MPEG2 over IP システム

広島大学で開発されている robst(Robust Streaming Tools)[25] は MPEG2-TS に対して FEC(Forward Error Collection) を用いて欠損率の高い通信環境においても映像品質を保持することを目的とした研究である．

このシステムはフレーム内圧縮フォーマットに比べて，データ欠損に対する情報欠損の割合が大きいフレーム間圧縮を利用している MPEG2-TS に対して，FEC を適用することで，その情報欠損を低減している．冗長符号化に対してパケット損失に対して有効とされる Reed Solomon を用いている．RFC ではパリティ符号 FEC[26] しか定義されていないため一部 RTP ペイロードフォーマットを拡張している．この研究，及び公開されている実装では，受信側は計算機に組み込んだ MPEG2 の特定のハードウェアデコーダを用いることを前提にしている．

民生用機器を用いた MPEG2-TS over IP

MPEG2-TS 転送システム [5] は DVTS と同様に送受ともに民生の映像機器を用いて，映像転送を MPEG2-TS を利用した機器間で行うシステムである．DV と比較して MPEG2 は複数の画質及びデータ量が規定されているため，IEEE1394 に対してのデータ量の調節を行う必要性がある．論文では RFC2250[27] に準拠した方法と IEEE1394 アイソクロナスパケットの情報をそのまま利用する方法が提示されている．

2.4 本章のまとめ

本章では，計算機上のフォーマット変換の有無を軸に既存の映像転送技術を 2 種類に分類した．映像機器で用いられるフォーマットを無変換で転送する映像転送技術は遅延を重視する環境を中心に利用される．現在，フォーマットを無変換で転送する映像転送技術の一部は実装が一般に公開され，実用的に利用されている．

第3章　入出力とフォーマットの自由な組み合わせを実現する機構の提案

3.1　映像機器のフォーマットを利用した映像転送における問題

第 2.3 項で述べた技術及び研究は基本的に

- 対象とする入出力，またはその通信方法
- 対象とするデータフォーマット

の 2 点以外はほぼ同一である．しかし，これらは個別的に実現されている．

特定の入力・出力と単一のフォーマットの組合せ毎に個別設計・実装が行われると， 機能性と拡張性において問題がある．

- 機能性の問題:異種入出力の利用
  映像転送システムが個別的に単一の入出力とフォーマットの組合せに特化すると，映像転送システムの利用者は状況に応じてシステムを使い分ける必要がある．さらに複数異種入出力の同時利用が不可能になる．例えば入力されたデータをネットワークとファイルに同時に出力することが第 2.3 項で述べたシステムでは不可能である．映像中継など再現が難しい実時間情報を扱う場合は，二次利用の為の蓄積と転送など異なった目的を同時に満たす必要がある．また，オーバレイマルチキャストが一般化したときに，試聴の為の映像機器へのデータ転送とオーバレイマルチキャストに参加する計算機への転送を実現する必要がある．利用用途の多様化への対応には，異種の入出力や通信方法に対して同時に入出力処理が可能な必要がある．

- 拡張性の問題
  入出力や通信方法とフォーマットの組合せが固定されているシステムは，新たな入出力や通信方法，または新たなフォーマットの登場などの技術的な進歩に対して追従しにくい．新しい入出力インタフェースの登場や，第 2.1.2 項で述べた新しいプロトコルなど，新たに登場した技術に対応する場合，入出力や通信方法とフォーマットの組合せが固定されているシステムでは，それぞれの対となる入出力や通信方法と扱うフォーマット毎に個別対応が必要となる．システムの個別化は映像転送システムの開発者に対して，開発量を増大させる原因となる．

特定の入力・出力と単一のフォーマットの組合せ毎の個別的な実現は問題があるため，入力・出力，フォーマットが自由に組み合わせ可能な機構が求められる．

3.2 入出力とフォーマットの自由な組み合わせを実現する機構の提案

第 3.1 項で述べた問題は，入力，出力，フォーマットが個々のシステム内で依存関係を持っていることに起因する．問題の解決には，これまで実現されて来た映像転送機構の機能を入力・出力・フォーマットの 3 項を分解し，複数の入力と出力の組合せをフォーマットに依存することなく自由に組み合わせ可能な機構が必要とされる．

図 3.1 に機能分解を行い複数の入出力を組み合わす概念図を示す．

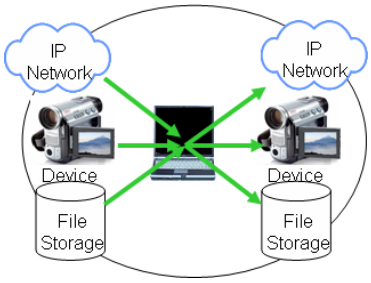

図 3.1: 入出力機能の分解と組合せ

各計算機がフォーマットに依存なく入力・出力・フォーマットを利用時に変更かつ組合せ可能になると，各計算機の入出力の組合せの連続によって，複合的な通信が可能となる．例えば転送された映像の受信及び視聴を行いながら，他の計算機への中継が可能となる．また，映像機器から入力された映像を他の計算機に送信しながら二次利用のために二次記憶装置に保存し，時間が経過した後に，二次記憶装置から映像転送を行うといった通信が可能になる．

機能要件

本研究で提案する機構に求められる機能は 3.1 で述べた問題から以下の 3 事項となる．

- フォーマットに依存した部分と入出力処理に依存した部分が分離され，それら 2 者間の依存性が低減されていること
- 異種の入出力が同時に利用可能であること
- 拡張が容易であること

さらに 既存の映像機器のデータフォーマットを無変換で転送する映像転送技術では以下の 3 つの事項が有用な機能として実現されて来た．以下に述べる 3 つの事項は本機構の設計時における機能要件とする．

- 映像・音声の分離・合成が可能であること
- 品質の変更が可能であること．
- データの冗長化が可能であること
  - 同一データの多重化による冗長化
  - 符号化による冗長化

3.2.1 既存の入出力の自由な組合せを可能とする技術及び研究

本項では既存の複数の入出力，複数のフォーマットに対応して，汎用的に扱うことを目指した技術や研究に関して述べる．

既存研究:VuSystem

VuSystem[28] は MIT で開発された連続メディアアプリーケーションの開発のためのプログラミングシステムである．この研究ではマルチメディアデータを処理する部分を In-band, ユーザインタフェースなど外部から操作される部分を Out-of-band と呼んでいる．図 3.2 に VuSystem の設計アプローチを示す.

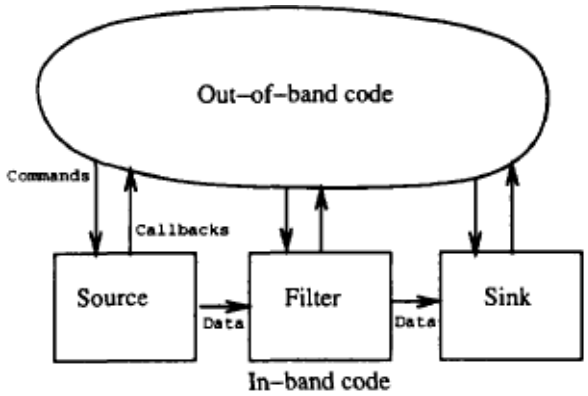

図 3.2: VuSystem のアプローチ

データ処理をする In-band の中でデータソース (入力)，フィルタ (加工)，データシンク (出力) という 3 種のモジュールに分割している．今後本論文においても，In-band, Out-of-band と言う表現を用いる．しかしこの研究はデータフローの規定がなされているのみで，この機構では第 2.1.1 項に述べた通信環境毎にそれぞれの通信品質を保持するといっ

た要求は満たせない．一つの入力から異種の入出力に対して，個別の環境に応じて映像転送を行えない．

既存技術

現在一般に利用できる映像・音声情報を汎用的に扱う為の機構は Windows 環境における DirectShow[29], Java 環境における Java Media Framework[30] がある．これらは API として実現されており，それぞれの環境において多種の映像・音声フォーマットと多種の入出力インタフェースに対応している．それらは VuSystem で述べられている入力・加工・出力の流れに基づき作られている．

- Java Media Framework
  Java Media Framework は Java 環境において映像音声情報を扱う為の API である．Java を用いて映像再生のアプリケーションを作成が可能となる．単一の計算機内で利用する場合は全て VuSystem と同等のデータソース (入力)，フィルタ (加工)，データシンク (出力) によって，各オブジェクトが表現されている．ネットワークへ送信することを前提に構築されていなかったため，データを RTP を用いて IP ネットワークに送信するための API 群を持つが，通常用いるデータの出力先とは異なるオブジェクトを介して表現されている．そのため Java Media Framework を利用する開発者は RTP を利用する場合は他のデータシンクとは異なった手続きを取る必要がある．また実用上の問題として，Java を用いて実装されているため，物理的なデバイスの操作が出来ない．各プラットフォームのライブラリを用いた実装も公開されており，それらは一部のハードウェアの利用を可能としているが，本研究が対象としている第 2.3 節に述べた映像転送に用いられている映像機器は利用できない．

- DirectShow
  DirectShow は Windows 環境における映像音声情報を扱う為の API である．基本的な思想は，VuSystem と同様に，入力・加工・出力から成り立つ．様々なフォーマット，様々な入出力を想定していて，自由度が高い．一方で，概念として入出力とフォーマットを分離していないモジュールも許容される．図 3.3 に入出力とフォーマットが分離していないモジュールの例を示す．

  DV は入出力インタフェースとして IEEE1394 を用いるが，図 3.3 に示したモジュールは IEEE1394 と DV フォーマットが分離していない．以上のように DirectShow は入出力とフォーマットの分離を行っていないため，第 3.1 項に述べた拡張性の問題は解決出来ていない．

3.2.2 本研究のアプローチ

本研究では第 3.1 項に述べた，特定の入出力や通信方法と扱うフォーマットの組合せ毎の個別化から発生する，異種の入出力が同時に利用できない機能面における問題と，拡張性の問題の解決として，フォーマットと入出力処理の機能を分離した機構を提案する．

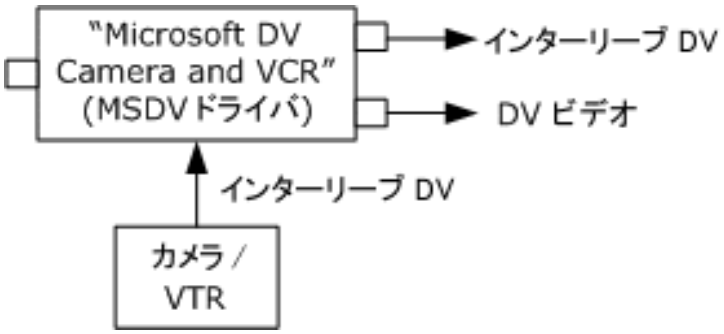

図 3.3: 入出力とフォーマットが分離していないモジュールの例

図 3.1 の機能を満たすための本研究の基本アプローチを図 3.4 に示す．図 3.4 は，入出力部分とフォーマットのそれぞれの共通部分を抜き出し，フォーマットと入出力及びその通信方法を分離させていることを示している．

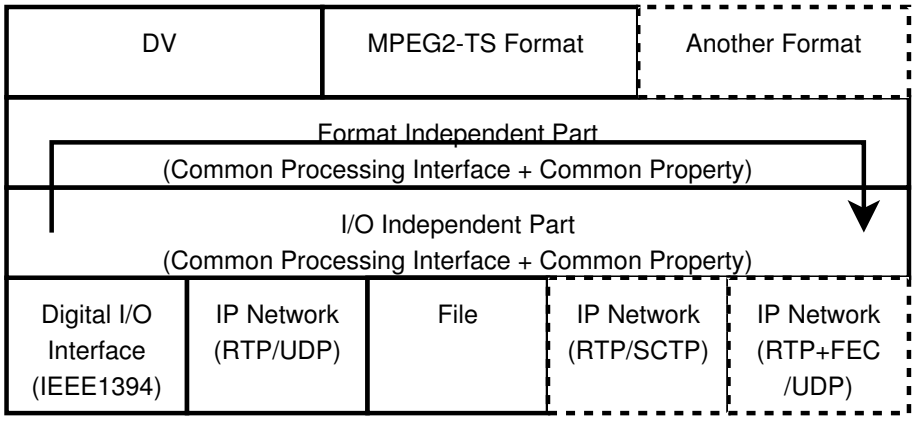

図 3.4: 本研究の基本アプローチを表す概念図

図 3.4 は最下段から，各入出力に依存する部分，全入出力に共通する部分，全フォーマットに共通する部分，各フォーマットに依存する部分を示している．また図内の矢印はデータの流れを示している．入出力を担当する部分から入力されたデータは，フォーマットを担当する部分に渡され，再度入出力を担当する部分に渡され，出力することを示している．図 3.4 では割愛しているが，実際にはデータは最下段の各入出力に依存する部分と，最上段のフォーマットに依存する部分に渡される．

図 3.4 で示した本研究のアプローチにおいて，IP ネットワークに対する通信は，以下の 3 つの理由から RTP を前提にする．

- IP ネットワーク上の実時間通信を行う為のプロトコルとして標準化されており，一般的に用いられている
- FEC を用いる場合や情報の多重化を行う場合などにおいて，それらの付加情報の表現方法が RTP ペイロードフォーマットにおいて標準化されている

- 下位のトランスポートプロトコル以下から独立している
  本アプローチにおいて下位のトランスポートプロトコル以下に何らかの情報を与える必要があった場合に，入出力依存部分として定義できる．

また，通信方法が異なる通信は成立しないため本アプローチではそれらを別々の異なった入出力とみなす．例えば UDP と SCTP の様にトランスポートプロトコルが異なる場合や，冗長化の有無の差異がある場合，それらの通信は成立しない．

これらの機能の分離を実現するためには，異なった入出力に透過的な一律的に定められた処理の呼び出し方法，並びに異なったフォーマットに透過的な一律的に定められた処理の呼び出し方法が定義される必要がある．

本研究では本アプローチを基に以下の点を実現する．

- 一律的な入出力処理インタフェースの定義による，異種入出力の利用における透過性の確保とその同時利用の実現
  映像音声情報の転送を前提にして，ネットワークもこの入出力の一つとみなす．全ての計算機が入力と出力の組合せで実時間情報を伝搬する．

- 入出力処理部分とフォーマット処理部分の分離
  第 3.1 節で述べた問題は特定の入出力とフォーマットの組合せしか想定されていないことが原因で発生しているため，その 2 者間を分離し独立的にすることが求められる．

本項で述べた機構を本論では入出力処理・フォーマット分離機構と呼ぶ．

3.2.3 入出力処理・フォーマット分離機構における環境に適応した通信

第 2.1.1 項で述べた通り，映像転送アプリケーションでは，通信相手の環境や状況に適応した通信を行うことが求められる．既存アプリケーションは第 3.2 項で述べた通り，環境や状況に応じた通信を行うために付加的な機能を追加していた．本研究で提案する入出力処理・フォーマット分離機構は異種の入出力を同時に利用することが前提となるため，複数異種の出力先に異なった品質で出力することが利用状況として想定される．図 3.5 に複数異種の出力先に異なった品質で出力を行う例を示す．

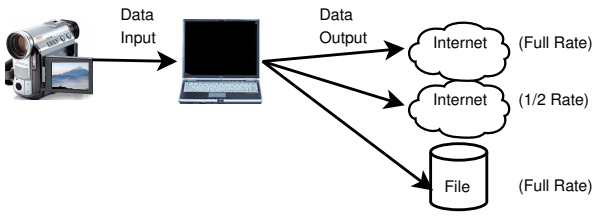

図 3.5: 複数異種の出力先に異なった品質で出力する例

図 3.5 で述べた例は映像機器から入力された映像を二次記憶装置と，ネットワーク上のある転送先にはデータを間引かずに出力し，他の転送先には，帯域を考慮してデータ量を半分にして出力することを示している．個別の環境に応じた配信を行うためには，各入出力とフォーマットに関する属性情報が対応関係を持つ必要がある．さらに動的に通信環境や状況に対応するためには，入出力からの品質報告と，それに基づいた各フォーマットに依存した処理による映像品質の変更を行う必要がある．図 3.4 で提案した機構におけるフォーマットの共通部分をさらに，フォーマットに関する属性情報と処理インタフェースに分離した図を図 3.6 に示す．

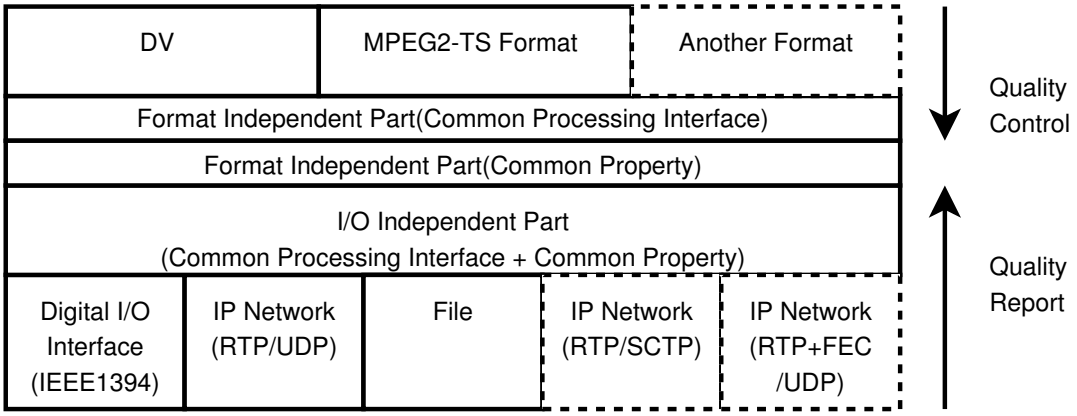

図 3.6: 入出力先毎に個別品質を扱うための入出力処理・フォーマット分離機構の概念図

図 3.4 と比較して，図 3.6 は，フォーマットに関して処理を行う部分と属性情報を持つ部分に分離していることを示している．また分離した属性情報の部分を介して入出力に依存した部分が通信品質の報告を行い，フォーマットに依存した部分が品質の変更を行うことを示している．

本機構ではフォーマットに関する部分をさらに処理を行う部分と属性情報を持つ部分に分離することで，入出力先毎の個別の品質管理を可能にする．

3.3 本章のまとめ

本章は既存のフォーマットを無変換で転送する映像転送システムの個別化から発生する問題を述べた．解決する手段として入出力処理とフォーマットに依存する部分の機能分離に関して述べた．また既存の入出力を自由に組み合わすことを目的とした機構が，入出力処理とフォーマットの分離を行う機構ではないため，拡張性の問題を解決できないことを述べた．本研究で用いるアプローチは処理と属性を各入出力に依存する部分，全入出力に共通する部分，全フォーマットに共通する部分，各フォーマットに依存する部分に分離することで実現する．さらに品質管理を考慮に入れ，フォーマットの属性情報に関する部分と，処理に関する部分に分離する．次章において各部分に含まれる処理や属性に関して述べ本機構の設計を行う．

第4章　入出力処理・フォーマット分離機構の設計

本章では，第3章で述べたアプローチを基に映像機器のデータフォーマットを無変換で転送する映像転送における入出力処理とフォーマットの依存性を低減し，入出力処理とデータフォーマットの自由な組合せを実現する機構の設計に関して述べる．

4.1　アーキテクチャ概要

本節では，本研究で提案する機構の概要を述べる．本節以降で，アーキテクチャの全体像を論じるにあたり，対象とする入出力とフォーマットを決定する必要がある．本論が対象とする映像転送形態は映像機器のデータフォーマットを変換することなく利用する映像転送である．現在，多種多様な映像機器があるが，大衆性のある民生用映像機器の代表的な機器に関して，その用途，圧縮方式，計算機と接続可能なインタフェースを表4.1に示す．

表 4.1: 民生用映像機器の種類

名称	用途	記録媒体	映像圧縮方式	データ構造	インタフェース
DV	撮影/蓄積	DV テープ	DV	DV	IEEE1394
DVD	配布	DVD ディスク	MPEG2 Video	MPEG2-PS	-
D-VHS	蓄積	D-VHS テープ	MPEG2 Video	MPEG2-TS	IEEE1394
MicroMV	撮影	MicroMV テープ	MPEG2 Video	MPEG2-TS	IEEE1394
デジタル放送機器	受信	-	MPEG2 Video	MPEG2-TS	IEEE1394
HDV	撮影	DV テープ	MPEG2 Video	MPEG2-TS/PES	IEEE1394

表4.1からIEEE1394インタフェースで接続する映像機器に焦点をあてる．計算機との接続可能なインタフェースを持たないDVD以外は，DVフォーマットかMPEG2-TSが用いられている．HDVは1080iの画質を利用する場合に限りMPEG2-PESを用いる．しか

し，計算機と接続可能なインタフェースである IEEE1394 上では，MPEG2-TS でデータを送受する．よって本論ではフォーマットに関しては，DV フォーマットと MPEG2-TS に焦点をしぼる．入出力及びその通信方法に関しては上述の IEEE1394 に加えて，VoD 形態で用いられる二次記憶装置に蓄積されているファイル，そして IP ネットワーク上での RTP の通信を対象とする．IEEE1394 上では実時間情報を転送することに適した，同期 (Isochronous) 転送方式がある．映像・音声情報は同期転送方式を用いて転送される．同期転送方式はバス帯域の 80%を占有できるなど，実時間情報を転送するための仕様を備えている．本論において IEEE1394 に関して論じる場合は同期転送を前提に論じる．

4.1.1 本機構の構成要素

本論で提案する機構は以下の 4 つの機能に分離した部分から構成する．名称の右側に本論で用いる略称を記す．今後分離した機能の総称を本論ではオブジェクトと呼ぶ.

- データを送受する部分
  - フォーマットデータオブジェクト:Format
    対象とするフォーマットの実際に扱うデータの保持とその属性を持つ部分
  - 入出力処理オブジェクト:IO
    対象とする入出力に関する処理を行う部分
  - フォーマット処理オブジェクト:Processor
    対象とするフォーマットに依存した処理を行う部分
- 他のオブジェクトを管理する部分
  - セッション管理オブジェクト:Session
    全体の多重入出力管理と時間管理を行う部分

これら 4 つの機能をクラスとして表現した場合のクラス関係図を図 4.1 に示す.

図 4.1 内の各クラスをつなぐ線にかかれた数字は対応関係を表している．映像と音声の情報の分離・合成を考慮すると，映像，音声と個別に分離された情報はそれぞれ，個別の入出力処理オブジェクトのインスタンスを介して送受される．そのためにフォーマットデータオブジェクトに対応する入出力処理オブジェクトの対応関係は 1 対 n となる．フォーマット処理オブジェクトのインスタンスはフォーマットデータオブジェクトのインスタンスに対して加工前，加工後のそれぞれ 1 つずつを保持するため最低 1 対 2 の関係になる．さらに複数の加工前のインスタンスを合成したり，1 つのインスタンスを複数の加工後のインスタンスに，分離，分配，複製したりすることが考えられるため 1 対 n(n≥2) となる．複数の入出力を扱うため，セッション管理オブジェクトと入出力処理オブジェクトの対応関係は 1 対多の関係になる．同様にセッション管理オブジェクトとフォーマット処理オブジェクトの対応関係は 1 対 n になる．

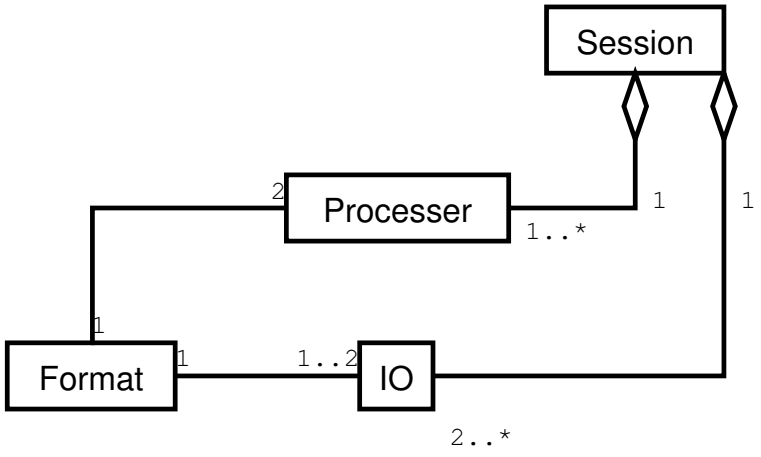

図 4.1: 本機構のクラス関係図

本機構では個々の入出力に依存する属性と処理は入出力処理オブジェクトと継承関係にある派生したオブジェクトで表現し，同様にフォーマットに依存した属性は，フォーマットデータオブジェクトと継承関係にある派生したオブジェクトで表現し，フォーマットに依存した処理はフォーマット処理オブジェクトと継承関係にある派生したオブジェクトで表現することになる．逆に共通する属性や概念は基底となる 4 つのオブジェクトで定義される．

4.1.2 各オブジェクトの役割と関係

本項では第 4.1.1 項に述べた，フォーマットデータオブジェクト，入出力処理オブジェクト，フォーマット処理オブジェクト，セッション管理オブジェクトにおいて，各部分の概要及び役割と他のオブジェクトの動作に関係する機能に関して述べ，オブジェクト間の関係について述べる．

フォーマットデータオブジェクト:Format

フォーマットデータオブジェクトは対象とするフォーマットの実際に扱うデータの保持とその属性を持つ．入出力処理オブジェクトとフォーマット処理オブジェクトに対してフォーマットに依存したデータの保持と参照を行う機能を提供する．フォーマットデータオブジェクトが他のオブジェクトに対して提供する機能を表 4.2 に示す．

フォーマットデータオブジェクトの役割はデータを保持し，かつ他のオブジェクトから参照可能にすることであり，他のオブジェクトに対しては，データの追加と，データの取得の機能を提供する．本論で対象とするフォーマットである，DV フォーマットや MPEG2-TS のデータを保持するオブジェクトはこのフォーマットデータオブジェクトと継承関係にある．図 4.2 にこの継承関係を示す．

表 4.2: フォーマットデータオブジェクトの提供する機能

機能名	概要
addData	データの追加
getData	データの取得

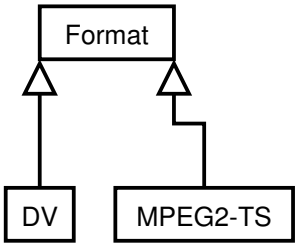

図 4.2: フォーマットデータオブジェクトの継承関係

入出力処理オブジェクト:IO

入出力処理オブジェクトは対象とする入出力に関する処理を行う．データ入力元，出力先に対して，入出力の機能を提供する．入出力を抽象化している概念では Unix におけるファイルによる抽象化がある．本機構においてもこの概念を利用する．入出力可能な状態にすることを *open*，入力することを *read*，出力することを *write*，入出力を終了することを *close* と表現する．それぞれの入出力は単純なバイトストリームではなくフォーマットデータオブジェクトを単位に行う．入出力オブジェクトの提供する機能を表 4.3 に示す．

表 4.3: 入出力オブジェクトの提供する機能

機能名	概要	詳細
read	読む	入出力対象からのデータ入力と脱カプセル化
write	書く	カプセル化と入出力対象へのデータ出力

本研究が対象としてる入出力である IEEE1394, RTP, ファイルはこの入出力処理オブジェクトと継承関係にある．この継承関係を図 4.3 に示す．

上述した通り入出力処理オブジェクトは必ず 1 つのフォーマットデータオブジェクトを参照する．*read* を行う場合はその処理内でフォーマットデータオブジェクトの *addData* が行われ，*write* を行う場合はその処理内でフォーマットデータオブジェクトの *getData* が行われる．

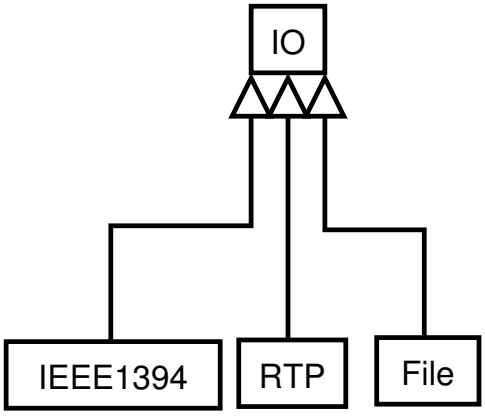

図 4.3: 入出力処理オブジェクトの継承関係

フォーマット処理オブジェクト:Processor

フォーマット処理オブジェクトは対象とするフォーマットに依存した処理を行う．フォーマット毎に可能な処理が異なるため具体的な処理内容は定義できない．例えば与えられたデータのデータ量を削減する場合に，フレーム棄却を行う方法が考えられるが，フレーム内圧縮方式と，フレーム間圧縮方式では全く処理の内容が異なる．

フォーマット処理オブジェクトの提供する機能を表 4.4 に示す．

表 4.4: フォーマット処理オブジェクトの提供する機能

機能名	概要
process	加工処理 (各オブジェクト事に異なる)

セッション管理オブジェクト:Session

セッション管理オブジェクトは全体の多重入出力管理と時間管理を行う．複数の異種入出力を同時に扱うため，各々の受信，送信タイミングの差異や，入出力の同期，非同期性の差異を吸収する役割を持つ．セッション管理オブジェクトの提供する機能を表 4.5 に示す．

セッション管理オブジェクトは，保持している関係するオブジェクトのインスタンスの機能を順々に実行する．実行するときの機能呼び出しの順序と，データの流れの例を図 4.4 に示す．

図内の黒色の矢印が処理の流れを示し，赤色の矢印がデータの流れを示す．セッション管理オブジェクトは入力側の入出力処理オブジェクトのインスタンスに対して *read*，フォーマット処理オブジェクトのインスタンスに対して *process*，出力側の入出力処理オブジェク

表 4.5: セッション管理オブジェクトの提供する機能

機能名	概要
run	セッションの開始
stop	セッションの停止

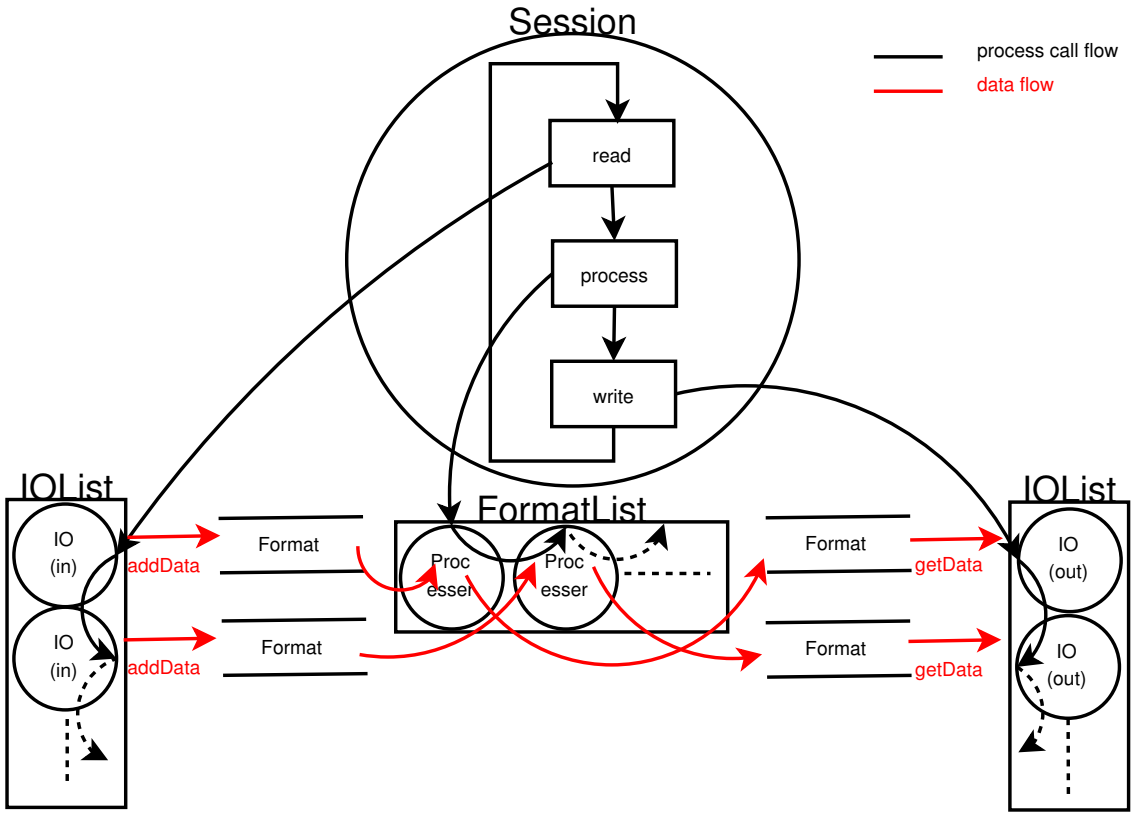

図 4.4: 機能呼び出しの順序とデータの流れ

トのインスタンスに対して *write* を順々に実行することで，複数の入出力に対して異なった品質で同時に通信を行う．

4.1.3 品質制御と各オブジェクトの役割

本項では品質制御を行う場合の，本機構の各オブジェクトの役割に関して論じる．

品質制御技術と RTCP の機能

RTP と対になって用いられる RTCP では，以下の情報が SR(SenderReport)/RR(Receiver Report) を用いて取得できる．

表 4.6: RTCP により取得できる情報

項目	取得，報告方法
帯域幅	経過時間と送信側のオクテットカウント及び受信側から報告された RTCP RR の欠損率による算出
ジッタ	受信側におけるパケット間 RTP タイムスタンプ差分とパケット間到着時間差の差分による算出, RTCP RR による送信側への報告
遅延	RR パケットの受信時間-SR パケットの送信時間-RR パケットの DLSR 値による RTT 算出
欠損率	受信側におけるパケット連続番号による算出,RTCP RR による送信側への報告

さらに RTCP XR(RTP Control Protocol Extended Reports) の Loss RLE Report Block によって，RTCP を用いた個々の RTP パケット欠損状態の通知方法が標準化されている．また，Packet Receipt Times Report Block において個々のパケット受信時間の通知方法が標準化されている．これによりペイロードフォーマットによっては RTP タイムスタンプだけでは正しく算出できないパケット間ジッタが算出できる．

RTCP による状態の検知が可能な時間間隔は，RTCP パケットの送受信間隔に依存する．RTCP パケットの送受信間隔の最小値は RFC3550[9] では RFC1889[31] の 5 秒から

$$\text{間隔の最小値} = \frac{360}{\text{セッションの帯域幅}\ (kbps)} \tag{4.1}$$

と変更されている．例として以下に MPEG2-TS フォーマットを採用している MicroMV 規格準拠のカメラのデータを Ethernet をデータリンクとして，IPv4 環境下でトランスポートプロトコルに UDP を利用し，IP フラグメントを発生させずに 1 対 1 の送信をする状況を想定する．この想定環境下で，データ量を 12Mbps, データリンクの MTU を 1500byte, IP ヘッダ，UDP ヘッダ，RTP ヘッダの和を 40byte として，RTCP パケットの間隔の試算を行う．IP フラグメントを起こさずに送信する場合は一つの IP パケット内には 7 つの MPEG2-TS パケットが同梱できる．以上から RTCP パケットの間隔の最小値は

$$\frac{360}{12 \times 10^3 \times ((188 \times 7 + 40) \div 188 \times 7)} = 2.91 \times 10^{-2}$$

となる．一方 RTCP は RTP の通信に影響を与えないために，RTP のデータ量の 5%に抑えることが推奨されている．さらに送信者の数と受信者の数によって計算方法が変わるがこの例では，送信者が全体の 25%未満となり以下の計算式が適用される．

$$\text{間隔} = \frac{RTCP\ \text{パケットの平均サイズ} \times \text{メンバの総数}}{RTCP\ \text{帯域幅}} \tag{4.2}$$

によって計算される．RTCP パケットの平均サイズを RTCP SR と RR を送受する前提でそれぞれの Report Count を 0 と 1 として 58byte とすると

$$58 \times 2 \div (12 \times 10^6 \div 8 \times ((188 \times 7 + 40) \div 188 \times 7) \times \frac{5}{100}) = 1.50 \times 10^{-3}$$

となる．

従ってRTCPの送信間隔は最小値の方が大きいため29.1ミリ秒となり，NTSC，SD画質の場合1フレーム以下相当になる．品質制御が映像音声情報として構成できる単位，即ちDVにおけるフレームや，MPEG2-TSの符号化・復号化単位となるフレームの集合であるGOP(Group of Picture)単位で行われる場合は，精度として十分であると言える．

同様の計算をDVフォーマットで行うとRTCPの送信間隔は12.1ミリ秒となり，フレーム単位以下での状態の通知が可能な為，有用である．

本項においてRTPの通信に対してRTCPが状況を通知するための十分な表現能力持っていることを示した．またRFCが推奨する計算方法で算出したRTCPの通信間隔が品質制御を行う精度をフレームを単位とする場合は必要十分であることを示した．従って本機構においてIPネットワークの情報取得にはRTCPのみを用いる．

圧縮方式の差違によるデータ欠損対応の違い

転送時のデータ欠損に対して有効な技術には欠損を回復する技術と予防する技術がある．利用可能帯域幅を越えたデータ量を送出するとデータ欠損が発生する．従って予め送出データ量を減少させることでデータ欠損を予防できる．

映像情報のデータ量を変化させる技術には空間によるスケール，時間によるスケール，SNR(Signal to Noise Ratio)によるスケールがある．

DVフォーマットのように，フレーム内圧縮方式のフォーマットで各フレームのデータサイズが一定である場合，データ量を変化させる手法としては，時間によるスケール，つまりビデオフレームの一部棄却を行う手法が適している．

フレーム間圧縮方式のフォーマットであるMPEGでは規格上スケーラビリティを持つ．SNRスケーラビリティと空間スケーラビリティを持つ規格が策定されていが，現在民生用デジタル機器に用いられている規格では利用できない．

またDCT係数の削除によって実現されるローパスフィルタを用いて，圧縮率を変更する方法もある[32]．しかし，この方法は，計算量が多く，またスライス層単位のデータ転送を行うなど，独自のデータ構造を前提にしているため，本研究が想定しているRTPへの準拠が不可能である．時間的なスケールはフレーム棄却であるが，フレーム間に依存性があるため，フレームの種類の判別を要する．そのためフレーム内圧縮と比較すると単純な棄却ができず，計算量が多くなる．従ってMPEG2のデータ欠損には回復技術が有用な技術である．データ欠損に対しての回復技術が，FEC(Forward Error Collection)やARQ(Automatic Repeat Request)[33]である．

品質制御と本機構における各オブジェクトの役割

本機構のフォーマットデータオブジェクトは，データ保持のために単位時間あたりのデータ量の情報を持つ．品質制御を目的としてフォーマットデータオブジェクトは，変更すべきデータ量の割合も保持する．データ量と割合の両方の情報を保持することで，変更後のデータ量も算出できる．本機構ではこの割合を変更データレートと呼ぶ．

利用可能な帯域幅の変動が少ないが，その帯域幅が，用いるフォーマットのデータ量よりも小さい場合など，データ量の変更割合を静的に保持すべき状況がある．上述の変更データレートを動的に変更するか静的に保持するかのブール値の情報を持つ必要がある．フォーマットデータオブジェクトが保持すべき品質制御に関する情報を表 4.7 に示す．

表 4.7: フォーマットデータオブジェクトが保持すべき品質制御に関する情報

名称	意味
変更データレート	変更すべきデータ量の割合
変更データレート静的フラグ	変更デ ータレートを静的に保持するか否かのブー ル値

通信品質の監視は，入出力処理オブジェクトにおいて行われる．データ削減が可能な部分はフォーマット処理オブジェクトである．この 2 つは，それぞれフォーマットデータオブジェクトを参照しているためフォーマットデータオブジェクトを介して情報の送受を行う．

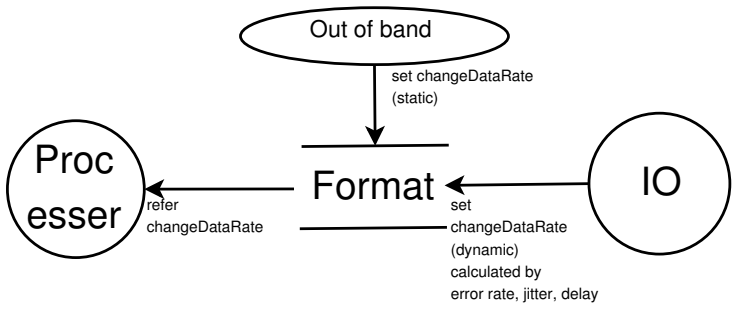

図 4.5: 変更データレートの設定と参照

FEC の冗長符号化率は Out-of-Band から指定するか，または RTCP の欠損率から決定できる．ARQ は，RTCP XR Loss RLE Report Block の情報から実現できる．RTCP が 1 フレーム時間内に送受されるので，どちらの機能も入出力処理オブジェクトのみで実現できる．

ジッタを吸収するためには適切なスケジューリングが必要である．本機構においては，出力の役割を担う入出力オブジェクトのインスタンスがパケットの送出すべき時間情報を保持する．その情報をセッション管理オブジェクトが回収して，送出すべき時間情報より早かった場合に休止処理を行い時間間隔を保持する．

4.2 各部分の詳細な機能と属性

本節では，各部分の持つ機能について詳細に述べ，必要となる属性に関して述べる．本節は以下の順番で各オブジェクト毎に設計の詳細を述べる．

- 分析
  フォーマットデータオブジェクトと入出力処理オブジェクトに関しては本機構が扱う対象のフォーマット及び入出力に対してそれぞれ分析を行う.
- 固有な情報及び処理
- 保持情報
- 初期化に必要な情報
- 各派生オブジェクトに依存する部分

4.2.1 フォーマットデータオブジェクト

フォーマットデータオブジェクトはフォーマットに依存した属性と，入出力に用いるデータを保持する役割を持つ．本研究で対象とするフォーマットは 4.1 に示した通り DV フォーマットと MPEG2-TS フォーマットを対象とする．本項はフォーマットデータオブジェクトにおいて，対象とするフォーマットの分析，本論で定義するデータ量の単位，保持するデータの領域とその状態を示すための情報，フォーマットデータオブジェクト全体において保持する情報，フォーマットデータオブジェクト初期化時に必要な情報，各フォーマットに依存する部分に関する事項を述べる.

対象とするフォーマットの分析

DV フォーマットと MPEG2-TS は内部で保持するデータの圧縮方式は全く異なるが，データ構造が持つ情報は共通項が多い．この 2 種類のフォーマットが持つ情報の比較を表 4.8 に示す.

表 4.8: 対象とするフォーマット間での比較

フォーマットの種類	DV	MPEG2-TS
パケット最小単位	DV DIF	MPEG2-TS packet
パケット識別子	sct	PID
パケットサイズ	80	188
映像と音声の分割	sct の識別	PID での識別
時間情報	タイムコード (subcode)	PCR, OPCR
データ単位	フレーム	GOP

データ量の単位

本機構ではデータ量の基本単位として PACKET と BLOCK を定義する．PACKET はそのデータフォーマットにおけるパケットサイズを示す．DV の場合 DIF ブロックの 80byte, MPEG2-TS の場合は MPEG2-TS パケットの 188byte を示す．BLOCK は映像音声情報として構成できるデータ単位を示す．DV の場合はフレーム，MPEG2-TS の場合は GOP を示す．

データ領域とその状態を示すための保持情報

フォーマットの加工処理を行う場合，フレームや GOP など映像音声情報として構成できる単位で行わなければならない．フォーマットデータオブジェクトは，映像，音声などのデータを映像音声情報として構成できる単位，第 4.2.1 項で述べた BLOCK を 1 単位として内部的に保持する．

データを保持する領域は実際に入出力処理オブジェクトで送受されるデータ以外に関連情報を持つ．本論が焦点をあてている DV フォーマット及び MPEG2 はともに単一のデータ構造内に映像，音声，時間などを格納したシステム情報が多重化されている．本論ではこれらの多重化された中の 1 種類の情報をエレメントと呼ぶ．図 4.6 に多重化されている状態を示す．

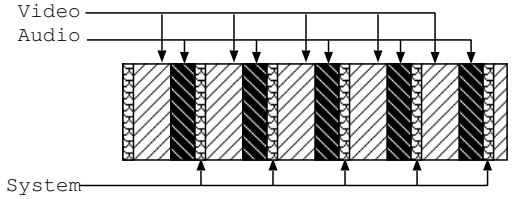

図 4.6: データが多重化されている状態

本機構のフォーマットデータオブジェクトはこの映像，音声，システム情報，及び全体のそれぞれにデータ保持状態を表す情報を個別に持つ．表 4.9 にデータ保持領域を表すために保持する情報を示す．この一覧を本論ではデータ保持領域表現情報と呼ぶ．

データ保持領域表現情報の状態は行われる処理によって遷移する．データ保持領域表現情報の状態の一覧を図 4.10 に示す．

BLOCK 単位分のデータを保持するための情報を本論ではデータ保持情報と呼ぶ．このデータ保持情報の一覧を表 4.11 に示す．映像と音声の分離と合成の処理の簡易化を目的として，映像，音声，システム情報で個別にデータ保持領域表現情報を持つ．

データ保持情報または，データ保持情報内で持つデータの実体を共有したい状況がいくつか考えられる．その状況は以下の 3 項目である．

- 映像，音声，システム情報が個別に入力される場合の情報の合成
  追加される情報が独立しているため，データ保持データ構造全体を参照する．

表 4.9: データ保持領域表現情報

名称	型 (とりうる値)
状態	状態識別子 (整数)，図 4.10 参照
データ領域	ポインタ
データ領域の長さ	整数
データの有無を示すフラグ	ブール値
PACKET 単位のデータ保持数	整数
PACKET 単位のデータ参照数	整数

表 4.10: データ保持領域表現情報の状態

状態名	内容
READY_READ	読み込み可能状態
READING	読み込み中
READY_PROCESS	処理可能状態
PROCESSING	処理中
READY_WRITE	書き込み可能状態
WRITING	書き込み中

- 映像，音声，システム情報が個別に出力する場合の情報の分離
  追加される情報が独立しているため，データ保持データ構造全体を参照する．
- 同一の情報を複数の出力先に出力する場合の分配
  データの実体のみ参照．データ参照数は宛先毎に個別に持つ．

それぞれの状況を図 4.7，図 4.8，図 4.9 に示す．

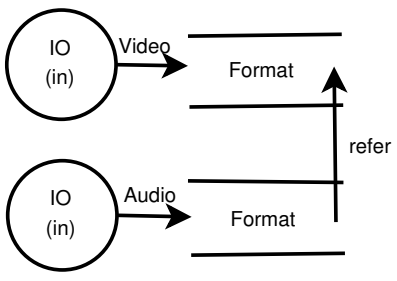

図 4.7: 合成:入力時に同じデータ領域を共有する

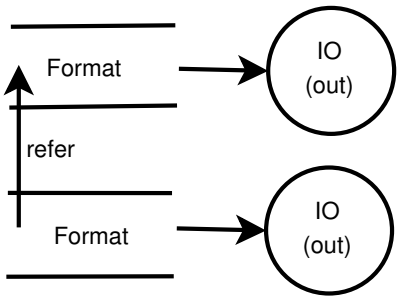

図 4.8: 分離:出力時に同じデータ領域を共有する

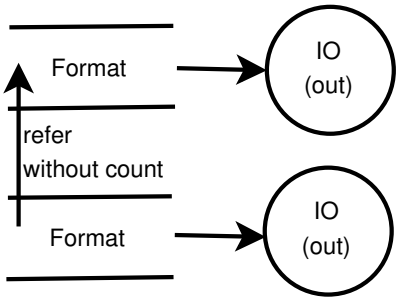

図 4.9: 分配:出力時に同じデータ領域を共有する

データ領域の確保時にこの 3 種類のいずれかが Out-of-band から指定される．

表 4.11: データ保持情報

名称	型 (とりうる値)
全体	データ保持領域表現情報
映像	データ保持領域表現情報
音声	データ保持領域表現情報
システム	データ保持領域表現情報
時間	時間
データ実体	ポインタ

多重化されているデータには各エレメント毎に識別子がある．この識別子の決定方法は各フォーマットによって異なる．これに関しては後述する．多重化されたデータから一部のエレメントをとりだすためには，その識別子番号が必要であり，前述のデータ保持領域表現情報を各エレメントで構成するためには BLOCK 内の PACKET の個数が必要である．またエレメント毎にデータの追加処理，データの取得処理が異なる可能性があるためそれらの機能を内部的に保持するデータ構造が必要である．これを本論ではエレメント識別子表として定義する．

表 4.12: エレメント識別子表

名称	型 (とりうる値)
識別子	整数
個数	整数
追加処理	ポインタ
取得処理	ポインタ

データの追加処理と取得処理

表 4.2 に示した処理のそれぞれには，各入出力が送受する時間情報の更新に関する情報が必要となる．時間情報の算出方法は各入出力と各フォーマットの組合せ毎に異なるため，詳細は第 4.2.2 項において述べるが，各フォーマット毎に，各入出力のヘッダ内で保持する時間情報の更新間隔が異なる．DV はフレームの更新時，MPEG2-TS は各パケット毎に時間情報が更新される．そのため，表 4.2 に示したデータの追加処理時には，入出力において時間情報が更新されたか否かの情報が入出力処理オブジェクトからフォーマットデータオブジェクトへ渡される必要がある．データ取得時には入出力において時間情報が更新されるべきか否かの情報が入出力処理オブジェクトからフォーマットデータオブジェクトへ渡される必要がある．

またデータの取得処理時には，転送するデータ単位を示す情報が必要となる．他のオブジェクトがフォーマットデータオブジェクトに対して PACKET 単位でデータ取得を行う必要がある状況は，入出力処理オブジェクトが PACKET 単位で出力する場合であり，IEEE1394 や RTP への出力時である．他のオブジェクトがフォーマットデータオブジェクトに対して BLOCK 単位でデータ取得を行う必要がある状況は，入出力処理オブジェクトが BLOCK 単位で出力する場合であり，ファイルへの出力時である．

上述した事項を踏まえたフォーマットデータオブジェクトにおけるデータの追加処理と取得処理の詳細を表 4.13 に示す．

表 4.13: データの追加処理と取得処理の詳細

メソッド名	概要	実行時の追加情報 (引数)
addData	データの追加	データ，時間情報の更新情報
getData	データの取得	データ，転送単位，時間情報の更新情報

フォーマットデータオブジェクトが保持する情報

フォーマットデータオブジェクトが保持する情報の一覧を表 4.14 に示す．

パケットサイズとブロックサイズにおいては既に第 4.2.1 項で述べた．伝送時の最小単位となる PACKET のサイズと映像音声情報としての意味を持つ単位の BLOCK のサイズをそれぞれ保持する．これが入出力に用いられるデータの単位となる．さらに，映像情報の時間的単位を統一するためにブロック内に含まれるフレーム数の情報が必要となる．映像周波数は，入出力処理において必要でかつ，加工処理においても必要となり得るフォーマットに依存した情報であるため，フォーマットデータオブジェクトにおいて保持する．

エレメントの数はフォーマット毎に異なる．エレメント数は各フォーマットに依存した情報であるが共通の概念であるため，フォーマットデータオブジェクト内で保持する．エレメント識別子表は表 4.12 に示した情報を内部的に持つが，エレメント識別子はフォーマットによって動的な場合があるため，エレメント識別子表が生成できたか否かの情報を持つ必要がある．エレメント識別子検知状態はその状態を保持する．

バッファは伝送路におけるジッタの吸収に用いられる．バッファBLOCK 最大数と初期バッファBLOCK 数は，バッファ量を表す．前者はバッファ可能な最大の大きさを保持し後者は通信開始時に確保するバッファの量を保持する．また，入出力処理オブジェクトがそれぞれバッファのどの BLOCK を利用しているかの情報を保持する．

フォーマットデータオブジェクト初期化時に必要な情報

フォーマットデータオブジェクト初期化時に最低限必要な情報はフォーマットの種類と映像周波数になる．これは Out-of-band から指定される必要がある．表 4.1 に述べた

表 4.14: フォーマットデータオブジェクトが保持する情報

名称	型 (とりうる値)	利用時の指定箇所
種類	種類識別子 (整数)	Out-of-band から指定
PACKET サイズ	整数	静的に保持
BLOCK サイズ	整数	静的に保持・動的に変化
フレームあたり PACKET 数	整数	静的に保持・転送開始時に決定
BLOCK あたりフレーム数	整数	静的に保持・転送開始時に決定
総 BLOCK 数	整数	動的に変化
映像周波数	小数	Out-of-band から指定・転送開始時に決定
変更データレート	分数	Out-of-band から指定・動的に変化
変更データレート静的フラグ	ブール値	Out-of-band から指定
エレメント数	整数	転送開始時に決定
エレメント識別子検知状態	ブール値	転送開始時に決定
エレメント識別子	エレメント識別子表	転送開始時に決定
初期バッファBLOCK 数	整数	Out-of-band から指定
バッファBLOCK 最大数	整数	Out-of-band から指定
現在読み込んでいる BLOCK 位置	ポインタ	動的に変化
現在書き込んでいる BLOCK 位置	ポインタ	動的に変化
データ保持領域	データ保持情報	オブジェクト生成時に初期化後静的に保持

機器で用いられる映像品質は DV の場合は SD 画質，MPEG2 の場合は MP@ML または MP@H-14 と呼ばれる規格で定められたいずれかの品質となる．MP@ML と MP@H-14 は規格上映像周波数が異なる．種類と映像周波数を与えることで用いるフォーマットと品質が一意に決まる．また映像方式には NTSC, PAL が存在するが，これも周波数の違いにより判別できる．

各フォーマットに依存する部分

- 映像音声データの識別方法
  - DV フォーマット
    DV フォーマットは基本単位である DIF ブロック内の DIF ヘッダ内にある SCT

フィールドがデータの識別に用いられる．SCT の値と情報の対応関係は静的に決まっている．SCT の値と情報の対応関係を表 4.15 に示す.

表 4.15: DV フォーマットの情報の識別に用いる SCT フィールド

値	情報の種類
000	Header
001	Subcode
010	VAUX
011	Audio
100	Video

- MPEG2-TS
  MPEG2-TS は PID(Program ID) と呼ばれる情報を持ち，PID を用いることで映像，音声，システム情報を識別することができる．PID は特定のパケット以外は動的に決定される．MPEG2-TS は放送用途を意識して設計されているため，単一の通信路で複数の番組を伝送することが想定されており，特定のパケットを除いて固定の PID を持たない．固定の PID を持つ情報が PAT(Program Association Table) であり PID が 0 である．PAT は多重化された複数の番組の識別番号と PID を関連付けるために用いられる．番組内の音声，映像，時刻情報，その他の情報と PID との対応関係を持つのが PMT(Program Map Table) である．PMT の PID は PAT によって指定される．図 4.10 に MPEG2-TS における多重化と PID の関係を示す.
  以上の MPEG2-TS の構造から，ある単一の番組 (映像，音声，システム情報が多重化された情報) から映像，音声の識別するためには以下の処理が必要となる．
  1. PID が 0 のパケットを探索し，解析を行い PMT の PID を取得する．
  2. PID が PMT の PID と一致するパケットを探し，解析を行いエレメントの PID を取得する．

- データ量の決定

  - DV フォーマット
    DV フォーマットの単位時間あたりのデータ量は規格で定められているため一意に決定される．
  - MPEG2-TS
    MPEG2-TS は，複数の圧縮率を選択できるため一意に定まらない．従ってデータ量の推定を行う必要がある．時間当たりのデータ量を算出するために，MPEG2-TS パケットの PCR(Program Clock Reference) を利用する．PCR は受信側の

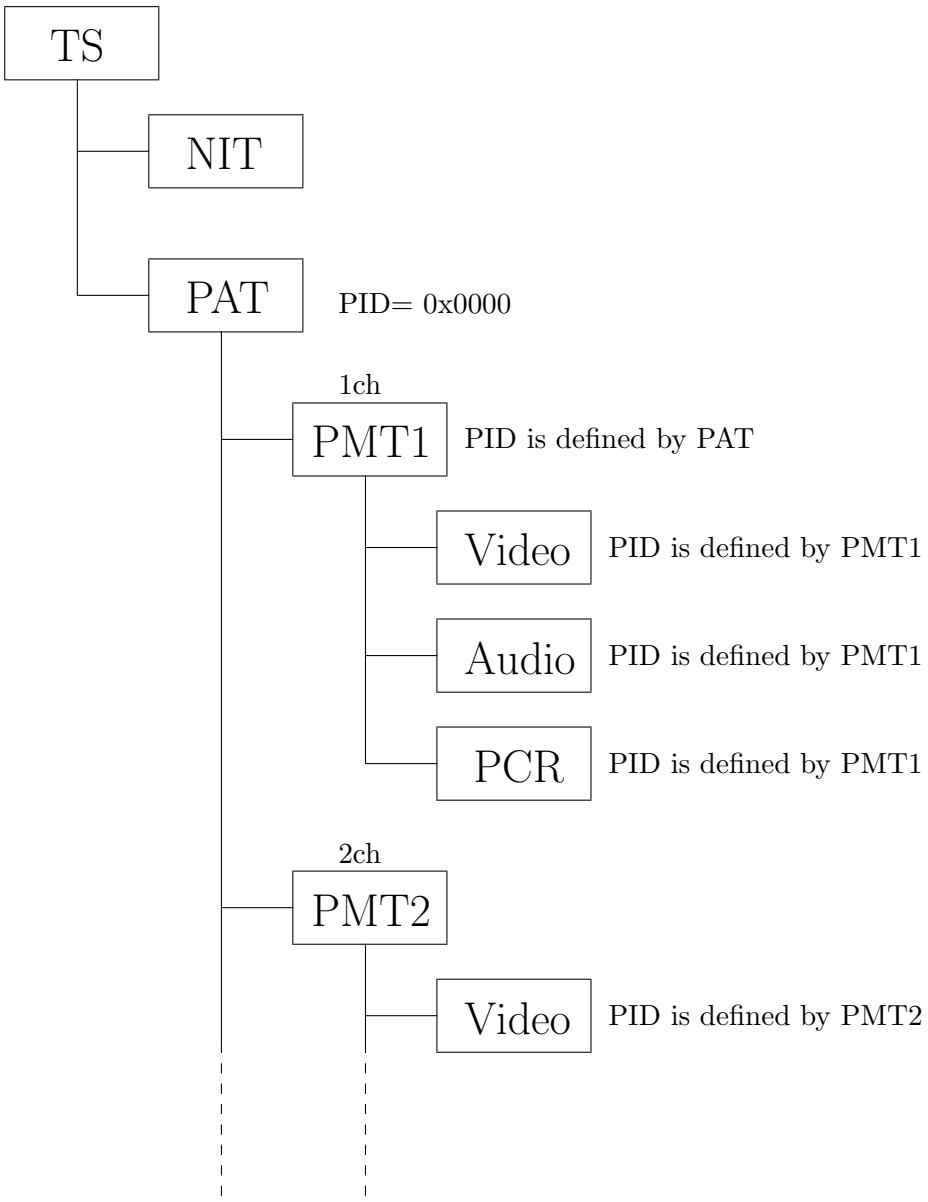

図 4.10: MPEG2-TS における多重化の仕組み

MPEG2-TS 機器の STC(System Time Clock) を設定するための基準となる時刻情報である．ある PCR を PCR_N とし，その次の PCR を PCR_{N+1} とすると PCR_{N+1} と PCR_N 間に流れたデータ量を PCR_{N+1} と PCR_N の値の差分で割ることで時間当たりのデータ量を推定できる．PCR の送信間隔は 100ms 以下と定められている．従ってこのデータ量の推定には最大で約 200ms かかる．PCR の PID は PMT に記述されているため前項に述べた処理で情報が取得できる．

4.2.2 入出力処理オブジェクト

本項では入出力処理オブジェクトの詳細な属性と機能に関して述べる．本項は，対象とする入出力の分析，対象とする入出力における同期性の差異，入出力処理オブジェクトの保持情報，入出力処理オブジェクト初期化時に必要な情報，各入出力に依存する部分に関して述べる．

対象とする入出力の分析

本論では対象としている入出力として映像機器に一般的に用いられる IEEE1394, IP ネットワーク上で実時間通信に使われる一般的なプロトコルである RTP, 計算機内の蓄積メディアとしてファイルを入出力対象とする．これら対象とする入出力の比較を表 4.19 に示す．

表 4.16: 対象とする入出力間での比較

	IEEE1394	RTP	File
通信可能な方向	入力/出力	入力/出力	入力/出力
帯域保障	○	△ (RSVP 等の併用)	×
CBR	8KHz	-	-
伝送路エラー発生確率	低い	高い	低い
伝送路上でのゆらぎ	ほぼない (12.5ns)	起こる	ほぼない

IEEE1394 の同期 (Isochronous) 転送のパケットを図 4.11 に，IP, UDP, RTP パケットを図 4.12 に示す． この両者が持つ情報は非常に酷似している．この両者の対応関係を

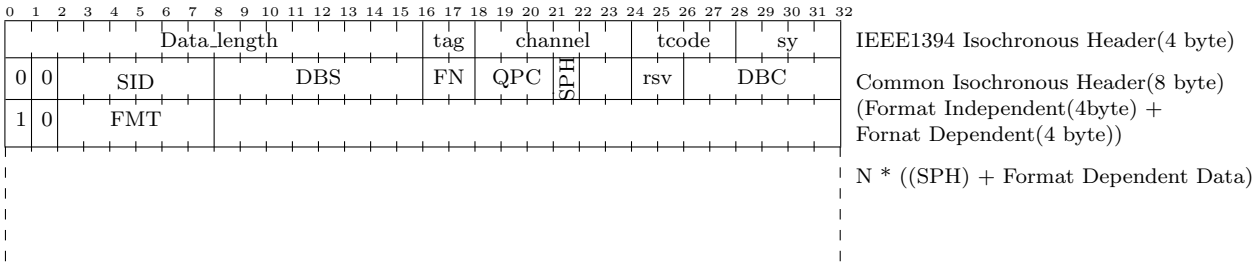

図 4.11: IEEE1394 Isochronous packet のデータ構造 (フォーマット依存部を除く)

表 4.17 に示す．IEEE1394 Isochronous Header が，IP 及び UDP とほぼ同等の情報を持ち IEEE1394 CIP(Common Isochronous Packet) Header が RTP とほぼ同等の情報を持つ．データサイズは RTP においては扱うフォーマット毎に RFC で明確化されており，ペイロードタイプによってフォーマットを判断することで決定されるためヘッダではその情

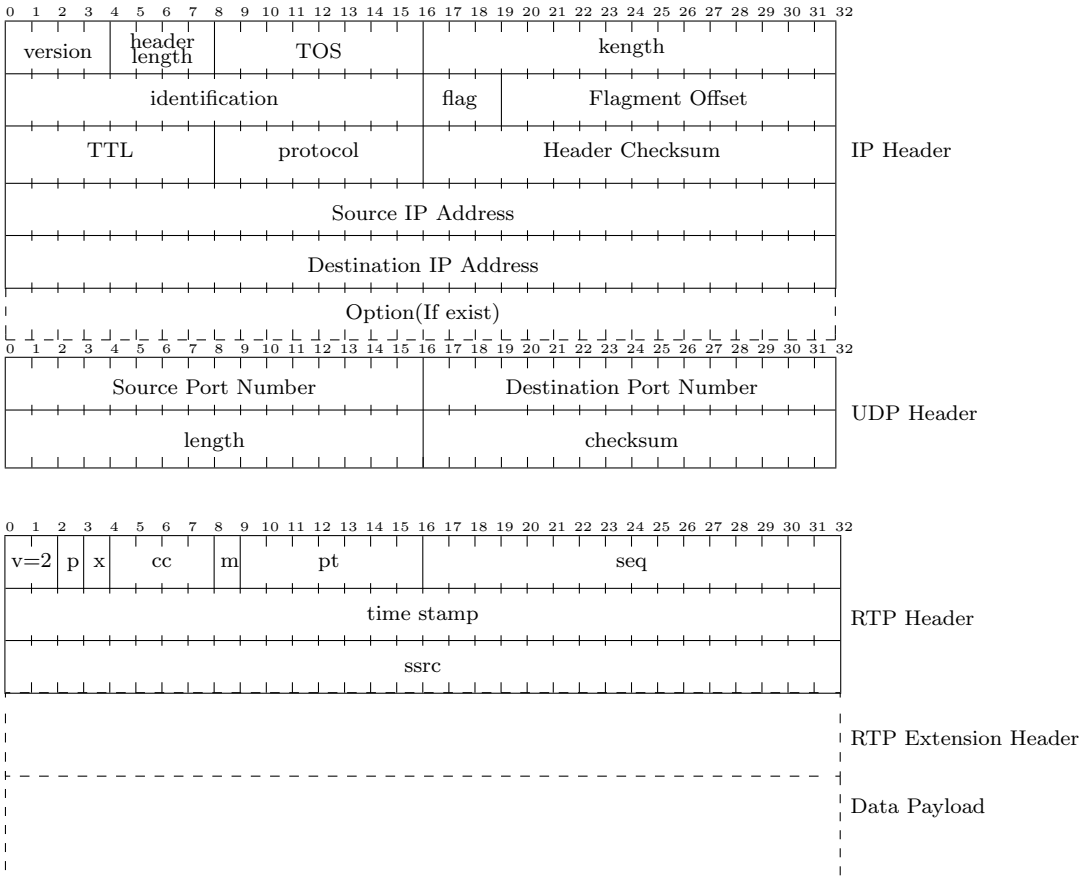

図 4.12: IP + UDP + RTP packet のデータ構造

報を保持しない．IEEE1394 のみが持つパディングの情報は，SD-DV，MPEG2-TS の 2 フォーマットとも利用しない．

表 4.17 に示した通り，ヘッダ内にフォーマットに依存した値が挿入されるため，各入出力オブジェクトは，出力時には保持しているフォーマットデータオブジェクトを参照し，適切な情報を埋め込む必要がある．

また時間情報の算出方法が入出力とフォーマットの組合せ毎に異なる．それぞれの時間情報の算出方法を述べる．

- RTP を用いて通信する場合の時間情報
  RTP のタイムスタンプは初期値は乱数であり，その増加方法がフォーマットによって異なる．以下 DV 及び MPEG2 TS に関して RTP タイムスタンプの増加方法に関して述べる．

  – DV フォーマット
    DV フォーマットはフレームを単位とした映像フォーマットであり RTP のタイムスタンプもフレームを単位に増加する．RFC3189[20] 2.1 に述べられている通り，SD-DV では NTSC 映像方式で 3003, PAL 映像方式で 3600 ずつ増加

表 4.17: IEEE1394 ヘッダと RTP ヘッダの類似点

	IEEE1394	RTP
データ長	Length(Isochronous)	length(IP)
通信の種別	tcode(Isochronous)	proto(IP)
宛先内での区別	channel(Isochronous)	port(UDP)
データの分割	FN(Fraction Number)(CIP)	Fragment Offset(IP)
フォーマット依存値	FDF(CIP)	PT(RTP)
送信元識別	SID(CIP)	SSRC(RTP
データサイズ	DBS(CIP)	RFC で規定
パディング	QPC(CIP)	-
時間情報	SYT/SPH(CIP)	TS(RTP)
パケット番号	DBC(CIP)	Seq(RTP)

する．RTP タイムスタンプは多くのフォーマットが 90KHz を単位にしている．タイムスタンプの増分は 90KHz をそれぞれの映像方式のフレームレートで除算した値になっている．同一フレーム内 は同一のタイムスタンプを用いる．

- MPEG2-TS フォーマット
  MPEG2-TS をフォーマットとして用いた場合の RTP タイムスタンプは 通信路のジッタ等の計測のために用いられるように設計されており 90KHz を単位にして経過時間を増分する．

- IEEE1394 を用いて通信する場合の時間情報
  IEEE1394 のタイムスタンプはフォーマットによって CIP ヘッダの SYT, SPH のどちらかのフィールドに保持されるがともにこの値は IEEE1394 CYCLE_TIME register を参照している．以下 DV 及び MPEG2 TS に関して述べる．

  - DV フォーマット
    時間情報に SYT フィールドを用いる．DV フォーマットの場合，フレームの更新時のみ時間情報が計算される．SYT は 16 bit のフィールドであり，上位 4bit が cycle count とよばれ，8KHz が単位となっており，下位 12bit がフレームの開始以外のパケットには，0xffff が挿入される．
  - MPEG2-TS フォーマット
    時間情報に SPH を用いる．各 MPEG2-TS パケット事に SPH が付く．SPH は 25bit のフィールドであり上位 13bit が cycle count になる．

上述の各伝送路における時間情報の差違を，表 4.18 に示す．

DV を用いる場合は，伝送路の種類に関わらず，フレーム開始にあたるパケットにデータが挿入されるため，必ず映像周波数の時間が用いられる．以上から各伝送路における時

表 4.18: 各伝送路における時間情報の差異

	IEEE1394	RTP
DV	フレーム開始に当たるパケットにCYCLE_TIME register を参照した値を挿入	フレーム開始にあたるパケットで90KHz/映像周波数を増分
MPEG2	全パケットにCYCLE_TIME register を参照した値を挿入	パケット毎に経過時間/90KHz を増分

間情報の決定には映像周波数が必要となる．

対象とする入出力における同期性の差異

入力と出力の種類が異なり速度差がある場合， 有限バッファ問題が起きる．対象とする入出力における同期性の差異を表 4.19 に示す．本論では同期性を保証しないものに対して非同期と表現する．

表 4.19: 同期性の差異

	IEEE1394	RTP	ファイル
実際	同期	非同期	非同期
要求	同期	同期	非同期

IEEE1394 の入力からネットワークへの出力の様に，同期性のある入力を受けた場合，計算機はその入力をそのまま出力すれば同期的な出力が可能となる．ファイルの入力からネットワークへの出力の様に同期性を保証しない入力を受けた場合，計算機はデータ転送先でバッファオーバーフローを起こさないために同期的な出力を行う必要があり，適切な間隔を保持する必要がある．本機構は，セッション管理オブジェクトが保持している入出力オブジェクトの送信間隔が最も短いオブジェクトの間隔に合わせて休止処理を定期的に行うことで，この問題を解消する．

入出力処理オブジェクトが保持する情報

前述の入出力の分析を踏まえて以下に入出力処理オブジェクトが共通して保持する情報を示す．

表 4.20: 入出力処理オブジェクト保持情報

属性	型 (とりうる値)	利用時の指定箇所
種類	種類識別子	Out-of-band から指定
実際の入出力の方向	ブール値 (入力/出力)	Out-of-band から指定
入出力対象の名前	文字列	Out-of-band から指定
入出力機器内識別子	整数	Out-of-band から指定
可能な転送 (同期性)	ブール値 (同期/非同期)	静的に保持
行うべき転送 (同期性)	ブール値 (同期/非同期)	静的に保持
転送のデータ量	整数	入出力依存
扱うフォーマット	ポインタ	Out-of-band から指定
転送単位	ブール値 (PACKET/BLOCK)	入出力依存
転送間隔	時間	動的に変化
最終送受信時間	時間	動的に変化
ディレイ	時間	動的に変化
ジッタ	時間	動的に変化
入出力カウントの総数	整数	動的に変化
エラーカウントの総数	整数	動的に変化

表 4.21: 共通属性の子オブジェクトにおける意味と値

名称	IEEE1394	RTP	ファイル
入出力対象の名前	デバイスファイル名	ホスト名/IP アドレス	ファイル名
入出力機器内識別子	バスの番号	ポート番号	-
可能な転送 (同期性)	同期	非同期同期	非同期
行うべき転送 (同期性)	同期	同期	非同期
転送のデータ量	静的に保持・データ量に応じて変化	Out-of-band から指定	フォーマット依存
転送単位	PACKET	PACKET	BLOCK

入出力処理オブジェクト初期化時に必要な情報

入出力処理オブジェクトの初期化時に必要な情報は，種類，入手力対象の名前，入出力の方向，参照するフォーマットデータオブジェクトとなる．転送時の時間情報として必要となる映像周波数はフォーマットデータオブジェクトを参照することで取得できる．第 4.2.1 項において映像周波数をフォーマットデータオブジェクトの初期化時に必要な情報

と定義しているため，必ず取得できる．

各入出力に依存する部分

本項では入出力に依存する部分を示す．

- IEEE1394
  Out-of-band から指定する追加の属性として IEEE1394 channel(整数) を保持する．
- RTP
  Out-of-band から指定する追加の属性として Multicast TTL(整数), Multicast Interfac(文字列) を保持する．また動的に変更される追加の属性として RTCP の送受信間隔 (時間)，RTCP の最終送受信時間 (時間) を保持する．

4.2.3 フォーマット処理オブジェクト

フォーマット処理オブジェクトはフォーマットに処理内容が依存するため処理内容は定義できない．本項はフォーマット処理オブジェクトが保持する情報，フォーマット処理オブジェクト初期化時に必要な情報に関して述べる．

フォーマット処理オブジェクトが保持する情報

フォーマット処理オブジェクトが保持する情報を表 4.22 に示す．

表 4.22: フォーマット処理オブジェクトが保持する情報

名称	型 (とりうる値)	利用時の指定箇所
種類	種類識別子 (整数)	Out-of-band から指定
入力可能数	整数	静的に保持
出力可能数	整数	静的に保持
入力 Format 配列	Format 配列	Out-of-band から指定
出力 Format 配列	Format 配列	Out-of-band から指定
終端フラグ	ブール値	Out-of-band から指定

表 4.22 で示した入力可能数，出力可能数は処理の入力側，出力側に必要なフォーマットデータオブジェクトの数を表す．入力 Format 配列と出力 Format 配列は，実際に加工処理対象となるフォーマットデータオブジェクトである．

フォーマット処理オブジェクトはフォーマット依存になる前提であるが，フォーマットに非依存なフォーマット処理オブジェクトの派生オブジェクトとして，転送 (リダイレクト) 処理オブジェクトを定義する．その役割は，入力フォーマットデータオブジェクトを一

つ入力し，複数の出力フォーマットデータオブジェクトに対して，入力フォーマットデータオブジェクトの情報を複製することである．

本機構は，この転送処理オブジェクトと各フォーマットに依存した処理オブジェクトを連結することで，複数の入出力において個別の品質を扱うことに可能にする．

表 4.22 で述べた終端フラグは，この連結の終端を指し，連結の終端時にデータ保持領域表現情報の状態遷移を次の状態に遷移する．具体的には入力側のフォーマットデータオブジェクトは表 4.10 で示した READY_READ に，出力側のフォーマットデータオブジェクトは READY_WRITE になる．

本オブジェクトの制限事項はフォーマットデータオブジェクトのみを参照するため，フォーマットデータオブジェクトが持つ情報でしか処理が行えない．

フォーマット処理オブジェクト初期化時に必要な情報

フォーマット処理オブジェクト初期化時に必要な情報は，入力 Format 配列と出力 Format 配列となる．

4.2.4 セッション管理オブジェクト

セッション管理オブジェクトは，全体の多重入出力管理と時間管理を行う．本項は，セッション管理オブジェクトにおいて保持する情報，セッション管理オブジェクト初期化時に必要な情報に関して述べる．

セッション管理オブジェクトにおいて保持する情報

セッション管理オブジェクトでは表 4.23 に示した情報を保持する．

表 4.23: セッション管理オブジェクトが保持する情報

名称	型 (とりうる値)	指定箇所
入力 IO 配列	IO 配列	Out-of-band から指定
出力 IO 配列	IO 配列	Out-of-band から指定
Processor 配列	Processor 配列	Out-of-band から指定
並行処理フラグ	ブール値	処理内容に依存
入力及び加工用並行処理 ID	整数	処理内容に依存
出力用並行処理 ID	整数	処理内容に依存

セッション管理オブジェクトは，1 つのセッション管理オブジェクトに対して複数の入力側の入出力処理オブジェクトのインスタンス，複数の出力側の入出力処理オブジェクトのインスタンス，複数のフォーマット処理オブジェクトのインスタンスを保持する．

入力側の入出力処理オブジェクトのインスタンスが非同期性で，出力側の入出力処理オブジェクトのインスタンスが同期性を持つものが含まれている場合，出力側の消費に対して間に合わない状況が想定される．このような場合に並行処理が必要となる．表 4.23 内の並行処理フラグは並行処理をするか逐次処理をするかの状態の判別に用いる．

セッション管理オブジェクト初期化時に必要な情報

セッション管理オブジェクト初期化時に必要となる情報は，入力 IO 配列，出力 IO 配列，Processor 配列が必要となる．これらは Out-of-band から指定する．

4.3 本章のまとめ

本章では，入出力処理とフォーマットの処理と属性を分離する機構の設計を行った．本設計では，入出力に関連した処理と属性情報保持を担当する入出力処理オブジェクト，フォーマットのデータと付随する属性情報を保持するフォーマットデータオブジェクト，フォーマット依存した処理を担当するフォーマット処理オブジェクト，全体の管理を担当するセッション管理オブジェクトを基底のオブジェクトとして定めた．入出力処理オブジェクトとフォーマット処理オブジェクトが，フォーマットデータオブジェクトを参照することで，品質制御に関する情報を交換し，品質制御を可能とした．IEEE1394，RTP，ファイルの各入出力は入出力処理オブジェクトの派生オブジェクトとして表現される．DV フォーマット，MPEG2-TS の属性情報はフォーマットデータオブジェクトの派生オブジェクトとして表現される．なお各入出力においてフォーマット固有識別子と時間情報がフォーマットに依存する．この部分は各入出力の派生オブジェクト内に隠蔽する．次章において本章で述べた設計を基に実装を述べる．

第5章　入出力処理・フォーマット分離機構の実装

本章では前章の設計に基づき本アーキテクチャの実装に関して述べる．
実装環境は以下の通りである．

表 5.1: 実装環境

CPU インストラクションセット	x86 系 (Pentium M)
OS	Linux 2.4.26 Kernel
使用ライブラリ	libraw1394 0.10.1(IEEE1394 入出力ライブラリ)
プログラミング言語	C 言語 (gcc 3.3.2)

5.1　実装概要

本実装は VMT(Virtual Method Table) を利用して，異種の入出力に対して同一の処理インタフェースを提供する．VMT の例として入出力処理オブジェクトで用いる VMT を図 5.1 に示す．

VMT は関数ポインタを集めた構造体として表現できる．継承関係は子クラスとなる構造体の最初のメンバで継承元となる構造体変数を宣言することで成立する．例として IEEE1394 用入出力処理オブジェクトが入出力処理オブジェクトの基底オブジェクトを継承する方法を図 5.2 に示す．

この VMT をプリプロセッサによるマクロで隠蔽し，入出力処理オブジェクトの子クラスにあたるオブジェクトに対して同一の処理インタフェースを提供する．

本実装ではこの VMT とマクロによる隠蔽を入出力処理オブジェクト，フォーマットデータオブジェクト，フォーマット処理オブジェクトに用いている．次節以降，フォーマットデータオブジェクト，入出力処理オブジェクト，フォーマット処理オブジェクト，セッション管理オブジェクトの順に述べる．

```
typedef struct GavIo GavIo;
/** @struct GavIo_VMT is common virtual method table for Io */
typedef struct {
int (*open)(GavIo *this);
int (*close)(GavIo *this);
int (*read)(GavIo *this);
int (*write)(GavIo *this);
int (*allocBuffer)(GavIo *this);
void (*freeBuffer)(GavIo *this);
void (*detectFormat)(GavIo *this);
void (*getFdset)(GavIo *this, int *setmaxfd, fd_set *rfds, fd_set *wfds,
                        fd_set *efds);
void (*setFdset)(GavIo *this, fd_set rfds, fd_set wfds, fd_set efds)
int (*setFormat)(GavIo *this, GavFormat *format);
int (*setName)(GavIo *this, char *name);
int (*setPort)(GavIo *this, char *portstr);
} GavIo_VMT;
```

図 5.1: 入出力処理オブジェクト VMT

```
struct GavIeee1394Io{
GavIo super; ///< super class object
// addtional public property
int channel; ///< IEEE1394 Isochronous Channel
(以下略)
};
```

図 5.2: 継承の例

5.2 フォーマットデータオブジェクト

入出力処理オブジェクトと同様に VMT とマクロを利用し，同一の処理インタフェースを提供する．

フォーマットデータオブジェクトが保持するバッファは第 4 章で述べたデータ保持情報を元にしたデータ構造 *avData* を定義し，これを一単位として，複数を環状リストとして保持する．*avData* 構造体とそこから参照される第 4 章でデータ領域保持領域表現情報として述べた *DataVector* 構造体を図 5.4 に示す．

フォーマットデータオブジェクトの基底オブジェクトにおける変数及び関数を図 5.5 に示す．これらの関数は表 4.14 の情報に準じて作成している．

```
#define read_GavIo(this) (((GavIo *)(this))->vmt->read((GavIo *)(this)))
/** @def A macro to call close_GavIo vmt method as close */
#define write_GavIo(this) (((GavIo *)(this))->vmt->write((GavIo *)(this)))
/** @def A macro to call close_GavIo vmt method as allocBuffer */
```

図 5.3: 同一操作方法を提供するためのマクロ (部分抜粋)

```
typedef struct DataVector{
  char state;
  struct iovec *iov;
  char **fillFlag;
  int fillCount;
  int useCount;
  int iovcnt;
}DataVector;

typedef struct GavAvData {
  DataVector allData;
  DataVector videoData;
  DataVector
  audioData;
  DataVector systemData;
  char *dataBlock;
  struct timeval pts;     /// presentation time stamp
} GavAvData;
```

図 5.4: データ領域保持領域表現情報を表す DataVector 構造体とデータ保持情報を表す GavAvData 構造体

5.3 入出力処理オブジェクト

本章では入出力処理オブジェクトから継承される IEEE1394 と RTP の入出力処理に関して述べる．入出力処理オブジェクトの基底オブジェクトにおける変数及び関数を図 5.6 に示す．図 5.6 の上段が変数を示し，変数名，型名の順で表記している．下段が関数を示し，関数名，() 内に引き数，戻り値の順で表記している．図 5.6 では変数に関しては他から参照される変数のみを示し，属性情報つまり変数の値を設定もしくは取得する為の関数やマクロは省略して表記している．

図 5.6 に示した変数は，表 4.20 の情報に基本的に準じている．ただし実装の都合上，以下の変数を増やした．

- *select* システムコールの引き数となる setmaxfd, fd_sets,
- セッション管理オブジェクトで発生する休止処理時の入出力回数保持する lastInter-

図 5.5: フォーマットデータオブジェクトの基底オブジェクトの変数及び関数

valCount

- 休止時間の誤差値を保持する変数である lastGapTime

5.3.1 IEEE1394 固有部分

IEEE1394 の入出力処理は libraw1394 を介して行う．libraw1394 は Linux における IEEE1394 デバイスファイルである **/dev/raw1394** をユーザランドから扱うためのライブラリである．libraw1394 は raw1394_handle と呼ばれる特殊なデータ構造をファイルディスクリプタの代わりとして用いる．処理の流れは以下の通りになる．

図 5.6: 入出力処理オブジェクトの基底オブジェクトの変数及び関数

open

1. libraw1394 関数の *raw1394_new_handle* を用いて，新しい handle を取得する．

2. *raw1394_set_port* を用いて，port を設定する．

3. Out-of-band によって指定された入出力の方向に従って動作する．IEEE1394 インタフェースから読み込む場合は，*raw1394_iso_recv_init* を呼び出す．*raw1394_iso_recv_init* はそれぞれコールバック関数を引数にとる．IEEE1394 アイソクロナス転送においてパケットが転送されるときに引数にとられたコールバック関数が呼び出される．

read

IEEE1394 インタフェースからデータを受信する場合は，初回実行時に，*raw1394_iso_recv_start* を呼び出し，その後は *raw1394_loop_iterate* を呼び出す．*raw1394_loop_iterate* によって既に設定しているコールバック関数が呼び出される．

コールバック関数において脱カプセル化をしてデータをバッファに追加する処理が行われる．脱カプセル化処理は以下の手順でおこなわれる．

1. CIP ヘッダのポインタを取得

2. SPH フィールドの値から SPH の有無を判別

3. ペイロードのポインタの取得と *addData_GavFormat* による，フォーマットデータオブジェクトへのデータの追加

この手順に従ったフォーマット依存性のない脱カプセル化処理を図 5.7 に示す．

```
the_speSize = 0;
struct ieee1394ciphdr *ciphdr = (struct ieee1394ciphdr *)data;

if(ciphdr->sph == 1){
  sph = (struct sph *)((char *)ciphdr + sizeof(struct
                       ieee1394ciphdr));
  the_sphSize = sizeof(struct sph);
}

for(i = sizeof(struct ieee1394ciphdr);
    i < len; i += io->super.format->packetSize){
  iov.iov_base = &data[i + the_sphSize];
  iov.iov_len = io->super.format->packetSize;
  addData_GavFormat(io->super.format, &iov, 1 ,0);
}
```

図 5.7: フォーマット依存性のない IEEE1394 の脱カプセル化処理

write

IEEE1394 インタフェースへデータ送信を行う場合は，以下の処理を行い送信すべきデータを作成する．

1. フォーマットデータオブジェクトからのデータ取得

2. 参照しているフォーマットデータオブジェクトに依存した SPH とフォーマット依存フィールドの生成

作成されたデータを *raw1394_iso_xmit_write* を用いて，IEEE1394 インタフェースへ送信する．IEEE1394 はデータ量の調整のために，Empty CIP パケットを生成することがある．DV の場合は生成タイミングがあらかじめ算出可能なため一定回数毎に行う．入出力処理オブジェクト基底部分に定義されている入出力回数保持している変数 ioCount の値を調べ定期的に Empty CIP パケットを生成する．MPEG2-TS を扱う場合は規格上 IEEE1394 パケットの受信側機器に 3264byte の受信バッファがあることが前提となる [34]．DV と比較するとジッタ耐性があることから，フォーマットデータオブジェクト及びその派生オブジェクトのデータ取得関数である *getData_GavIo* の戻り値が 0 の場合に，Empty CIP パケットを生成する．

close

raw1394_iso_shutdown() により IEEE1394 のアイソクロナス転送を停止する．

5.3.2 RTP 固有部分

RTP は RTCP と対になり動作する．また RTP が行う通信の方向に関わらず RTCP は送受，つまり両方向の通信を行うため，複数のファイルディスクリプタを用いる必要がある．RTCP は RTP より通信頻度が低い．そのため RTCP を受信する場合はそのソケットのファイルディスクリプタを非ブロック IO にして用いるか，*select* または *poll* システムコールを用いて入出力の多重化をする必要がある．本研究では複数の異種の入出力を扱うことが前提としており，本実装では第 5.5 節で後述する，セッション管理オブジェクトで *select* システムコールを用いる．そのために入出力処理オブジェクトである RTP 入出力オブジェクトとセッション管理オブジェクトの間で *select* システムコールの引数となる fd_set 構造体を相互に交換し，これにより RTP と RTCP の多重化を実現する．

RTCP の送受信間隔は初期化関数 *init_GavRtpIo* で算出する．RTCP の送受信は RTP の送信または受信のたびに計測する経過時間が，RTCP の送受信間隔より大きくなっている場合に行われる．

open

ソケットファイルディスクリプタと送受に用いる sockaddr 構造体を *socket()* システムコールにより RTP 用に 1 つと RTCP の送受用に 2 つ生成する．また，RTCP のポート番号は RFC3550 で推奨されている通り RTP に用いるポート番号の値に 1 を加算した値を用いる．このとき指定した名前がマルチキャストアドレスであった場合，TTL とインタフェースを指定し，入出力方向が入力側であった場合にそのマルチキャストアドレスに参加する．

read

recvfrom() システムコールにより，データを受信し，RTP ヘッダを外し，フォーマットデータオブジェクト及びその派生オブジェクトのデータ追加関数である *addData_GavFormat* を呼び出しフォーマットデータオブジェクトにデータを追加する．

write

フォーマットデータオブジェクト及びその派生オブジェクトのデータ取得関数である *getData_GavFormat* を呼び出しフォーマットデータオブジェクトにデータを取得し RTP ヘッダを付加し，*sendto()* システムコールによりデータを送信する．

close

ファイルディスクリプタを閉じる．

5.4 フォーマット処理オブジェクト

フォーマット処理オブジェクトの基底オブジェクトにおける変数及び関数を図 5.8 に示す．表 4.22 の情報に準じて作成している．

5.5 セッション管理オブジェクト

セッション管理オブジェクトは時間管理を行いながら多重処理を実現する．セッション管理オブジェクトの基底オブジェクトにおける変数及び関数を図 5.9 に示す．表 4.23 の情報に準じて作成している．

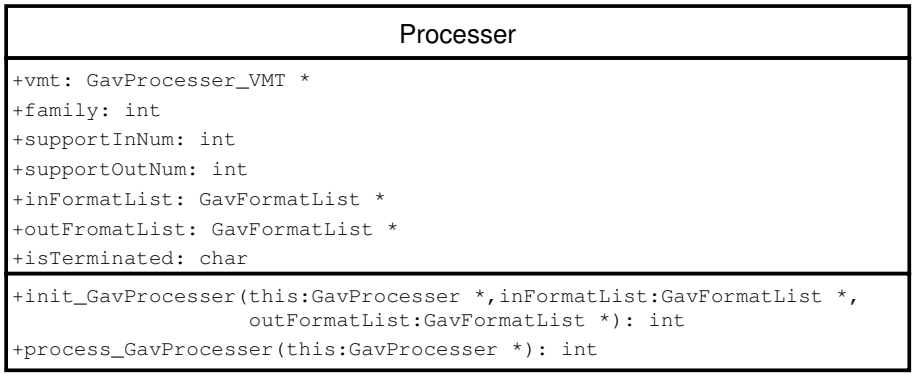

図 5.8: フォーマット処理オブジェクトの基底オブジェクトの変数及び関数

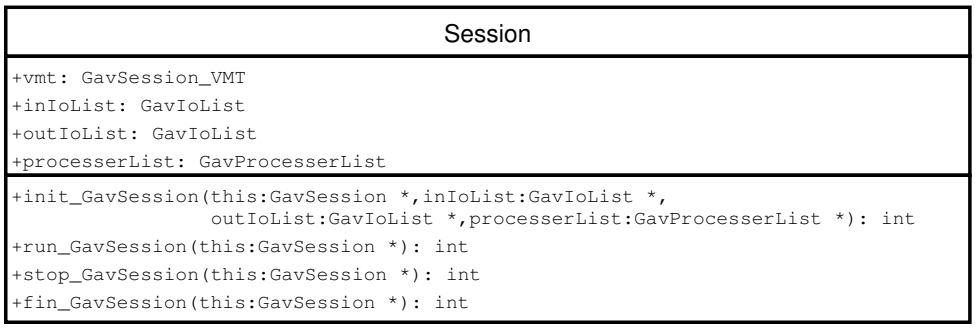

図 5.9: セッション管理オブジェクトの基底オブジェクトの変数及び関数

5.5.1 時間管理

入力か出力のどちらかに同期性を持った入出力オブジェクトのインスタンスがある場合は，そのインスタンスの入出力処理を時間管理に用いることで自律的に一定の処理間隔を保つことができる．しかし入出力同期性を持たない入力と出力を組合せ，逐次処理を行った場合に入出力非同期の問題が起きる．この場合において適切な出力間隔を保つために，リアルタイムクロックデバイスを用いる方法と，OS の休止処理のシステムコールを用いる方法が挙げられる．前者は環境に依存するため，本実装ではまずリアルタイムクロックデバイスを用い，利用できなかった場合に OS の休止処理のシステムコールを用いる．

これらの休止処理は汎用 OS で実現する場合に以下の 2 項目の問題点がある．

- 休止処理可能な時間精度が求められる時間精度より大きくなることがある
- 休止による時間的誤差が発生する

事前に休止可能な時間を計測し，その時間精度になるまで休止を実際に行わないことと，休止時に誤差を計測し，次の休止処理にその誤差を考慮して休止時間を決定することで，上述の 2 項目の問題を解消した．

図 5.10 に実際の休止処理を示す．

```
if(session->altSyncFd > 0){ //use rtc
  unsigned long syncdata;
  for(i = 0; i< blockCount; i++){     // block
    if(read(session->altSyncFd, &syncdata, sizeof(syncdata)) < 0){
      perror("read altSyncFd");
    }
  }
} else if(session->altSyncBySleep == TRUE){
  if(blockCount >0){
    nanosleep(&sleepTime, NULL);
  }
}
```

図 5.10: 実際の休止処理部分

リアルタイムクロックデバイスが利用可能な場合は，リアルタイムクロックのデバイスファ イルに対して *ioctl* システムコールでピリオディックインタラプトを有効にして *read()* システムコールにより，処理をブロックさせ休止させる．リアルタイムクロックデバイスが利用不可能な場合は，*nanosleep()* システムコールを用いて処理を停止させる．リアルタイムクロックデバイスが利用不可能な場合として，以下の事項が挙げられる．

- デバイスファイルが存在しない
  リアルタイムクロックデバイスのデバイスファイルは Linux 上では /dev/rtc で存在するが，例えば FreeBSD では kernel module を別途作成しないと利用できず，NetBSD/i386 では存在しないため利用できない．本実装は Linux で行ったが，リアルタイムクロックのデバイスファイルを用いる実装だけを行うと移植性の低下につながる．

- ロックされる
  ある 1 プロセスで利用すると，そのプロセスによりロックされ，他のプロセスから利用できなくなる

5.5.2 多重入出力処理

図 4.4 に示した通り，本機構の入出力はセッション管理オブジェクトから各入出力オブジェクトに対して *read*，*write* を呼び出し，フォーマット処理オブジェクトのインスタンスへ *process* を呼び出すことで実現する．具体的には各入出力オブジェクトのインスタンスは入力，出力ともにリストに保持されており，このリストを順々に辿り VMT を参照にして各インスタンスにおいて同種のメソッドを呼び出す．実際の処理の一例として，リストに格納されている入力用の入出力処理オブジェクトのインスタンスに対して *select* を行った後に，*read* を呼び出す例を図 5.11 に示す．

図 5.11 内の *applyWOC_GavIoList* は引数でとったリストの中のメンバに対して指定された関数を実行する．

```
void setFdset_GavIoList(void *visitor, GavIo *io)
{
  GavSession *session = (GavSession *)visitor;
  setFdset_GavIo(io, &session->fd_sets);
}

void read_GavIoList(void *visitor, GavIo *io)
{
  read_GavIo(io);
}

int read_once(GavSession *this)
{
  if(select(this->maxfd + 1, &this->fd_sets.rfds, NULL, NULL,
    NULL) < 0){
    perror("select");
  }

  applyWOC_GavIoList(this->inIoList,
     (ApplyMethod_GavIoListApply)setFdset_GavIoList,
     (void *)this);

  applyWOC_GavIoList(this->inIoList,
     (ApplyMethod_GavIoListApply)read_GavIoList,
     (void *)this);
}
```

図 5.11: VMT を辿りメソッドを呼び出す例 (fd_sets 構造体の複製と read)

具体的には以下の処理を *read, write* を行う前に実行する．

1. *select* システムコールに必要な引数の取得
   入力用リスト，出力用リストのそれぞれ入っている各インスタンスから *applyWOC_GavIoList* を用いて *select* システムコールに必要な引数を取得する．*select* システムコールに用いる情報を返すための関数が *getFdset_GavIo* であり，この関数が *applyWOC_GavIoList* から呼ばれる．

2. *select* の実行

3. 各インスタンスへの結果の返還
   select の実行結果をら *applyWOC_GavIoList* を呼び出し，各インスタンスへ結果を返す．*select* システムコールの結果を各入出力処理オブジェクトのインスタンスに設定するための関数が *setFdset_GavIo* であり，この関数が *applyWOC_GavIoList* から呼ばれる．

4. 入出力処理の実行
   FD_ISSET を用いて入出力可能か検査し入出力処理を実行する．

5.6 サンプルアプリケーション

本章で述べてきた，入出力処理とデータフォーマットの組合せを実現する機構の実装を利用して，入出力処理オブジェクトに RTP, IEEE1394, ファイル，フォーマットデータオブジェクトに DV と MPEG，フォーマット処理オブジェクトに転送プロセッサを利用したサンプルアプリケーションを実装した．サンプルアプリケーションのコマンドインタフェースを図 5.12 に示す．

```
./gavotte -I 入力用パラメータ (URI) -O 出力用パラメータ (URI) [-F 利用フォーマット]
```

図 5.12: サンプルアプリケーションコマンドラインインタフェース

サンプルアプリケーションは入力用パラメータ，出力用パラメータは URI 形式を用いて表現する．各パラメータの URI 形式の書式はプロトコル名に”file”, “ieee1394”, “rtp” を，“?” 以降に属性を指定可能である．複数の入出力を扱う場合には”,” を区切り文字に用いる．図 5.13 に実際の使用例を 2 例示す．利用フォーマットに指定がなかった場合は，既定のフォーマットとして DV を利用する．

IEEE1394 からの入力された MPEG2-TS を 192.168.0.2 の 1234 番ポートに送信すると同時に，ファイル stored.m2ts に保存する．

```
./gavotte -I ieee1394:///dev/raw1394 \
          -O rtp://192.168.0.2:1234,file://stored.m2ts -F MPEG2TS
```

ファイル stored.dv からの DV 入力を 192.168.0.2 の 6000 番ポートと同時に 192.168.0.3 に対して mtu サイズを 1300 にして送信する．

```
./gavotte -I file://stored.dv \
          -O rtp://192.168.0.2:6000,rtp://192.168.0.3?mtu=1300
```

図 5.13: サンプルアプリケーションを利用する例

5.7 本章のまとめ

本章では入出力処理とフォーマットを分離する機構の実装に関して述べた．本実装はC言語を用いて行った．VMT とマクロの利用により，派生オブジェクトで表される各入出力，各フォーマットで一律化した呼び出しを可能にして，異種の入出力の同時利用を可能にした．サンプルアプリケーションでは URI 表記を用いることで利用時の各入出力の操作を統一した．次章では本機構の評価に関して述べる．

第6章　入出力処理・フォーマット分離機構の評価

第3.1項で述べた問題意識，第3.2項の本モデルの要求から以下の項目を評価項目とする．

- 定性的評価項目
  - 特に拡張性の観点から考えた場合の，既存のデジタル映像機器のフォーマットを無変換で利用する映像転送システムと，本機構を用いたアプリケーション開発の際に必要となる手続きの比較
  - 相互接続性
- 定量的評価項目
  - 再利用性の観点から考えた場合の，既存のデジタル映像転送システムと本実装のソースコード行数の比較
  - 映像転送システムとしての性能評価
    - 計算機資源に対するオーバヘッド (単一入出力時及び異種複数入出力時)
    - 処理時間

6.1　定性評価

本節では定性評価として，拡張性及び，拡張性に深く関係する再利用性の観点から，既存のデジタル映像機器のフォーマットを無変換で利用するデジタル映像転送システムと，本実装を用いた場合のアプリケーション開発における処理の手続きの違いを比較評価する．また第5.6で述べたサンプルアプリケーションの相互接続性に関して評価し，映像転送機構としての妥当性を評価する．

6.1.1　既存のデジタル映像機器のフォーマットを無変換で利用する映像転送システムとの手続きの違い

第5章で実装した本機構は一定の処理手順を呼び出すことで，複数の異種の入出力に対応する．本機構は異種の入出力に対して透過的かつ統一的な処理インタフェースを定めている．本機構を用いた場合の必要となる処理手続きの流れを図6.1に示す．

1. 入力用の入出力オブジェクトリストの初期化
2. 出力用の入出力オブジェクトリストの初期化
3. 以下を各入出力処理オブジェクトのインスタンス事に行う．
   - 3.1 インスタンスの生成と初期化
   - 3.2 フォーマットデータオブジェクトの初期化
   - 3.3 フォーマットデータオブジェクトのデータ領域の確保
   - 3.4 入出力処理インスタンスとフォーマットデータオブジェクトの関連付け
   - 3.5 入出力処理オブジェクトリストへの追加
4. フォーマット処理リストの初期化
5. 以下を各フォーマット処理オブジェクトのインスタンス事に行う．
   - 5.1 フォーマット処理オブジェクトのインスタンスの生成
   - 5.2 フォーマット処理オブジェクトリストへの追加
   - 5.3 必要な場合はフォーマット処理オブジェクトの終端フラグの設定
6. セッション管理オブジェクトの生成と初期化と入力用入出力オブジェクトリスト，出力用入出力オブジェクトリスト，フォーマット処理リストの関連付け
7. セッションの開始

図 6.1: 本機構を用いた場合の処理の手続きの流れ

一方，既存アプリケーションは特定のフォーマットと入出力毎に個別化されている．比較対象の既存アプリケーションとして実装が公開されている DVTS をとりあげる．DVTS の複数のアプリケーションの集合であり，各個別アプリケーションの処理の流れを表 6.1 に示す．表 6.1 内で示している dvsend, dvrecv, dvsave, dvplay はそれぞれ，IEEE1394 からの入力を受けネットワークへの出力を行うアプリケーション，ネットワークからの入力を受け IEEE1394 への出力を行うアプリケーション，IEEE1394 からの入力を受けファイルへの出力を行うアプリケーション，ファイルからの入力を受け IEEE1394 への出力を行うアプリケーションである．

本機構は大きくわけて 7 つの処理手続きを必要とし，さらに細分化するとフォーマットデータオブジェクトで 4 つの処理手続き，フォーマット処理オブジェクトで 2 つの処理手続きで構成されている．一方の既存アプリケーションである DVTS は 4 つのアプリケーションで全て個々の処理手続きが行われている．表 6.1 に述べた 4 アプリケーションは 19 通りの処理手続きで構成されている．既存のシステムが他のフォーマットや他の入出力イ

表 6.1: 既存アプリケーションである DVTS の処理の流れ

dvsend	dvrecv	dvsave	dvplay
IEEE1394 の初期化	UDP ソケットの生成	IEEE1394 の初期化	ファイルを開く
宛先アドレスリストの生成	マルチキャストの設定	ファイルを開く	IEEE1394 の初期化
宛先アドレスの追加	共有メモリの確保	メインループの実行	メインループの実行
UDP ソケットの生成	fork システムコール		
メインループの実行	共有メモリのアタッチ (入力側)		
	UDP からのデータ入力 (入力側)		
	IEEE1394 の初期化 (出力側)		
	IEEE1394 への出力 (出力側)		

ンタフェースや通信方法に対して対応する場合はさらに増える．

新たな入出力に対応する例として新しいトランスポートプロトコルである SCTP への対応を行う場合を想定する．本機構は *init*, *open*, *close*, *read*, *write* の 5 つの機能の追加の実装を行うことにより対応できる．一方既存の実装を用いて対応するには本研究の想定下では，少なくともネットワークからの入力を映像機器インタフェースである IEEE1394 へ出力するアプリケーション，ネットワークからの入力をファイルへ出力するアプリケーション，ファイルからの入力をネットワークへ出力するアプリケーション，映像機器インタフェースである IEEE1394 からの入力をネットワークへ出力するアプリケーション，ネットワークからの入力をネットワークへ出力するアプリケーションのネットワーク部分に関係する部分のそれぞれに対して変更を加える必要がある．さらにこれをフォーマット毎に行う必要がある．

フォーマットの数を F，入出力の数を I として一般化すると，既存の方法で機能を実現する場合，

$$F \times I^2$$

の対応が必要となり，本実装を用いると

$$F + I$$

の対応のみで実現できる．

第 4 章で焦点にあて第 5 章で実装を行ったフォーマットは DV と MPEG で 2 種類，入出力の数はファイル，映像機器のインタフェースである IEEE1394，ネットワークのプロ

トコルとして RTP/UDP の 3 種類を扱ったが，対応すべきフォーマット，入出力インタフェースや通信方法にあたるプロトコルが増加したときに本実装は非常に有用であるといえる．

以上から本機構は，処理の再利用性を向上し，システムの機能拡張性を簡易化した．

6.1.2 相互接続性

本節では第 5.6 項に示したサンプルアプリケーションと RFC に準拠した他のアプリケーションとの相互接続性を検証の結果を述べる．

DV フォーマット

DV フォーマットの RTP ペイロードフォーマットが定められている RFC3189 を実装したアプリケーションが DVTS である．本機構の入力に IEEE1394，出力に RTP を用いた場合に本機構の RTP の出力が Windows 版 DVTS で問題なく再生されることを確認した．

MPEG2-TS

MPEG2-TS の RTP ペイロードフォーマットが定められた RFC2250 に準拠した VLC(Video Lan Client)[35] において問題なく再生された．このスクリーンキャプチャを図 6.2 に示す．またパケットキャプチャツールである Ethereal[36] で本機構の RTP パケットの出力が正しく RFC2250 のパケットフォーマットに準拠していることを確認した．このスクリーンキャプチャを図 6.3 に示す．

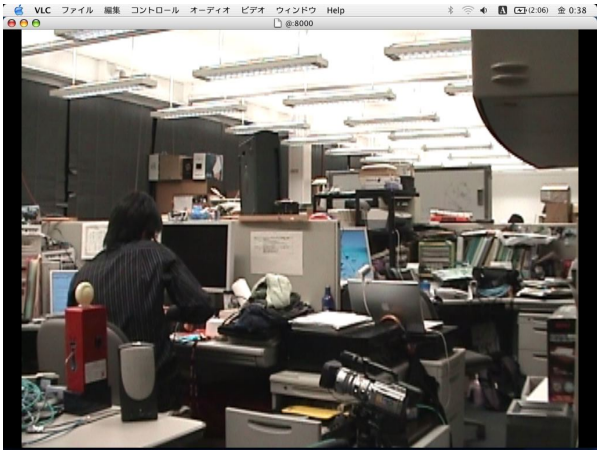

図 6.2: 他の MPEG2-TS を再生可能なアプリケーションである VLC で再生している様子

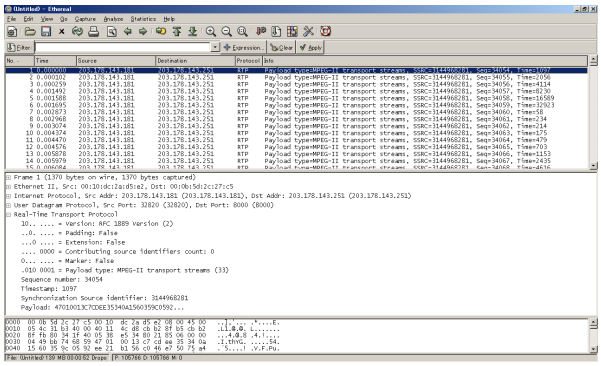

図 6.3: RFC2250 に準じて正しくデータが送受されている様子

6.2 定量評価

本節では，本機構と既存実装と開発量としてソースコードの行数を比較し拡張性に関係する再利用性における定量評価を行う．また，本機構の処理量に関して定量評価を行い，映像転送機構としての妥当性を評価する．

6.2.1 開発量の比較

開発にかかるコストの比較として，ソースコードの行数の比較を行う．DVTS はIEEE1394 からデータを受信しネットワークへ送信する dvsend，ネットワークからデータを受信し IEEE1394 へ送信する dvrecv IEEE1394 からデータを受信しファイルへ保存する dvsave ファイルからデータを読み出し IEEE1394 へ送信する dvplay が含まれる，本実装はこの全ての用途に利用できるため本基盤部分の実装及びサンプルアプリケーションの行数と，DVTS に同梱される dvsend, dvrecv, dvsave, dvplay の総和を比較した．

表 6.2: 本実装と DVTS のソースコード行数比較

サンプルアプリケーション部分	本実装 (基盤部分)	DVTS
134	4947	5341

本実装が同等以上の機能をより少ない行数で実現している．これは，各オブジェクトに機能分解し機能の重複部分を省いたためであり，機能の再利用性が向上したことを示している．第 3.1 項で述べた既存システムのシステム乏しい拡張性から発生していた，開発者に対する生産性の低下の問題を改善した．

6.2.2 既存のデジタル映像機器のフォーマットを無変換で転送する映像転送システムの実装と本実装の性能比較

本項では，第 5.6 項に述べた本実装を用いたサンプルアプリケーションと個別的に作られ特定のフォーマットと特定の入出力に特化した既存システムの実装との性能比較の結果に関して述べる．比較の対象となる既存の実装として一般に公開され広く利用されている DVTS を用いた．第 5 章で述べた本アーキテクチャの実装は同期性入力から非同期性出力に対して通信を行う場合は，逐次処理を用いて行う．一方，DVTS の送信アプリケーションである dvsend も逐次処理を用いて行う．そのため，特定の入出力とフォーマットに特化したアプリケーションである dvsend との比較により，本機構の各オブジェクト単位への機能の分解とその連結による処理の増分が計測できる．逆に非同期性入力と同期性出力を組み合わせた場合は並行処理として動作する．しかし本実装では pthread を用い，DVTS の受信アプリケーションである dvrecv は *fork* システムコールと共有メモリを用いているため，この 2 者の比較では正確な本実装の処理の増分が計測できない．このため本実装の

逐次処理を行う状態と dvsend の比較を行った．なお，dvsend は IEEE1394 に dv1394 デバイスドライバを用い，本実装は raw1394 デバイスドライバを用いるためデバイスドライバの実装の差が評価に影響する．そのため本評価用に IEEE1394 の入出力を dv1394 を用いるように本実装を変更し，評価を行った．

評価用いた計算機環境を表 6.3 に示す．

表 6.3: 評価用計算機環境

項目	内容
CPU	1268.95MHz
メモリ	512MB
OS	Linux 2.4.26 Kernel

処理コストの比較

評価手法は各プログラムを動作させその間に 1 秒毎に ps コマンドを実行した．ps コマンドの出力において，実時間に対してプロセス処理に使った時間の割合を示す CPU, 及び実メモリに対してそのプロセスが使ったメモリの割合を示す MEM の出力を利用した．DV カメラから IEEE1394 経由でデータを受け取り，1500 フレーム受けた時点で終了する．評価は，本サンプルアプリケーションにおいて，入力に IEEE1394 を用い，出力に RTP を用いた場合，入力に IEEE1394 を用い，出力に RTP とファイルを用いて多重入出力を行った場合，既存実装の dvsend を用いた場合を対象にして行った．

図 6.4 に約 50 秒間の CPU 使用率の遷移を，表 6.4 に CPU 使用率, メモリ使用率の約 50 秒間の値の平均の比較を示す．図 6.4 内の凡例，表 6.4 内の項目の “本実装 (rtp)“が入力に IEEE1394 を用い，出力に RTP を用いた場合，“本実装 (rtp+file)“が入力に IEEE1394 を用い，出力に RTP とファイルを用いて多重入出力を行った場合，“dvsend” が既存実装の dvsend を用いた場合を示している．

表 6.4: 本実装と既存実装 (dvsend) の使用計算機資源比較

	本実装 (rtp)	本実装 (rtp+file)	既存実装:dvsend	既存実装 (rtp+file)
CPU 使用率 (%)	6.152	10.948	3.806	対応不可
メモリ使用率 (%)	0.2	0.2	0.1	対応不可

本機構はおよそ処理量は 1.61 倍必要とし，メモリ領域は 2 倍必要とすることがわかった．また多重化したときの処理量はさらに 1.78 倍かかることがわかった．

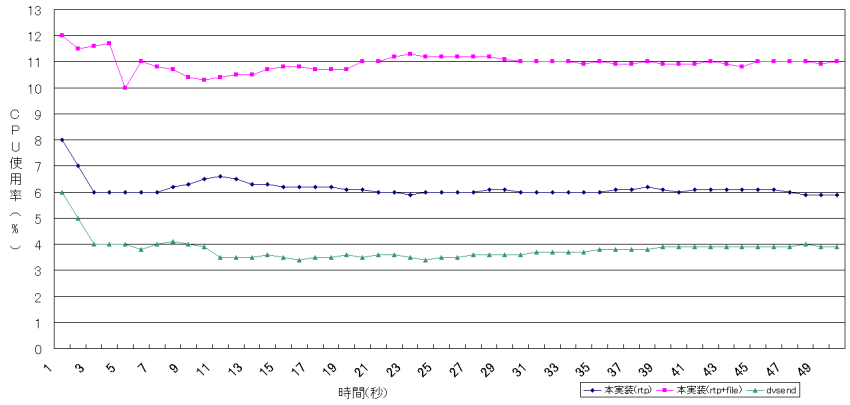

図 6.4: 本実装と既存実装 (dvsend) の処理における CPU 使用率の遷移

処理時間

先に示した計算処理量の増加と映像品質との関係を評価するために処理時間を計測した．この処理時間は遅延を始めとした品質に影響する．遅延時間は本論が対象としている映像機器が利用するフォーマットを無変換で転送する機構において重要な項目である．遅延時間は受信側のバッファ量によって左右されるため受信環境に応じて異なる．そのため送信側における，既存実装との処理内容の差異による処理時間差を計測した．計測には Unix システムコールの *gettimeofday* を用いた．計測結果を表 6.5 に示す．DV フォーマットを用いた比較であるので，評価項目として最初のデータ入力までの時間，最初のパケット送出までの時間，最初のフレーム送出までの時間，1 フレームにかかる平均送出時間を計測した．

表 6.5: 本実装と既存実装 (dvsend) の処理時間比較

	本実装 (μ 秒)	既存実装 (μ 秒)	差分 (μ 秒)	割合 (%)
最初のデータ入力	27383	224	27159	122.25
最初のパケット送出までの時間	84029	83177	852	1.01
最初のフレーム送出までの時間	90242	89295	947	1.01
1 フレームにかかる平均送出時間	6213	6118	95	1.02

表 6.5 の結果から，本機構が最初のデータ入力までの時間に大きな差があるが，その後の連続した処理ではほとんど差がないことを示している．最初のデータ入力までの時間に大きな差があるのは，図 6.1 と表 6.1 に示した処理の差の影響となる．表 6.3 に示した環境では，この差が 1 フレーム以内に収まっているため計測した処理量の増加が，映像品質

に影響しないことを示している．この計測結果を基に試算した結果，1032MHz を下回る CPU を用いた計算機では，実行時の最初の 1 フレーム分の映像が取得できない．しかしその後の映像転送には本実装の計算処理量の増加がほとんど影響しないことが示された．

6.3 本章のまとめ

本章は本機構の評価を行った．本章は第 5 章で述べた実装を用いて，既存のシステムと比較を行った．定性評価により，本機構の既存システムと比べて本機構は拡張が容易であることを示した．一方で定量評価により既存実装と比較して，本機構は計算処理量を増加させるが，その増加は映像の品質にはほとんど影響しないことが示された．

第7章　結論

7.1　まとめ

本研究は，映像機器が用いるデータフォーマットを中間的なフォーマットに変換することなく転送する映像転送システムにおいて，入出力処理とフォーマットに関する処理の分離による入出力機能の同時利用と拡張性の確保を実現した．映像機器が用いるデータフォーマットを変換することなく転送する既存の映像転送技術では，特定のフォーマットと特定の入出力または通信方法毎に，個別の映像転送システムによって実現されてきた．特定の入出力または通信方法と特定のフォーマットの組合せに個別化している映像転送システムは，機能の拡張性に乏しかった．利用の面においても，異種の入出力が同時に利用できないことや，その操作体系が統一されないことなどの機能的な問題があった．

本研究は，その解決として映像機器が扱うフォーマットの共通部分の定義，及びフォーマットが入出力に対して依存性のある部分と依存性のない部分の分離を行った．フォーマット間の共通部分，非共通部分を定義するにあたり，本研究は現在一般に用いられている大衆性のある民生用映像機器に用いられるフォーマットを対象とした．具体的にはフォーマットは DV フォーマットと MPEG2-TS を対象とした．また入出力の対象及び通信方法として，映像機器で標準的に利用される入出力インタフェースである IEEE1394 の同期転送モード，IP ネットワーク上で実時間情報の通信を行うための RTP，VoD に用いられる二次記憶装置に保存されたファイルを対象とした．これらのフォーマットと入出力のそれぞれに比較分析を行った結果，入出力がフォーマットに依存する部分は，各入出力のヘッダ情報における各フォーマット毎に算出方法の異なる時間情報となった．時間情報はそれぞれの入出力の対象とフォーマットの組合せ毎に計算方法が異なる．その他の情報は，共通のパラメータの保持によってフォーマットに依存なく表現できる．

本研究において実現された機構は複数の異種の入出力の対象に異なった品質で入出力を可能にするために，1) 入出力の属性及びを保持し処理を担当する部分，2) フォーマットに依存した属性を保持する部分，3) フォーマットに依存した処理を行う部分，4) 全体の多重入出力と時間管理を行う部分に機能を分割し設計，実装を行った．本機構の実現にあたり，入出力の属性及び処理を担当する部分は入出力非依存な一律した処理インタフェースを保持した．本機構はその一律化された処理インタフェースを，全体を統括する部分において複数の入出力に対して一斉呼び出し可能な機構を提供し，多重入出力処理を可能にした．

評価において，本機構は再利用性が高く，拡張性に富むことがわかった．定量評価により，特定のフォーマットと入出力の組み合わせのみで動作する既存の実装と比較して，本実装を利用したサンプルアプリケーションが必要とする計算機の処理量は既存実装の約

1.6 倍に増加することが示された．本機構は既存実装と比較して処理量の増大が起こるが，処理時間の計測によりその処理量の増大が映像の品質に対してほとんど影響しないことが示された．

本機構はフォーマット依存なく入出力の自由な組合せを可能にした．また本機構を利用したサンプルアプリケーションは統一された記述方法でフォーマット依存性なく複数の異種入出力を同時に利用可能にした．本機構は入出力とフォーマットに依存する部分を明確に分離したため，将来的に新しいフォーマットや新しい入出力インタフェースまたは IP ネットワークにおけるプロトコルが増えた場合に比較的容易に拡張可能である．本研究は個別的に作られ拡張性に乏しかった，映像転送機器で用いられるデータフォーマットを変換することなく利用する既存の映像転送アプリケーションと比較して，拡張性に富む機構を提案し実現した．

7.2 本研究の成果

本節では，本研究の成果に関して述べる．

7.2.1 開発者が本研究の成果を利用する場合の利点

フォーマットを変換することなく映像機器で用いるデータフォーマットを利用する映像転送システムを開発する場合，本機構を利用すると，映像転送に関係する部分は本機構で提供されるため，主にユーザインタフェースに相当する部分のみをを開発すれば良い．また，本機構を拡張する場合，即ち新しいフォーマットまたは新しい入出力や通信方法に対応する場合は，その新しく追加するフォーマットや入出力固有の部分を作成すれば良い．これまでの個別化して機能を提供してきたシステムでは，実現されて来た機能を再利用できないため，同等の機能を再開発する必要があった．

7.2.2 利用者が本研究の成果を利用する場合の利点

映像転送の利用者が本研究の成果を利用する場合は，複数の入出力の多重化可能であり，その操作が統一される利点がある．

7.3 今後の課題

本節では本研究の設計上の制限と，今後の課題に関して述べる．

7.3.1 本研究で提案した機構の制限

本研究はデジタル映像機器を直接利用した映像転送に着目し，実際に現在の民生用デジタル映像機器で用いられているフォーマットと入出力を分析して論じた．フォーマットに関しては DV フォーマットと MPEG2-TS に関して分析を行った．他のフォーマットに関しては以下の制約事項を満たせば対応可能である．

- 固定長のパケットが基本単位であること
  本機構が対象とした DV や MPEG2-TS は固定長のデータ構造を用いたフォーマットである．本機構は設計上，扱えるフォーマットが固定長のデータ構造になる．また映像・音声・システム情報のデータ量がセッションの途中で変更されない必要がある．一般に映像機器，特にテープメディアを使った機器はデータの蓄積を，固定長のデータを基本単位として行う．例えば最新のテープメディアである D-10[37] も，映像・音声・システム情報が多重化されており，テープ内でのデータ領域が固定長のデータを基本単位として記録されている．また，情報欠損が発生する可能性がある伝送路で用いる目的のために MPEG2-TS が固定長になっている通り，伝送用途にはデータ欠損を考慮して固定長フォーマットが用いられる傾向にある．そのため本研究の映像機器に用いられるデータフォーマットを無変換で転送する目的としては設計上妥当性がある．しかし，それ以外の用途のために作られたフォーマットは本機構での適応が難しい．

- 入出力部分に自己のデータ構造を写像しないフォーマットであること
  データフォーマット内の特定の情報を一部 RTP ペイロードフォーマットで表すフォーマットが存在する．それらは今回の設計では対応が出来ない．

7.3.2 今後の課題:Out-of-band に関する定義の欠如

本研究の第 4 章及び第 5 章では，In-band の情報に関して述べた．これは本論が，特定のフォーマットと入出力の組合せによる映像転送の実現に問題の焦点をあて，それを分離する方法に関して述べたためである．一方で，機能の連携を行う場合などに Out-of-band からの情報の指定が必要となるが，本論では，その機能に対して論じなかった．実運用などに用いるには本来 In-band を操作する部分として Out-of-band の情報の拡充が必要となる．特に一般に映像転送アプリケーションの情報は RTSP や SIP などで用いられる SDP によって記述されることが多い．本論では Out-of-band で交換すべき情報に関して十分に論じなかった．

謝辞

本論を進めるにあたり，主査である慶應義塾大学環境情報学部教授の村井純博士に感謝致します．また副査である慶應義塾大学環境情報学部助教授の中村修博士と慶應義塾大学環境情報学部の稲蔭正彦教授に感謝致します．

また御指導を頂きました，政策・メディア研究科助手の杉浦一徳博士，環境情報学部専任講師の重近範行博士に感謝致します．また私の拙い英語に御指導頂きました大喜多優博士に感謝致します．

そして常に近くで御助言頂きましたSFC研究所上席所員の小川浩司氏，慶應義塾政策・メディア研究科博士課程の今泉英明氏，海崎良氏，細やかな作業を手伝っていただきました慶應義塾政策・メディア研究科修士課程の久松剛氏，三島和宏氏，慶應義塾大学環境情報学部大薮勇輝氏，元日にも関わらず本論に指摘を下さったNTT-Communicationsの有賀征爾氏，MSN Messenger越しに常に相談に乗って下さいました東芝研究開発センターの菅沢延彦氏，ベトナム料理で励まして下さいました東芝研究開発センターの若山史郎氏，残留生活中における健康維持を目的としたの納豆同盟の鈴木貴晶氏，小椋康平氏，苦楽をともにした渡里雅史氏，小柴晋氏，岡田耕司氏，久松慎一氏，西原サヤ子氏に感謝致します．そしてモービル広域ネットワーク (MAUI) プロジェクト及び徳田・村井・中村・楠本・南研究室の諸氏，特にSTREAM研究グループの皆様に感謝致します．

最後に修士論文に取り組んだ期間をはじめとして，私の学生生活を常にサポートして下さった両親に感謝致します．

以上を持って謝辞といたします．

参考文献

[1] Microsoft Corporation. Windows Media Technology, Jan 2005. http://www.microsoft.com/windows/windowsmedia/default.aspx.

[2] Akimichi Ogawa. DVTS(Digital Video Transport Sysytem) Web Page, January 2005. http://www.sfc.wide.ad.jp/DVTS/.

[3] 小川 晃通. 協調的輻輳制御を用いた映像配信機構の設計と実装. 修士論文, 慶應義塾大学大学院政策・メディア研究科, 1999.

[4] 菅沢　延彦. IP マルチキャストを用いた放送型 VoD 機構の実現. 修士論文, 慶應義塾大学大学院政策・メディア研究科, 2002.

[5] 入野 仁志. MPEG2-TS over IP の設計と実装. 卒業論文, 慶應義塾大学環境情報学部, Feb. 2003.

[6] J. Postel. RFC 791: Internet Protocol, September 1981.

[7] J. Postel. RFC 768: User datagram protocol, August 1980.

[8] J. Postel. RFC 793: Transmission control protocol, September 1981.

[9] Audio-Video Transport Working Group, H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson. RFC 3550: RTP: A transport protocol for real-time applications, July 2003. Status: PROPOSED STANDARD.

[10] Audio-Video Transport Working Group and H. Schulzrinne. RFC 3551: RTP profile for audio and video conferences with minimal control, July 2003. Status: PROPOSED STANDARD.

[11] R. Stewart, Q. Xie, K. Morneault, C. Sharp, H. Schwarzbauer, T. Taylor, I. Rytina, M. Kalla, L. Zhang, and V. Paxson. RFC 2960: Stream Control Transmission Protocol, October 2000. PROPOSED STANDARD.

[12] T. Friedman, R. Caceres, A. Clark. RFC 3661: RTP Control Protocol Extended Reports (RTCP XR), November 2003. PROPOSED STANDARD.

[13] Christophe Diot and Brian Neil Levine and Bryan Lyles and Hassan Kassem and Doug Balensiefen. Deployment issues for the ip multicast service and architecture. In *IEEE Network Vol.14, num 1*, pages 78–88, 2000.

[14] Yang hua Chu, Sanjay G. Rao, and Hui Zhang. A case for end system multicast (keynote address). In *Proceedings of the 2000 ACM SIGMETRICS international conference on Measurement and modeling of computer systems*, pages 1–12. ACM Press, 2000.

[15] John Jannotti, David K. Gifford, M. Frans Kaashoek, and James W. O'Toole Jr. Overcast: Reliable multicasting with an overlay network. In *5th Symposium on Operating System Design and Implementation (OSDI)*, December 2000.

[16] Kohei Ogura, Hideaki Imaizumi, Nakamura Osamu, and Jun Murai. Overlay multicast protocol for delivering hierarchical structured data. In *12th Workshop on Distributed Processing System (SIG-DPS)*, December 2004.

[17] RealNetworks, January 2005. http://www.realnetworks.com.

[18] Institute of Electrical and Electronics Engineeers, Inc. IEEE Std 1394-1995 High Performance Serial Bus. *http://standards.ieee.org/reading/ieee/std/busarch/1394-1995.pdf*, Aug 1996.

[19] HD DIGITAL VCR CONFERENCE. Specifications of Consumer-Use Digital VCRs PART2 SD Specifications of Consumer-Use Digital VCRs. *Specifications of Consumer-Use Digital VCRs using 6.3mm magnetic tape*, pages 1–380, Dec 1994.

[20] K. Kobayashi, A. Ogawa, S. Casner, and C. Bormann. RTP Payload Format for DV (IEC 61834) Video, January 2002. Status: PROPOSED STANDARD.

[21] K. Kobayashi, A. Ogawa, S. Casner, and C. Bormann. RTP Payload Format for 12-bit DAT Audio and 20- and 24-bit Linear Sampled Audio, January 2002. Status: PROPOSED STANDARD.

[22] A.Ogawa K.Sugiura O.Nakamura J.Murai. Implementing tcp-friendliness in digital video over ip, Feb. 2002.

[23] Akimichi Ogawa, Kazunori Sugiura, Osamu Nakamura, Jun Murai. Enhancing the quality of dv over rtp with redundant audio transmission. *The International Conference on Information Networking 2003 Proceedings (vol.3)*, 2003.

[24] Edsger W. Dijkstra. The structure of the "the"-multiprogramming system. *In Commun. of the ACM*, 11:341–346, May 1968.

[25] Robst (robust streaming tools), January 2005. http://net.ipc.hiroshima-u.ac.jp/robst/.

[26] J. Rosenberg and H. Schulzrinne. An RTP Payload Format for Generic Forward Error Correction, December 1999. Status: PROPOSED STANDARD.

[27] D. Hoffman, G. Fernando, V. Goyal, and M. Civanlar. RFC 2250: RTP payload format for MPEG1/MPEG2 video, January 1998.

[28] Christopher J. Lindblad, David L. Tennenhouse. The VuSystem: A Programming System for Compute-Intensive Multimedia. *IEEE Journal on Selected Areas in Communications*, 14(y):1298–1313, September 1996.

[29] Microsoft Corporation. DirectShow for Windows XP, Jan 2005. http://msdn.microsoft.com/library/default.asp?url=/library/en-us/directshow/htm/introductiontodirectshow.asp.

[30] Java Media Framework API, Nov 2004. http://java.sun.com/products/java-media/jmf/index.jsp.

[31] Audio-Video Transport Working Group, H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson. RFC 1889: RTP: A transport protocol for real-time applications, January 1996. Status: PROPOSED STANDARD.

[32] 山田達也, 若宮直紀, 村田正幸, 宮原秀夫. 実時間動画像マルチキャストのための動画像品質調整機構の実装と評価 . In **電子情報通信学会技術研究報告** *(NS2002-60)*, pages 31–34, June 2002.

[33] G. Fairhurst and L. Wood. RFC 3366: Advice to link designers on link Automatic Repeat reQuest (ARQ), August 2002. PROPOSED STANDARD.

[34] HD DIGITAL VCR CONFERENCE. Specifications of Digital Interface for Consumer Electonic Audio/Video Equipment, PART4 Annex. *Specifications of Consumer-Use Digital VCRs using 6.3mm magnetic tape*, pages 1–10, Dec 1995.

[35] Videolan, January 2005. http://www.videolan.org/.

[36] Ethereal web page, January 2005. http://www.ethereal.com.

[37] The Society of Motion Picture and Television Engineers. Proposed SMPTE Standard for Television -Type D-10 Stream Specifications- MPEG2 4:2:2P@ML for 525/60 and 625/50. *SMPTE Journal*, Oct 2001.