第 3 部

リアルタイム通信

第 1 章

はじめに

今まで、Internet は主に TCP/IP プロトコル体系を基にしたパケット交換方式を用いた大規模なネットワークであった。しかし、ユーザの求める要求が多様化し、従来のコンピュータデータ処理にはなかった新しいアプリケーションがさまざま実現されるようになった。そのうちの一つの新たな要求はリアルタイム性、または実時間性であり、そのためネットワークの研究は時代を遡り、従来 IP では利用されていなかった仮想回線交換型の通信が注目されるようになった。

本稿では、インターネットにおけるリアルタイム通信の実現を目的としてる我々のワーキンググループの研究動向、およびリアルタイム通信の現状について述べる。

1.1 RT ワーキンググループについて

リアルタイム通信については、1990 年辺りにさまざまな研究機関が取り組みを開始し、現在では、いくつかのプロトコルはインターネットドラフトとしてかなり詳細な仕様が決まってきており、いくつかの実装例もある。このような時代において、WIDE Internet を利用したリアルタイム通信の実現、また IP だけに拘らず、次世代のリアルタイム通信アーキテクチャを実現するため、Real-Time ワーキンググループは、1994 年 9 月に形成された。まだ、活動を開始し間もなく、今後の研究方針が固まりつつある段階である。しかし、この分野は言うまでもなく、現在の流行りの一つなので、素早く取り込まないと時代遅れになってしまう。

1.2 リアルタイム通信プロトコルの必要性

計算機の高速化および、高帯域ネットワークの広域な普及によって、音声や映像といった連続メディアを広域ネットワークで送受信するアプリケーションが出現し、実用的に使われることが期待されている。現在、仮想マルチキャストネットワーク MBone 上では、実際に音声や映像の実時間的な転送を行なうことのできる、vat, nevot (音声), nv, vic (映像), ivs(音声と映像) などが開発されている。従来のアプリケーションはいかに情報を正確に伝

送するかが問題であったが、連続メディアを転送するアプリケーションは、いかにそれらの連続メディアの時間的制約を保持しつつ情報を伝送するかが問題となる。

連続メディア処理では、例えば、映像なら、1 秒間に 30 フレームの画像を転送する必要があるため、 33 ミリ秒の間隔で、1 フレームを周期的に伝送する必要がある。また、映像は、例えば、20 ミリ秒分の符号化された音声データをやはり周期的に転送する必要がある。もし、これらの連続メディア処理のある周期の処理の時間的制約が満たされなかった場合は、音声の途切れや映像の歪みが生じる。その時間的制約は、録音および再生を行なう計算機および、データの転送を行なうネットワークにおいても満たされる必要がある。そのため、連続メディア処理を行なうためには、計算機上だけではなく、ネットワーク上においても実時間処理を行なう必要がある。

これらの連続メディア処理で、必要なネットワーク、計算機資源の定量化を汎用的に行なうことができる。つまり、各周期時間を表すパラメータと、その周期内での処理量を表すパラメータである。従って、連続メディア処理の時間的制約を満たすためには、周期時間内の処理量を、周期時間で行なう実時間処理を行なえばよい。

この連続メディア処理の時間的制約を満たす一つの方法として、資源予約をあげることができる。資源予約に関しては、ATM や FDDI といった資源予約可能なデータリンク層を前提とした、資源予約管理のためのプロトコルが存在する。代表的なものとして、ST-II や、RSVP をあげることができる。これらの資源予約プロトコルとデータリンク層での資源予約とのマッピングにはまだ問題が残っている。

また、時間的制約を満たすためには、資源予約に頼らないでも、別のアプローチも考えられるであろう。

さらに、ネットワーク上での実時間処理によって、ネットワークは、ユーザに対してし保証できるサービスの品質 (QOS) を提示することが可能である。QOS パラメータとしては、帯域のみならず、遅延、遅延ゆらぎ、信頼性などが用意されている。ユーザはこのネットワークが提示する QOS を用いて、連続メディア処理以外の応用の可能性を見つけることができると考えられる。

また、品質の保証されたネットワーク上で、連続メディアの転送を行なう場合、個々の周期時間を制御する必要があり、そのためのプロトコルが必要である。現在、IETF Audio-Video Transport WG で標準化されている Real-Time Transfer Protocol(RTP) や カリフォルニア大学バークレー校の Tenet Group で開発された Real-Time Message Transport Protocol (RMTP) および、Continuous Medial Transport Protocol (CMTP) がある。これらの連続メディア様のプロトコルが、資源予約を前提としている下位層をいかに有効に使うかはまだ問題が残っていると考えられる。

1.3　ATM

ATM は技術は次世代の高速通信媒体として注目され、インターネット社会でも関心がかなり高まっている。しかし、ATM は単に高速通信媒体ではない。リアルタイム通信を実現するための優れた可能性をもち、これも今後開発されるリアルタイム通信プロトコルに大きな影響を及ぼす (もう既に及ぼしているかもしれない)。ここでは、簡単に ATM の概要について述べ、リアルタイム通信との関連を採り挙げる。

1.3.1　ATM の概要

ATM は、高帯域公衆網を実現するものとして設計されて来た。ATM の特徴は、固定長の短い (53 byte) セルを交換することと仮想コネクションが設定できそれぞれに対して異なる QOS 保証をできる事である。

固定長のセルを用いる事により、交換機のハード化を行い、高速な交換を行えるようにしている。現在、帯域としては、25Mbps,155Mbps が主流であるが、今後 622Mbps の物が出て来ている。

QOS クラスとしては、何も保証しない UBR(Unspecified Bit Rate) サービスから、帯域を予約出来る VBR(Variable Bit Rate),CBR(Constant Bit Rate) などのサービスがあり、それぞれの仮想コネクションに対して、設定する事ができる。

1.3.2　ATM とリアルタイム通信

ATM で IP を用いてリアルタイム通信を行うする時の問題点としては、IP の QOS 保証として考えられている Flow Spec と ATM の QOS パラメータのマッピング方法や画像と音声を異なる仮想コネクションに流した場合に、画像と音声を同時に再現するための同時性の機構を考える必要がある。

1.4　リアルタイム通信プロトコルの基礎

この章では、リアルタイム通信の研究では、最近標準用語として使われているキーワードについて解説する。第 では、リアルタイム通信の現状について述べ、今まで提案されてきたプロトコルについても解説するが、その予備知識も兼ねている。

1.4.1　資源予約

資源予約 (resource reservation) とは、通常、通信に必要なシステムの資源を通信が実際に行われる前に予約、すなわちあらかじめ確保しておくことを示す。

通常、予約する必要がある資源には、以下のものが挙げられる。

- プロセッシング時間 (CPU 時間)
- ネットワークバッファ (メモリ、キュー)
- ネットワークバンド幅

1.4.2 アドミッション制御

アドミッション制御 (admission control) とは、通常、資源予約機構と共に利用される。すなわち、あるユーザ、またはアプリケーションプログラムが通信に必要とする資源の量を、システムの状況に応じて提供可能であるか決定する機構である。

1.4.3 QOS

QOS は、Quality of Service の略であり、通常日本語ではサービスの品質として訳される。現在では、お馴染みのキーワードとなってきてるが、さまざまな捉え方をできる複雑な意味も秘められている。通常は、あるユーザ、またはアプリケーションプログラムに対して、ネットワークアーキテクチャが保証できるサービスを複数のパラメータの指定により提供する際のパラメータの抽象化法である。したがって、QOS とは通常ユーザが何らかのパラメータを指定し、システムはそれをシステム動作および資源にマッピングし、提供する。場合によっては、前節で述べたアドミッション制御を行う。いくつかのリアルタイム通信プロトコルでは、これらのパラメータのことを flowspec と呼んでいる。

第 2 章

リアルタイム通信プロトコルの現状

本章では、リアルタイム通信プロトコルの現在の研究動向について述べる。前章で述べた理由から TCP/IP プロトコル体系や OSI プロトコル体系では不十分であったリアルタイム通信のサポートが必要になり、ネットワークアーキテクチャおよびプロトコル体系に新たな機構を採り入れたものが研究成果、また、通信規約として発表されている。

通常、新たに提案されたリアルタイムプロトコルは、OSI 参照モデルの 第 3 層、または第 4 層、すなわちネットワーク層またはトランスポート層をターゲットとしている。また、これらのプロトコルは主に以下の二つのものに分類することが可能である。

1. データ転送の高速化を重視したプロトコル

2. 資源予約を採り入れたプロトコル

前者に挙げたデータ転送の高速化を重視したプロトコルでは、時間制約を満たす必要があるデータを特別に処理し転送効率の向上を図る方式を採る。主に、VMTP [1] や XTP [2, 3] などがこのタイプとして挙げられる。これらのプロトコルでは、ネットワーク処理における遅延時間を最小にするため優先度機構が導入されている。VMTP については、2.2 節、XTP については、2.1 節で述べる。なお、VMTP はトランスポート層プロトコルであり、XTP は、ネットワーク層とトランスポート層の両方を統合している。この他にも、2.3 節ではトランスポート層プロトコルである RTP について解説する。

後者に挙げたプロトコルでは、通信セッション毎に必要な資源 (ネットワーク帯域・バッファ・CPU サイクル) を予約することによって、デッドライン、ジッタの上限、スループットなどを保証する。この方法の場合、予約された資源の利用状況が変化しなければ、リアルタイム性を保証することができる。例として、Session Reservation Protocol (SRP) [4]、Tenet プロジェクトの RTIP を基盤としたプロトコル体系 [5]、ST-2 [6]、RSVP [7] などが挙げられる。これらのコネクション指向ネットワーク・プロトコルは、ストリーム、セッションまたはチャネルと呼ばれるコネクションを通信前に確立することによって資源を予約する機構を実現している。また、ユーザは自分の必要とする QOS を指定できること、さらにそれを通信時に動的にできることを望むことなどが重視されている。リアルタイム通信チャネルをインターネット用に初めて提案したものとして、SRP と Ferrari の研究 [8] があるが、これらは現在、Tenet プロジェクトで引き継がれ研究されている。

本章では、CBSRP について 2.4 節、ST-2 について 2.5 節、RSVP について 2.6 節、Tenet プロトコル体系について 2.7 節で詳細を述べる。

2.1 Xpress Transfer Protocol: XTP

2.1.1 XTP の概要

Xpress Transfer Protocol(XTP) [2, 3] は、Protocol Engines Inc. が開発した高速なネットワークで高いスループットを得ることを目的とした通信プロトコルである。

- オーバヘッドの小さいコネクション確立
- レート制御
- マルチキャスト通信
- 選択的再送機能
- リアルタイム転送
- データグラム転送
- 複数アドレス形式のサポート
- ルーティング機能

XTP は、ネットワーク層とトランスポート層の機能を統合することによって処理を簡略化している。

XTP は、ホスト間でコネクションを確立し、各ホスト上での XTP の状態情報を交換しあうことによって信頼性の高いデータ転送を行なう。

ここでは、XTP の基礎とコネクション管理、およびマルチキャストの機能について簡単に解説する。

2.1.2 XTP パケット

XTP パケットは、40 バイトヘッダ、可変長中間セグメント、4 バイトトレーラから構成される。ヘッダには、使用するモードや機能、パケットの制御を行なう制御ビットフィールドが含まれる。中間セグメントは、情報セグメントか制御セグメントからなる。制御セグメントからなる場合、パケットは制御パケットと見なされ、XTP の状態を伝達するために利用される。情報セグメントには、上位層のプロトコルデータが含まれる。トレーラはチェックサム計算に使用される。

データパケットには FIRST、DATA、PATH、DIAG、MAINT、MGMT、ROUTE の 6 個のタイプがある。コネクションの確立は FIRST パケットを用いて行なう。ユーザデータは DATA パケットで送信するが、FIRST パケットで送信することもできる。PATH パケットは、アソシエーションが既に存在し、変更が必要な時に利用される。DIAG パケットは、異常状態、またはエラーを通知するために用いられる。

制御パケットは状態情報の交換、コネクションを削除するために利用する。制御パケットには CNTL、RCNTL と 2 つのタイプがある。

2.1.3　コネクション管理

XTP では、エンドシステム同士で、全二重双方向仮想回線を確率する。各エンドシステムに保持される XTP 状態のことをコンテキストと呼び、二つ以上の通信コンテキストのことをアソシエーションと呼ぶ。データ転送は、バイトストリームを基本とする方式を用いるが、送信側では、透過的に記録境界をストリームに設定することが可能であり、これは受信側で充実に再現される。コンテキストは FIRST パケットの送受信により生成される。コンテキストにはキーが割り当てられ、このキーを含んだパケットを送受信することによって状態情報の交換を行なう。データ受信の確認応答はパケットを受信するたびに行なうのではなく、送信側から要求された場合に CNTL パケットを返送することによって行なう。XTP では、コネクション確立のオーバヘッドを削減するために、FIRST パケットをユーザデータと共に送信することができる。

2.1.4　データ転送

高速なネットワークでは、受信側の処理能力以上でパケットが到着すると、パケットロスが生じ、それに伴う誤り処理などによってスループットが低下する恐れがある。

そのため、XTP では輻輳制御として送信データ量を制御するフロー制御の他に、転送速度を制御するレート制御機構を導入している。これによって送信側と受信側の処理能力の違いから生じるパケットロスを減らし、スループットを向上する。このレート制御機構はリアルタイム通信に非常に重要である。

また、XTP のフロー制御はバイトシーケンス番号 (32 bit) によるスライディングウィンドウ方式を採用している。

レート制御では、一定時間内に送信したバイト数が、(コネクション毎に設定される最大値を越えた時点で送信を中断し、一定時間経過後、再び送信を開始することによって転送速度を調整する。またこのレート制御は RTIMER と credit という二つの内部変数を用いて実現されている。Credit は送信したデータのバイト数を示し、RTIMER は credit を更新する間隔を表す。これらの値は、次に示す制御パケット中の二つのフィールドの値から計算される。

- rate : 1 秒あたりに送信する最大バイト数

- burst : 1 burst(複数パケットから成る) で送信される最大バイト数

データを送信する度に送信したバイト数を credit から引き、値が負になったときに送信を中止する。RTIMER 経過後に credit を次のように更新する。

1. credit 値が 0 または負の場合、burst の値を加える
2. credit が正の場合、credit=burst とする

フロー制御およびレート制御は、パケットヘッダ中のフラグにより設定することができる。
また、XTP ではリアルタイム通信を行なうために優先度機構が用意され、ヘッダ中のフラグを指定することによって利用できる。優先度を設定するために 32bit 用意されている。

2.1.5 マルチキャスト機能

XTP では、信頼性のあるマルチキャスト機能を提供する。特に、マルチキャストアドレッシング方式を定義しておらず、IP クラス D アドレス や MAC アドレスが使用される。
送信者から送られた制御パケットに対する応答は、送信者でなくマルチキャストグループに対して行なわれる。この場合、同一パケットが各受信者に送られ、ネットワークの負荷を増大することになるため、これを回避するために damping/slotting アルゴリズムが使用される。また、送信者側でも多数の受信者からの制御パケットを処理するために bucket アルゴリズムが用意されている。

2.2 Versatile Message Transaction Protocol: VMTP

2.2.1 VMTP の概要

Versatile Message Transaction Protocol (VMTP) [1] は、分散環境に必要不可欠である高性能なネットワーク通信機能を統合して提供する汎用プロトコルである。VMTP は、Stanford 大学で開発されたプロトコルで、RFC [9] も用意されている。以下に VMTP が提供する主な機能を示す。

- ホストアドレスに依存しないネーミング機構
- 多重パケット要求および応答メッセージ
- 選択的再配送機構
- さまざまな暗号化機構に対応する安全なメッセージトランザクション機構
- マルチキャストメッセージトランザクション

- サーバオーバヘッドおよび状態を最小限にとどめたべき等 (idempotent) メッセージトランザクションを利用するリアルタイム通信

VMTP はメッセージトランザクションという機構を基本とし、ネットワークに存在する VMTP エンティティ間のトランスポート通信を実現している。メッセージトランザクションは、クライアントエンティティが一つ以上のサーバプロセスに送信する要求メッセージ、および一つのサーバエンティティ毎に最大 1 回 (at most once) 送信される 0 個以上の応答メッセージからなる。例えば、クライアントが単一のサーバに要求メッセージを送ると、一つの応答メッセージを受信する。マルチキャスト通信は、このようなサーバのグループに対して要求メッセージを送信することによって実現されている。

VMTP は、トランスポート層用のプロトコルであり、上位層である RPC プレゼンテーション層と統合して利用する。また、信頼性を保証しないデータグラムネットワークとインターネットワーク機能を提供する下位層を想定している。

ここでは、簡単に VMTP の基礎について述べ、その後リアルタイム通信用のサポート、マルチキャスト機能、および耐故障性についてまとめてある。

2.2.2　エンティティ識別子

VMTP のエンティティは、64 ビット長の識別子を用いて指定される。この識別子は、一意的に割り当てられるもので、ホストアドレスに依存しない。64 ビット長とは、既存の TCP また TP4 と比較すると長いが、ネットワーク層のアドレッシングに依存する必要がなく、エンティティのマイグレーションを簡単に実現することができ、移動ホストおよび多重ホーム型ホストなどの処理を提供することができる。エンティティ識別子の一部は、あるグループを表すグループ識別子のために利用される。VMTP はエンティティグループの作成、問い合わせ、変更用の管理プロトコルも提供する。

2.2.3　メッセージ構造

VMTP の要求/応答メッセージは、一つ以上のパケットグループとして転送される。パケットグループは同じトランザクション識別子とメッセージ制御ブロックを含む一つ以上のパケットの組を表す。

2.2.4　リアルタイム通信用機能

VMTP では、メッセージの処理リアルタイムに行うために優先度機構が提供されている。主に、以下に示す 4 段階からなる。

- 1100 緊急 (urgent/emergency)
- 1000 重要 (important)

- 0000 通常 (normal)
- 0100 バックグラウンド (background)

以上の優先度をパケットフォーマットで用意されている 4 ビットの`Priority` 識別子で指定し、要求メッセージの送信および受信はこの優先度に従って処理される。4 ビットの上位ビットは、符合ビットとしてみなされ、値が低いほうが優先度が高くなる。下位の二ビットは、各レベルにおいてさらに細かい優先順位を設けるために用意されている。したがって、実装で 4 ビットを全て利用する場合、最大 16 レベルの優先度を設けることが可能となる。

さらに、特別なデータグラム Request と呼ばれる要求メッセージを送ることによって、クライアントは再送、確認応答などの機能を利用せずブロックしない通信を行なうことができる。このデータグラム Request は通常 (ストリームモードでない場合) 同一クライアントがその前に送信した全ての要求メッセージより優先される。

最後に、リアルタイム処理を実現するために、以下の機能を提供し、プロトコル処理のオーバヘッドを低下させている。

- Conditional Message Delivery (CMD) 条件メッセージ配送
- Header Checksum Only (HCO) ヘッダチェックサムのみ
- No Retransmission (NRT) 再送を行なわない

以上の機能は、パケット中の制御フラグを用いて指定することができる。

CMD フラグを指定すると、要求を受け取った瞬間に受信側がその要求を処理できない場合 (その要求メッセージ待ち状態でなかった場合)、要求は無視される (応答の場合も同様である)。HCO フラグを指定すると、VMTP ヘッダのみに対して、チェックサムを計算する (通常はデータセグメントを含むパケット全体に対してチェックサムを計算する)。これにより、音声、または画像データを送信する際に生じるオーバーヘッドを低下することができる。NRT を指定すると、メッセージ受信側は破損したメッセージを部分的に受け取った場合、再送要求を出さず不完全なメッセージを利用するか、または全く放棄する。

この 3 つの機構は新しいプロトコル状態を加えることなく、ビット操作で処理を済ませることができる。

2.2.5 マルチキャスト通信

VMTP は、実時間アプリケーションを実現するために非常に重要であるマルチキャスト通信を提供している。

前述のエンティティ識別子を用いてエンティティグループを指定することが可能である。グループに対する通信は、マルチキャスト機構を用いて行なわれ、最低一つの応答メッセー

ジが返されるまで、またはエラータイムアウトが発生するまで、要求は送り続けられる。クライアントは一つの要求に対してグループに属す各エンティティから応答を複数受信することができる。

VMTP のマルチキャスト機構は信頼性を考慮していない。信頼性を維持するマルチキャストを提供するにはコストが多くかかり、通常のアプリケーションはそれを必要とせず VMTP の多重応答機能を利用する実装に応じて、考慮することが可能である。

しかし VMTP は、ポジティブ確認応答を利用する信頼性のあるマルチキャスト機構を提供するための機能を用意している。この機能を利用して、信頼性を考慮したアプリケーションを実装することができる。例えば、各クライアントはグループに属す全エンティティの情報を保持し、全てのメンバから確認応答が返されるまで、要求メッセージを送り続けるように実装を変更することができる。

最後に VMTP には、グループ応答機構が実験的に導入されている。グループ応答メッセージは、マルチキャスト要求を受信したサーバが、要求メッセージを送信した元のクライアントとマルチキャスト要求が送られたグループの全メンバに返す特別な応答メッセージである。ヘッダ中の MDG ビットによって、クライアントが要求メッセージを送信したグループに属していることがわかる。したがって、サーバはクライアントおよびサーバグループに応答メッセージを返す際に、このビットを参照し、必要に応じて送り先が異なる要求メッセージをもう一つ作成する。サーバはグループ応答メッセージを受け取ることを明示的に指定する必要があり、通常の場合、クライアントから要求を受け取り応答を返していないときのみ他のサーバからのグループ応答を受け取る。

2.2.6　耐故障性

耐故障性を考慮した実時間分散システムに VMTP を応用するために耐故障性を維持するための拡張なども考えられている。VMTP プロトコル仕様 [9] には、クライアントサーバで交換されるメッセージのログを記録するという方法が述べてある。ネットワーク中のノードが故障した場合、ログに従って再びメッセージの交換を行なう。VMTP は、実験的に`NotifyVmtpClient` および `NotifyVmtpServer` 命令に Receive Sequence Number フィールドを提供している。また、応答パケットに Next Receive Sequence (NRS) フラグも追加され、送信側は送信するメッセージ毎に対して受信シーケンス番号をログに記録することが可能となる。故障が起きると、以前受信したメッセージを再び同じ順番に従って受信することが可能であり、故障発生前の状態に復帰することができる。

2.3 RTP: A Transport Protocol for Real-Time Application

2.3.1 RTP の概要

Real-Time Tranport Protocol (RTP) [10] は、音声や映像といった時間的制約を持つ対話的な音声や映像を end-to-end に配送するサービスを提供する。しかし、RTP 自身は、時間的制約や QOS を保証する機構を一切提供せず、そのような保証は下位レイヤに任せている。

当初、RTP は複数地点間のマルチメディア会議を支援するために設計されたが、その適用範囲はそのような特定のアプリケーションだけとは限らず、例えば、蓄積された連続メディア、対話的分散シミュレーションといったアプリケーションにも適用可能である。

RTP は以下の 2 つのパートから成る。

- Real-time transport protocol (RTP)

  リアルタイム性のあるデータを転送するためのプロトコル

- RTP control protocol (RTCP)

  QOS の監視、セッションの参加者の情報を伝達するためのプロトコル

2.3.2 データ転送プロトコル - RTP -

RTP のヘッダには、バージョン、ペイロードタイプ、SSRC, CSRC などのフィールドが含まれている。

現在のバージョン は 2 である。最初の RTP の仕様で実装されている nv や ivs などは 1 である。さらにその前に実装された vat は 0 である。vic は、version 1 のツールとの相互接続を考慮してオプションによって、 1 と 2 に切替えることができる。

ヘッダ中のペイロードタイプ (PT) の定義やペイロードフォーマット（メディアの符号化）へのマッピングは、参考文献 [11] を参照すること。

SSRC は同一のセッションで同一な番号が存在しないように決められる。基本的には乱数で決める。SSRC は、例えば、同一セッションにあるホストから異なる複数のメディア、例えば、音声と動画を同時に送出した時に、それらのパケットを区別するために用いられる。

CSRC は後述する Mixer によって加えられてゆく。

2.3.3 制御プロトコル - RTCP -

制御プロトコルパケットには以下の型がある。

- SR : Sender Report

  データの送信している参加者の統計情報

- RR : Receiver Report

  データ送信をしていない参加者の統計情報

- SDES : Source description

  Source description の項目には、CNAME, NAME, EMAIL, PHONE, LOC, TOOL NOTE, PRIV がある。

- BYE
  セッションから抜ける

- APP

  アプリケーション独自の仕様

2.3.4 Mixer と Translator

色々な回線速度のネットワークからなるセッションの参加者が音声フォーマットを選ぶ時、低帯域の回線を気にする必要がある。こういった場合、細い回線に流れ込む異なる送り手からの複数の音声ストリームを同期させ、 1 つにまとめる Mixer が用意されている。

逆に、高帯域の回線で音声会議には参加したいが、firewall によって、直接IP multicast で接接続できない参加者のために、translator が用意されている。この場合、2 本の translator を用いる。1 本は firewall の外側からマルチキャストパケットを流し込むのに用いられ、もう一本は、内側からマルチキャストパケットを適切なマルチキャストグループに流し込むのに用いられる。

2.4 Capacity Based SRP (CBSRP)

Capacity Based SRP (CBSRP) [12] は、単一 FDDI ネットワークセグメント内の動的 QOS 制御機構を導入している。CBSRP は、FDDI の同期モードを利用して実際のネットワーク帯域を予約する機構を実装している。CBSRP の主な特徴としては、QOS を主に以下の二つのものに区別している。

- 時間的解像度 (temporal resolution)

- 空間的解像度 (spatial resolution)

ユーザは、この二つの属性を表す QOS クラスを複数個指定することによって、通信セッション毎に必要とする品質を得る。

CBSRP でユーザが指定可能な QOS パラメータを以下に示す。

- 最小 C_{spt}
- 最大 C_{spt}
- s_res[MAX_SPT]
- 最小 C_{tmp}
- 最大 C_{tmp}
- t_res[MAX_TMP]
- 重要度
- エンド–エンドの最大遅延時間
- 最大パケット損失レート

ここで、 C_{spt} と C_{tmp} は、それぞれ時間的解像度および空間的解像度の任意のクラスを表す。MAX_SPT と MAX_TMP は、それぞれシステムで定義されている時間的解像度および空間的解像度のクラス数である。s_res と t_res は、各 C_{spt} の一周期毎のデータサイズと各 C_{tmp} におけるサンプリングレートを表す。また、重要度は、各セッションの重要度を表し、C_{spt} と C_{tmp} を求めるために用いられる。

C_{spt} と C_{tmp} は、QOS の初期値を求めるため、およびセッションが有効である際に最低必要条件が維持されることを保証するために重要である。

セッション生成時に、二つの配列 s_res[MAX_SPT] と t_res[MAX_TMP] を用いてデータ転送に必要な時間的および空間的離散値を指定する。この時間的および空間的値の組合せがセッション用のシステム資源必要条件を定義することになる。D_{user} と R_{user} がそれぞれ、s_res[C_{spt}] と t_res[C_{tmp}] を表すとする。$D_{user}R_{user}$ の積が大きくなると、必要とする容量も増加する。したがって、受け入れ可能な C_{spt} と C_{tmp} は、ユーザが要求するパラメータおよびシステムの容量によって決定される。

2.4.1 QOS 変換

前節で述べた QOS パラメータは、実時間通信に必要な資源を簡単に確保するために用意された。ユーザが指定するこれらのパラメータは、システム内のプロセッサ資源、メモリ資源に関連する情報、およびさらに低いレベルにおけるプロトコル属性値に変換される。

例えば、与えられたフレームレートを、FDDI フレームの周期的転送に変換するには以下の計算が行なわれる。ユーザのビデオフレームストリームの周期 P_{user} ($1/R_{user}$ に等しい) とメディアアクセス制御 (MAC) の周期 P_{MAC} が常に同じであるとは限らない。したがって、MAC における一周期毎のデータサイズ D_{MAC} を P_{user} と D_{user} に基づいて計算する必要がある。FDDI を用いた場合、一つのフレームの最大長が 4500 バイトなので、フラグメント化する必要が生じる可能性がある。しかし、フラグメント化された各パケットは、プロトコル処理用のヘッダとトレーラ D_{proc} が必要である (この場合、56 バイト)。したがって、ユーザデータ D_{user} の最大値は、4500 バイト$-D_{proc}$ を越えてはいけない。

D_{MAC} を計算することができると、ユーザが要求する容量 D_{user}/P_{user} を D_{MAC} の値に沿って一次元的に順序をつけることができ、C_{res} と C_{prd} は、一次元のクラス $C_{session}$ に変換することができる。QOS 制御を動的に行なうため $C_{session}$ と D_{MAC} を利用する。

2.5 Internet Stream Protocol, Version 2: ST-2

2.5.1 概要

ST-2 [6] は TCP/IP Internet アーキテクチャのもとで、固有の広帯域ネットワーク技術やハードウェアに依存せず、パケット化した動画、音声のマルチメディア通信を実現するためのリアルタイム通信プロトコルである。

動画、音声などの連続メディアは大きな帯域幅を必要とし、また一定の速度で生成される同じ大きさのパケットとして通信されることが一般的である。このようなことから、ST-2 は以下のような性質を持ったプロトコルとして設計された。

- 高速であること
- 通信品質の保証ができること (下位層の帯域予約機能が必要)

また、

- マルチキャスト通信 (ひとつのソースから複数に同時に転送ができること)
- 障害への対応

なども同時に考慮に入れている。

ST-2 は IP と同じ階層 (インタネット層) に属し、片方向ストリーム通信を実現する[1]。各ネットワークノードは ST エージェントと呼ばれる。ST-2 をサポートしていないルータを介して通信を行なうために、IP にカプセル化することもできる。

なお、ST-2 の最新の仕様は、[13] に記述されている。このドラフトには、ST2+ という改良バージョンの仕様が含まれている。本節では、RFC1190 に基づいた ST-2 について解説しているが、2.5.7 節に ST2+ で変更された点をまとめてる。

2.5.2 高速データ転送

ST-2 ではデータパケットのヘッダ情報を極小化しており、データ転送の高速化を図っている。ST-2 パケットの最初の 64 ビットであるヘッダでパケットの識別を行うが、ここには優先度、パケットサイズ、HID、ヘッダチェックサムが入っている。また、タイムスタンプを追加することができる。HID とはコネクションの識別情報であり、コネクション確立時に要求側と受け入れ側で調整して 16 ビットのユニークな番号を決定する。

2.5.3 通信品質の保証

連続メディアの時間的条件を実現するには、マルチメディアストリーム用に一定の帯域を割り当て、これを他のトラヒックから使われないようにする帯域予約が必要である。また、音声や動画などのリアルタイムデータは、一定の速度で生成される同じ大きさのパケットとして送られる。このようなリアルタイムデータを扱うネットワークでは、遅延時間、遅延のばらつきが小さいことが要求される。

ST-2 では、コネクション毎に一定の QOS を得られるフレームワーク (後述する FlowSpec) を提供している。これに基づいてコネクションが設定される。要求条件を充足できない場合には、それより低優先度のストリームを切断し、要求条件を満たそうとするが、それでも満たせない場合にはコネクションは拒否される。

ST-2 では帯域予約が可能であるが、ST-2 自身がリソースの予約や管理を実施するわけではない。プロトコルモジュールが複雑にならないように下位のモジュールで実現している。

2.5.4 FlowSpec

FlowSpec は Flow Specification の略で、エンド–エンドのサービス要求を伝えるために使われる。優先度、遅延時間の上限、誤り率、最小帯域幅、最小/希望パケットサイズ、最小/希望パケット速度などから構成される。平均遅延時間、遅延のばらつきなどの情報も FlowSpec に持つが、これを送り側で指定することはない。

FlowSpec はコネクションを確立する前に指定するが、通信の途中で変更することも可能である。

[1]プロトコルバージョン番号は、5。

2.5.5　マルチキャスト通信

ST-2では、1箇所の送信元から複数の宛先に持つルーティングツリーを設定できる。ルーティングツリーにはパスを記述しており、メッセージが複数のサブネットワークを経由する場合も正確に記述される。データ転送中でも宛先ノードの追加削除が可能である。ST-2は特定のルーティングモデルには依存しないが、帯域予約のためにはスタティックルーティングが最も適している。

2.5.6　障害検出、回復

ネットワークまたはノードの障害により、通信が途切れることがある。リアルタイムシステムに適用するためには、これに対応したプロトコルが望ましい。ST-2では、障害の検出と回復の機能が提供される。

STプロトコルによって通信しているノード (STエージェント) は、周期的に隣のエージェントにHELLOメッセージを送り、返事を受け取る。これにより、障害の検出ができる。ただし、これだけではネットワークの障害とノードの障害とを区別できない。下位プロトコルが障害の報告の機能を持つことが望ましい。

障害に備えて、各エージェントはストリームの状態情報を持っている。障害が発生したときに、新しくストリームを設定できる場合には、この情報を使って通信を再開する。

上位プロトコルは、NoRecoveryオプションを指定することにより、障害からの回復をしないでエラーの通達を受けることができる。

2.5.7　RFC1190 と ST2+

ST2+ と RFC1190 [6] の相違点を以下に示す。

- ホップ識別子 (HID) が除去された。HIDはプロトコルを複雑化し、インターオペラビリティを妨げる大きな要因であった。HIDは、グローバルにユニークであるストリーム識別子 (SID) に置き換わった。

- ストリームオプションが多数削除された。実装ではほとんど利用されず、利点が少なくプロトコルを複雑化しているオプションが多数あった。これらのオプションは削除された。削除されたオプションは、point-to-point, full-duplex, reverse change, source routeである。

- スブセット実装という概念を除去した。RFC1190では、簡単な実装および実験を可能にするためサブセット実装を規定していた。これにより、インターオペラビリティの問題も生じた。ST2+ [13] の仕様に記述されているプロトコルを実装したエージェントは、プロトコルを完全に実装しなければいけない。プロトコル中のいくつかの機

能は、best effort である。実装によっては、他のものと比較して、特定なプロトコル要求を満たすために、

- ターゲットがストリームに対する参加要求を発行できることを明白にした。RFC1190においてもターゲットからの要求はサポートしていると解釈できたが、多くの実装例では勘違いが生じ、この機能をサポートしてなかった。この機能を提供しなかったことにより、単一 ST ストリームの参加者の数を拡大する際に大きく制限されると判明した。この機能提供の明白化は、IBM Heidelberg で行われた研究を基にしている。

- ST およびそのサポートモジュール間の機能を分離した。ST が提供する機能と、その他のモジュールが提供する機能をより効果的に分離することを試みた。この結果、文書、およびいくつかの PDU フォーマットも再構成された。また、試験運用、およびインターオペラビリティを目的とするための FlowSpec は定義されているが、FlowSpec に依存しないかたちで ST は作成された。

- 仕様の再構成と書き直しを行った。ST2+ 仕様書 [13] は、さらに読みやすく、明白化することを目的として、再構成された。いつかの節が加えられ、概念の紹介の文章も改善された。

2.5.8 ST-2 の実装

ST-2 は、いくつかの研究団体によって実装されている。以下に、その実装例を示す。

- IBM European Networking Center (Heidelberg) [14]

- Swedish Institute of Computer Science (SICS) [15] SunOS 4.1.1 で動作する ST-2 のパッケージが配布されている。また、ATM 用の実装もある [16]。

- NTT [17] Real-Time Mach 上のユーザレベルサーバ内に ST-2 を実装している。

2.6 RSVP

RSVP [18] とは、主に Xerox PARC のメンバが中心となって開発している新しいタイプの資源予約プロトコルである。RSVP は、基本的には、単方向にネットワーク資源を予約するための通信プロトコルである。しかし、従来のリアルタイム通信プロトコルとは、異なるコネクションの確立方法を採用している。最新の RSVP の仕様は、[7] に記述されている。

以下に RSVP の目的を示す。

- 異種の受信ノードに対応する

- 変化するマルチキャストグループメンバシップに適合する
- ネットワーク資源を効率良く利用するために、異なるアプリケーションの資源要求を満足する
- 受信ノードにおいて、チャネルの切替えを可能にする
- 下位層が提供するユニキャスト、またはマルチキャストの経路の変化に適合する
- 参加者 (ノード) の数に応じて、線形的にプロトコルのオーバヘッドが増加すること、またはそれ以上のオーバーヘッドが生じることを防ぐ
- 異種の下位層技術に対応するためにモジュール性の高い設計にする

RSVP とは、ReSerVation Protocol の略であり、上記で述べた目的に現れるマルチキャストルーティングを始めとするネットワーク機能は提供しない。RSVP 自体は、資源を予約するために利用するプロトコルであり、特に、point-to-point ではない、通信相手が複数存在するマルチキャストを用いた場合に、効率良く資源を利用するために設計された。まとめると、アプリケーションプログラムは効率良く、および正しくネットワーク上に分散している資源を用いるために、RSVP を利用することができる。RSVP は、このようなアプリケーションの要求をネットワーク中に伝達する役目を果たすのみで、その他のネットワーク機能は、提供しない。例えば、各フロー (ストリーム、セッション、チャネルと同等な抽象概念) 毎に必要な同期機構、または信頼性を維持したマルチキャスト送信の必要性などを RSVP の flowspec によって伝達することができる。この場合、同期機構、または信頼性のあるマルチキャストを実現するのは、中間ノード (スイッチ、交換機) の役目になり、RSVP とは関係ない。その他、flowspec を用いて、さまざまな情報を伝達することが可能なので、まだ先の時間で利用する資源を確保すること、または資源の横取りが可能な予約方法などといった新しい予約方法が考えられる。RSVP は、このような新しい機能を提供することが可能であるという記述は見られるが、実際には、具体的には手法は提示されていない。

2.6.1　設計方針

前章で述べた目的を果たすために、RSVP では以下の 6 つの設計方針が考えられた。

- 受信側が資源予約要求を発行する方法
- 予約機構とパケットフィルタリング機構の分離
- 異なる予約手法 (reservation styles) の提供
- ネットワーク中における “soft-state” の管理

- プロトコルオーバヘッド制御
- モジュール性

以下に、上記の設計方針について具体的に述べる。

2.6.2 受信側が予約要求を発行方法

RSVP の設計において、従来の資源予約プロトコルでは、データソース (源) が資源予約要求を発行するということが注目された。これ対して、RSVP は、データを受信する側が資源の予約要求を発行するという逆の手法が提供された。詳しく述べると、データ受信側は、予約する資源のレベルを選択し、予約要求の発行し、データを受け取る間は予約状況を維持する責任をもつ。

この受信側が予約要求を発行する方法は、受信側が自分のもつ処理能力の限界を最も良く知っているということ、または自分が必要とする QoS を直接操作する必要性があることから生まれた。さらに、今後資源に対する課金制度が導入されると、料金を払う受信側が、必要とする資源を支払うことのできる金額に応じて、選択的に確保できることが望ましい。

2.6.3 予約機構とパケットフィルタリング機構の分離

スイッチ中の予約機構は、予約要求を発行した実体に対して、バッファ、ネットワーク帯域などの資源を割り当てる。しかし、このような予約機構においては、ある通信に相手に対する資源の量を割り当てることだけが重視され、どのパケットが実際に予約された資源を利用して良いかという、概念は注目されていなかった。

したがって、RSVP では予約機構と分離されたパケットフィルタ機構という、予約された資源を利用可能であるパケットのみを通し、その他のものは遮断するという機構を導入する。ここで、この機構では、現在予約されている資源の量を変更せず、その機能を変えることが可能であることが重要である。さらに、RSVP ではパケットフィルタの機構は動的に変更可能である。すなわち、資源を予約している際に、変更可能である。この二つの機構の分離により、従来ではなかった新しい資源予約手法が提供できる。

2.6.4 異なる予約手法

マルチキャストアプリケーションのサービス要求に応じて、各々の受信ノードから受ける資源予約要求をまとめて処理した方が、効率良くネットワーク上の資源を利用することが可能である。例えば、人間同士の会話を考えてみると、ある瞬間言葉を話す人間は通常一人である。それ以上の人間が同時に会話を始めると、会話全体が聞きとれなくなってしまう。したがって、音声コンファレンスシステム用の資源を予約する際には、何も参加している全ての人が同時に音声データを流す可能性を考えて、大量の資源を割り当てる必要は

ない。すなわち、同時に発生しうる小量の音声ストリーム用の資源をネットワーク上においてまとめて確保すれば良い。しかし、同じコンファレンスアプリケーションにおいても、ビデオ信号が送信される場合は状況が異なる。この場合、同時に見たいビデオストリーム分の資源を予約する必要がある。また、ネットワークの形状を考慮し、下流に存在する 2 つの受信ノードが、アプリケーションが終了するまで、同じビデオストリームのみを見る場合 (ビデオで講義を受ける場合など) は、その下流に対するネットワークリンクにおいては、ストリーム一つ分の資源を確保すれば十分である。しかし、常にこのような状況が想定できるわけではなく、一方の受信ノードがデータを送信するソースを切替える必要がある場合は、一つ確保するだけでは済まない。このようなさまざまな状況に応じて、ネットワーク中の中間ノードにおいて、RSVP は資源をまとめて予約する手法をいくつか提供する。現在のところ以下の 3 つの手法がある。

- フィルタなし (no-filter)、wildcard とも呼ばれる
- 固定フィルタ (fixed-filter)
- 動的フィルタ (dynamic-filter)

受信ノードがマルチキャストアプリケーション用に資源予約要求を発行する際に、データソースフィルタを用いるかどうかを指定することができる。フィルタを利用しない場合、そのマルチキャストグループに送られるパケットは全て予約された資源を利用することができる [2]。前述の音声コンファレンスシステムの場合は、このような予約手法を用いることができる。すなわち、現在、話している人が予約されている資源を利用するということである。

また、ソースフィルタ機能が必要な場合は、ソースのリストを指定することによってフィルタを実現する。この場合、リストに含まれる指定されたソースのみが予約された資源を利用することができる。フィルタされて予約された資源は、ビデオコンファレンスにおいて、参加者はある特定の人の画像を得るために利用できる。固定フィルタの場合は、資源を予約した期間中は、指定したソースのみからデータを受けとることができ、このリストは不変になる。動的フィルタの場合は、受信側は資源を利用可能なソースのリストを時間に対して動的に変更することが可能となる。

固定フィルタの手法は、マルチキャストアプリケーションへの参加者が全て同じ映像、または音声を受信する場合に適用できる。先に述べた、ビデオ講義システム (multicast video lecture) は良い例である。動的フィルタは、受信する映像のソースを動的に変更する場合に適用できる。RSVP では、資源予約機構とパケットフィルタの機構を分離することによってこのような手法が生まれて、マルチキャストアプリケーションの場合、一つ一つの資源要求を可能な限り融合し、ネットワーク中のスイッチの資源を効率良く利用することを目指している。

[2] ここでは、まとめて処理されるストリームが予約された資源以上を利用してしまうことを防ぐ機構が必要ではあるが、それについては触れていない。

2.6.5 “soft-state” の管理

長期に渡ってマルチキャストアプリケーションが動作すると、必ずグループに対する参加、および脱退要求を複数発生する。またネットワークの状況など、すなわち中間スイッチおよび中間リンクなどの変更によって動的にネットワーク経路に変更が生じる場合もある。このような動的な変化に対応するためには、中間スイッチ (ノード) において、ある程度、状態に関する情報を維持する必要があり、RSVP は、 このような情報を用いて、end user に資源の予約状況を管理する責任を与える。RSVP ではこの維持する必要のある状態のことを “soft-state” と呼んでいる。“soft-state” とは、各スイッチにおいて維持される状態で、損失された場合は、自動的に RSVP によって復活されるものを表す。状況が常に変化している前提としている環境においては、非常に重要である。

さらに、RSVP では、各中間スイッチにおいては状態に関する情報はパス状態と資源状態の二つに区別される。各データソースは、定期的にパスメッセージを送信し、パス状態を確立、または更新する。また、各受信ノードは、予約メッセージを定期的に送信し、予約状態を確立、または更新する。パスメッセージは、スイッチのルーティングテーブルを用いて転送される。したがって、ルーティング (経路制御および決定) は、RSVP では行なわれない。各パスメッセージには、データソースが与える flowspec およびフィルタ機能の情報を表す F フラグが含まれる。パスメッセージを受けとるとスイッチは、パス状態を更新する。パス状態には、上流のソースに対する入力リンク、および下流であるグループに属すソースに対する受信ノードへ出力リンクの情報が含まれる。また、F フラグがオンの場合、スイッチはソースとソースに向かって上流の方向の一つ前のノードの情報も維持する。この情報によって、スイッチは任意の予約手法に対応することができる。F フラグがオフの場合、ソースに関する情報は維持せず、パス状態に入力リンクを加えるだけである。したがって、このようなソースからのデータストリームに対しては、フィルタなしの予約手法しか用いることができない。

予約メッセージには flowspec、予約手法、およびフィルタを用いた場合、パケットフィルタの指定が含まれる。各予約メッセージを処理する際に、メッセージが送信されてきた出力リンクの情報が含まれる予約状態を更新する。この更新では、以下の情報が記録される。

- 予約された資源の量
- 予約された資源のソースフィルタ
- 予約手法
- 動的フィルタを用いている場合、予約したノード (予約したノードとは、この予約メッセージを送信したマルチキャストグループに属すメンバのことである)

上記から分かるように、フィルタなし、または固定フィルタの場合は、予約された資源はマルチキャストグループ全体に割り当てることが可能なので、余計な情報を維持する必要はない。

予約メッセージは、ソースに対して転送されるが、パスメッセージ中のパスを逆にすることによって行なわれる。実は、パス情報とは、予約メッセージをこの逆方向に送信するためのみに維持される。実際には、フィルタなしの予約メッセージは、マルチキャストグループに対する全ての入力リンクに向かって転送され、フィルタを用いるものは、フィルタ中にリストされたソースに向かった前のノードに対して転送される。

パスメッセージ、および予約メッセージの両方とも中間スイッチが用いるタイムアウト値を含む。タイムアウトしたメッセージに対する状態は削除される。この機構によって、受信側が明示的に削除メッセージを送信できなくなった場合、または下位の経路が変更された場合、残留が発生することを妨げる。フィルタなし、または固定フィルタの予約時に資源を解放するための唯一の方法でもある。このような場合、スイッチは予約された資源が複数の受信ノードによって共有されていることを判断することができない。したがって、予約された資源はタイムアウトが発生した段階で始めて解放される。このように、定期的にパス状態および予約状態を更新することによってネットワーク中の資源予約状況を正しく維持するのは、送信側および受信側の責任である。

メンバシップおよび経路情報が変更されると、RSVP の下位層にあるルーティングプロトコルは、新しい経路の情報を含むパスメッセージを転送する。したがって、スイッチ中のパス状態は更新され、その後の予約メッセージは、新しい経路に従って転送される。古い経路上で予約された資源、またアプリケーションへの参加を中止した送信側および受信側に対する経路上の資源は、上記のタイムアウト機構によって自動的に解放される。また、パスおよび予約メッセージは、定期的に送信されるので時たまに起きるメッセージの損失などには柔軟に対応することができる。この “soft-state” により、RSVP の適合性および robustness が向上される。

しかしながら、このように定期的に更新メッセージを転送することによってプロトコルのオーバヘッドが増加する。その問題を解決するために、プロトコルオーバヘッドの制御を試みる。

2.6.6　プロトコルオーバヘッド制御

RSVP のオーバーヘッドは、以下の 3 つの要因によって決定される。

- 送信される RSVP メッセージの数
- 送信される RSVP メッセージのサイズ
- パスおよび予約メッセージの送信頻度

上記の要因の効率を向上するため、RSVP ではパスおよび予約メッセージをネットワーク中を伝達するに渡って融合 (merge) する。パスメッセージを融合することによってたいていの場合、一つの更新周期においてあるリンクを通るパスメッセージが一つに限定され

ることになる。予約メッセージおいても同様である。パスおよび予約メッセージの最大サイズは、上流に存在するデータソースの数に正比例する。

3 つ目のオーバーヘッドは、すなわち送信頻度は、タイムアウト値を最適化することによって制御する。当然、タイムアウト値を大きくすると、更新メッセージの送信頻度を低くして構わない。ここで、RSVP の応答性と送信頻度によって生じるオーバーヘッドと間において、トレードオフが生じる。

現在の RSVP の実装では、あらかじめ判断し決定した静的タイマ値を用いている。今後、適合性の高い、タイムアウトアルゴリズムを検討し、タイムアウト値の最適化を測る必要がある。

2.6.7 モジュール性

実時間およびマルチキャストアプリケーションに対応するため、RSVP は以下の 3 つのコンポーネントからなる。

- アプリケーション、または RSVP を呼び出すたのセッション制御プロトコルが提供する flowspec
- RSVP のパス状態をパスメッセージを全ての受信ノードに転送するルーティングアルゴリズム
- 予約メッセージに含まれる flowspec によって新たな参加者の受理を決定するネットワークアドミション制御機構

このように RSVP は他のモジュールから切り離され、独立して実現できることが重視されている。これらのコンポーネントに関していくつか仮定してきたことはあるが、flowspec に関しては、何も仮定はされていない。RSVP では、flowspec を任意長のバイト列からなるデータで、アプリケーションとネットワークアドミション制御機構との間で交換ものとみなされる。

下位層のルーティングプロトコルに関しては、ユニキャスト、またはマルチキャストルーティング機能を提供すること、またマルチキャストグループに対して送信されたデータ通常のネットワーク状況においては全てのメンバに送られることを仮定している。グループに対してデータを送信するノードは、必ずしもグループに属している必要はなく、送信側から受信側に対する経路と受信側から送信側に対する経路が同じであることをも仮定していない。

2.7 The Tenet Real-Time Protocol Suite

2.7.1 Tenet Real-Time Protocol Suite の概要

Tenet Group は、カルフォルニア大学バークレィ校で 1989 年、9 月に高速コンピュータ通信の研究を行なうために設立された。1991 年夏に Tenet Real-Time Protocol Suite [5] は、詳細に仕様化され、FDDI の LAN 上に実装された後、Project Sequoia 2000 という WAN 上で、数カ月様々な実験が行なわれた。Tenet Suite は現在、 新しい testbed として、XUNET2 (FDDI リングを接続するための米国の東海岸と西海岸を結ぶ ATM バックボーン) や XUNET3 (1.5 Km シリアル HIPPI テストベッド) に移植中である。

Tenet Real-Time Protocol Suite には、ネットワーク層として、Real-Time Internetwork Protocol (RTIP) と 2 種類のトランスポート層として Real-Time Message Transport Protocol (RMTP) および、Continuous Media Transport Protocol (CMTP) がある。制御用のプロトコルとしては、コネクション管理をする Real-Time Channel Administration Protocol (RCAP) および、メッセージ送信の失敗、再送の制御をする Real-Time Control Message Protocol (RTCMP) がある。

これらのプロトコルレイヤを図 2.1に示す。

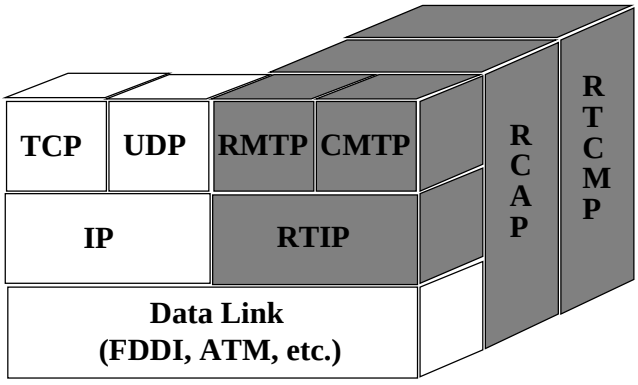

図 2.1: Tenet Real-Time Protocol Suite

RCAP Real-Time Channel Administration Protocol

RCAP の主な機能は、リアルタイムチャンネルの生成と削除を行なうことである。RCAP では制御エンティティ間でやりとりされるメッセージが定義されている。それぞれのメッセージには明示的な方向を持っている。方向は、“upstream”という sender 方向か、“downstream”という receiver 方向のどちらかである。RCAP のそれぞれのメッセージを以下に説明する。

establish_request	Downstream	新しいチャンネルの設立の要求をする。そのため、そのチャンネルの経路上のそれぞれの hop で、アドミッションテストを実行させ、そのチャンネルのために資源を一時的に割り当てる。
establish_accept	Upstream	新しいチャンネルの受理と資源の割当を示す。
establish_denied	Upstream	チャンネルの設立要求がネットワーク内もしくは接続相手側で拒否されたことを示す。チャンネルに割り当てられていた資源は解放される。
status_request	Downstream	チャンネルのそれぞれの node の状態情報の要求をする。
status_report	Upstream	status_request メッセージによって収集されたデータを返す。
close_request_forward	Downstream	チャンネルを閉じさせる送信者側アプリケーションからのメッセージ
close_resuest_recerse	Upstream	チャンネルを閉じさせる受信者側アプリケーションからのメッセージ

RCAP インタフェイスの QOS parameters および、トラフィックパラメータを以下に示す。

D_max	end-to-end メッセージ遅延の上限
Z_min	メッセージの配送時間の下限
J_max	メッセージの遅延ゆれの上限
W_min	バッファオーバフローしても損失しないメッセージの下限
X_min	メッセージ間隔 (inter-message time) の最小値
X_ave	メッセージ間隔 (inter-message time) の平均値
I	メッセージ間隔の平均値
S_max	メッセージサイズの最大値

RTIP Real-Time Internetwork Protocol

RTIP の主な機能は、対応するチャンネルの実時間要求に合うようにパケットの配送をすることである。

IP データグラムサービスと対照的に RTIP はコネクション指向である。RTIP のデータ転送には信頼性はない。パケットは、バッファオーバフローのために破壊されたり消失する可能性があるからである。RTIP は、それぞれのチャンネルの QOS パラメータに基づいて、レートコントロール、ジッター制御およびスケジューリングを行なう。RTIP コネクション上のすべてのパケットは同じパスを通るので、パケットは

順序が保たれてで配送される。RTIP は信頼性がないが、保証された品質で、一方向、保たれた順序のパケット配送サービスを提供する。

コネクションの QOS を保証するために、それぞれの RTIP エンティティには、レート制御モジュールとスケジューリングモジュールがある。レート制御モジュールはそれぞれのコネクションのトラフィックをモニタし、そのコネクションのトラフィック仕様に従って、シェイプする。加えて、レート制御モジュールは、コネクションの遅延のゆれの要求を満足させることを保証する。スケジューリングモジュールはパケット伝送の遅延のデッドラインを保証させるために、パケットに優先度をつける。

RMTP Real-Time Message Transport Protocol

RMTP は、TCP とは異なり信頼性は保証しない。また、フロー、輻輳制御、レート制御、パケット順序の保証はその下位層である RTIP を利用している。

RMTP の主な機能は、パケットの分解と組み立てである。RTMP モジュールでは、RTIP のパケット最大長よりも大きなサイズのパケットの送受信を行なうことが可能である。それぞれのパケットは、RMTP ヘッダーに含まれるメッセージシーケンス番号および、RTIP ヘッダーに含まれるパケットシーケンス番号が含まれている。RMTP の受信モジュールは受信したメッセージを組み立てるために、これらのシーケンス番号を利用する。その時にメッセージヘッダのチェックサムを計算し、組み立てたメッセージが正しいか確かめる。

第 3 章

Real-Time WG 内のプロジェクトの紹介

Real-Time WG には設立前から、各個人が係わっていたプロジェクト、および今後新たに発展させていくプロジェクトがいくつかある。この章では、これのプロジェクトについて簡単に解説する。

3.1 プロトコル制御に注目したリアルタイム通信プロトコル

RtP [19] は、従来の資源予約プロトコルでは考慮されていなかったプロトコル制御メッセージ専用の資源を予約する機構を導入している。ノードまたはホスト起動時に、プロトコル制御用に微小な資源を確保しておくことによって、データ用のチャネルの資源を効率良くおよび予測可能に制御する機構を提供する。本章では、RtP Version 1 の RT-Mach 上での設計および実装について解説する。

3.1.1 従来の予約プロトコルの問題点

従来の資源予約プロトコルは、全てコネクション指向ネットワーク・プロトコルであり、ストリームまたはセッションと呼ばれるコネクションを通信前に確立することによって資源を予約する機構を実現している。

一般的に、資源を予約するプロトコルでは、次の手順に従ってセッション、またはストリームなどは確立される。

1. ソース・ノードで出力用ネットワーク資源 (ネットワーク・バンド幅、ネットワーク・バッファなど) および CPU 時間を予約する。
2. 中間ノードは、プロトコル処理用の CPU 時間を予約し、ネットワーク資源を確保する。
3. 目的ノードで、CPU 時間に加えて入力用ネットワーク・バッファを確保する。
4. 確認応答 (ACK) をソース・ノードに返す、必要に応じて、資源の予約状況は変更される。

5. ソース・ノードが ACK を受信する。

6. セッションまたはストリームは確立され、リアルタイム通信が可能になる。

RSVP では、この手順がソース・ノードからではなく、受信ノードから行われる。しかしこれらのプロトコルでは、障害発生時、および過負荷状態などのシステム状態を考慮し、通信セッションを制御するための処理および動作は具体的に規定されていない。通常アプリケーションにはネットワーク通信が不可能、または輻輳のためデータ転送が遅くなったことなどを判断するためのインタフェースは提供されていない。分散システムは本質的に予測不可能で常時変化するので、ネットワーク・アーキテクチャは、障害 (ネットワークの場合、物理的な回線の切断、仲介ノードのクラッシュなど) に対する処理、また規定時間 (タイムアウト値) 内に通信を終了できない場合の処理などを明確に定義しないと、アプリケーション、また場合によってはシステム全体が一時的に停止してしまい。処理を先に進めることができなくなってしまう。障害回復機構を提供する必要まではないかもしれないが、実時間性を保つことができなくなってしまった時のシステムのとるべき動作、また上位レベルまたはアプリケーションに提供するインタフェースなどを規定する必要がある。

資源を予約する機構という概念は有効的な手法ではある。しかし、これだけでは実用的なリアルタイム・ネットワーク・アーキテクチャを築くことはできない。すなわち、ネットワークバンド幅を確保するための機構が用意されている最下位層の通信デバイスをただ導入するだけでは不十分であり、他にも考慮しなければいけないことがある。前述の予約プロトコルでは、資源を確保するため、および資源の予約状況を変更するための制御情報がネットワークにおいてタイムアウトした場合の処理を考慮していない。CBSRP では、ネットワーク資源の予約したパラメータを動的に変更するという制御機構は実現されている。ST-II の場合、SCMP という制御メッセージプロトコルを導入してネットワークの多様な変化に対応している。しかし、このような制御機構を導入するだけでは十分ではない。すなわち、制御メッセージのリアルタイム性についても考慮しなければ、任意のシステム状態におけるリアルタイム処理は規定することができない。例えば、 QOS パラメータの変更用制御メッセージのリアルタイム性などは考慮されず、障害によってそれが正しく転送されなかった場合、データ通信に影響を及ぼす可能性がある。限られた資源を予約する場合、複数の通信セッションにおいて、それを有効に利用するためには、同時に使用される資源の量を異なるセッション間で協調して利用する必要がある。このような状況に応じてプロトコル制御メッセージを利用して動的に QOS 制御をする際には、やはり制御メッセージの実時間性を維持する必要がある。

3.1.2　VSL モデル

RtP は、利用できる下位層の通信媒体を 2 つの仮想回線に抽象化する VSL モデルを基本としている。 VSL は、Virtually Separated Link の略で制御情報専用チャネルおよびデータ転送専用チャネルからなる。

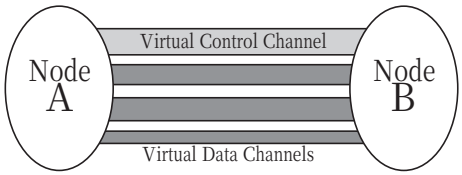

図 3.1: VSL モデルの概略図

VSL モデルの基本構造を図 3.1 に示す。制御情報専用チャネルを VCC (Virtual Control Channel)、データ転送用チャネルを VDC (Virtual Data Channel) と呼ぶ。既存のシステムでは言うまでもなく、将来のシステムにおいても全ノードを互いに複数の回線によって接続することは困難なので、仮想チャネルを用いてリアルタイム通信を保証する。従来の通信プロトコルでも、制御情報およびデータの転送は区別されていて、この 2 つの概念を区別することは新しくない。

複数の回線がある場合、このインタフェースを提供することによってネットワーク・アーキテクチャはシステムのハードウェア構成をそのまま抽象化することができる (例えば、2 つのノードが 2 つの point-to-point リンクによって接続されている場合、一方を VCC、他方を VDC に直接マッピングできる)。また、物理回線が 1 つしかない場合でも、仮想的に 2 つの区別された回線があるとみなし、オペレーティング・システムに制御情報およびデータ用処理を区別してスケジューリングするためのインタフェースと、ネットワーク媒体に対して、ネットワークバンド幅を区別するためのインタフェースを提供する。

このモデルは、日常的に利用されている電話網の概念を採り入れたものである。全ノードが制御情報用チャネルで相互に接続されることによって、任意の時点でリアルタイム通信が実現可能であるかをリアルタイムに知る機構を提供することができる。また、out-of-band 通信の概念とも類似している。

ST-II などの資源を予約するプロトコルの場合は、VSL モデルを導入することによって、リアルタイム通信プロトコルに拡張することができる。

3.1.3 RtP の概要

RtP は、7 階層モデルのネットワーク層ように開発されたコネクション指向ネットワークプロトコルである。先に述べた VSL モデルを導入することによって、プロトコル独自の制御メッセージのやりとりにも実時間性維持しながら実現できる。RtP には、VSL モデルの仮想制御チャネル (VCC) と仮想データ・チャネル (VDC) の 2 つの予約手順が定義されている。

RtP では、チャネルはノード毎、またはセッション毎 (すなわち、通信しているアプリケーション毎) に確立することができる。前者を node oriented、後者を session oriented と呼ぶ。VCC は node oriented で、通常はネットワーク中の 2 つのノードは 1 つの VCC で

結ばれている、VDC は session oriented でアプリケーションの通信セッション毎に確立される。したがって、2 つのノード間において複数の VDC が存在する可能性はある。

RtP の通信は次の 3 つに分類できる、

- 通常のデータグラム通信
- VCC を介した通信
- VDC を介した通信

通常のデータグラム通信は、時間に依存しない従来の通信である (すなわち、IP などのプロトコルで実現される通信)。これ以外には、すでに確立されたチャネルを用いて通信する方法がある。VDC の場合は、通常のデータが転送され、VCC の場合は制御情報が転送される。

RtP が提供する制御メッセージを表 3.1 に示す、

Message Type	Synopsis
EST_VDC	VDC establish request
EST_VDC_ACK	VDC establish success ack
VDC_FAIL	VDC establish fail
EST_SESSION	session establish request
ACK_SESSION	Normal datagram (same as ST-II)
MEASURE_RTT	timing parameter notification
NODE_STAT	resource availability notification
BQOS_CHANGE	bounded QOS change request
BQOS_CHANGE_ACK	BQOS success ack
BQOS_FAIL	BQOS fail
QOS_CHANGE	QOS change request (unbounded)

表 3.1: RtP message types.

3.1.4　VDC の確立手順

他の予約プロトコルでは、セッションまたはストリームに対応する VDC を確立する方法は 2 つある。まず、通常のデータグラム通信を用いた方法がある。これは、ST-II でストリームを確立する手順と同等である。この方法を用いて確立される VDC の場合、確立時のリアルタイム性は重視されない。

通常 VDC は、VCC を介して確立されるので、ある規定時間内までにシステム資源の予約を完了することができる。通常のリアルタイム通信を実現するためには、この形態の VDC を用いてリアルタイム性を保証する。したがって、資源使用率が過負荷状態に近くなり VDC の確立に失敗した場合や、物理的障害が発生した場合に、アプリケーションは規定時間内に適切な処理を行なうことができる。しかし、この場合 VCC がすでに確立されていなければいけない。

以下に VCC を介して VDC を確立する方法を示す。

1. EST_VDC 要求をあらかじめ確立された VCC を介して転送する (図 3.2 参照)、規定時間内にこの処理は行なわれ、要求を受けとると、各ホストではこのセッション用の資源を予約する。

2. 1 が終了すると、2 つのアプリケーション・エンティティ用の VDC が確立され、データ転送を開始することができる (図 3.3 参照)。

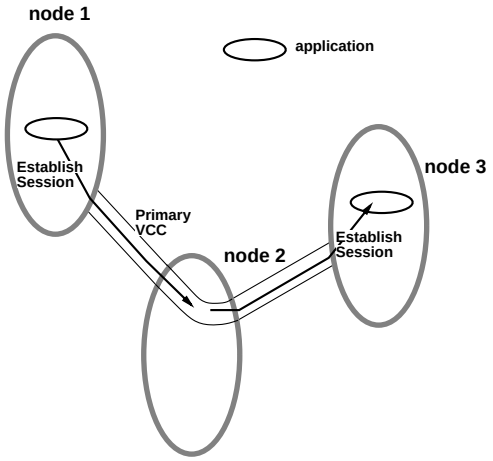

図 3.2: アプリケーション間における VDC の確立

VDC はセッション毎に確立され、ノード毎には作成されない。

3.1.5 VCC の確立手順

RtP では、プロトコル制御メッセージの伝達とそれに対する応答がある規定時間内で処理されることを保証するために VCC をホスト間で確立する必要がある。

VCC の確立手順は、Virtual Control Setup Phase (VCSP) によって行われる。VCSP は、RtP のサブセットであり、各ネットワークセグメントで、VCC を確立するために利用する。VCSP を導入することによって、各ネットワークセグメント内におけるプロトコル

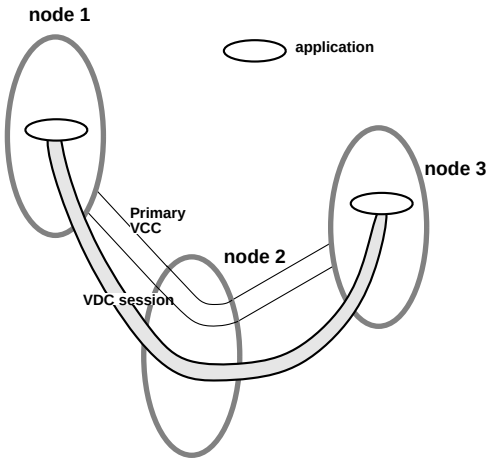

図 3.3: VDC の確立終了

制御用の資源をモジュール化することが可能となった。VCSP では、ゲートウェイと他のノードを区別し、VCC の管理オーバーヘッドを少なくするため、ゲートウェイでは、利用できる資源の上限を設ける。VCSP は、前述した VDC とは全く関係なく VCC を確立するためにだけ利用される。

図 3.4 に VCSP の手順を示す。

システム初期化時に、設定ファイルを参照することによって、VCSP が動作する。このファイルには、同一セグメント内にある通信するため資源を予約する必要があるホストの一覧表を記述する。他のセグメントに接続しているホストと通信する場合、該当するゲートウェイに対して資源を予約する必要がある。

図 3.4 は、VCSP の一例を示している。要求ホストは、自身のバッファ、CPU をローカルで予約し、ネットワークリンクのバンド幅を予約した後、VCC_SYN を通信相手に送ることにより、VCC を確立する。通信相手は、要求ホストと同様に必要な資源を予約し、VCC_SYN_ACK を返す。この後、VCC_INIT という制御メッセージを再度送信することによって、VCSP の手順は終了する。

VCSP は、このように 3 way handshake 機構を導入している。これにより、VCC 用の状態情報が無駄にならないことを保証している。VCC_SYN_ACK が失われた場合、実際には利用されない資源が予約されたというかたちで保持される。VCSP は、通常ホスト立ち上げ時に行われ、障害が起こるまで有効である。最後に、VCSP の手順は、すべて通常のパケットベースの通信で行われるので、規定時間内に行われない。

この他にも、単純は、request/reply モデルを導入することも考えられる。その場合、未使用の資源を定期的に発見する処理を行う必要がある。

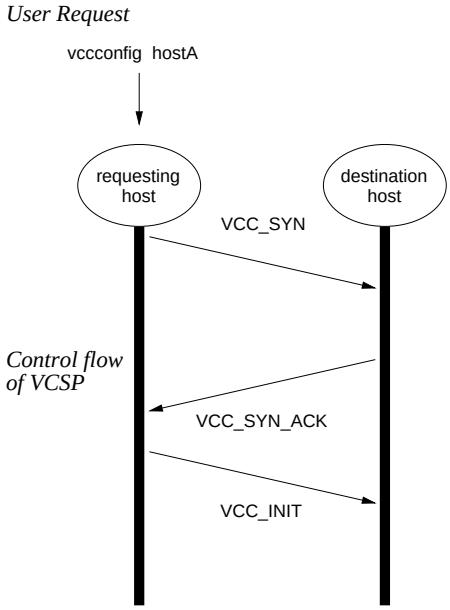

図 3.4: Virtual Control Setup Protocol の概略。

3.1.6 実装

RtP Version 1 は、 PC-AT 互換機で駆動している RT-Mach マイクロカーネル上で動作する NPS 内に実装された。

現段階では、VCC の管理を行なう CCH (Control Channel Handler) および VDC の管理を行なう DCH (Data Channel Handler) は、 RT-Mach のスレッドとして NPS 内に実現されている。通常は 1 つの DCH が複数のアプリケーションのリアルタイム通信のデータ処理を行なうが、セッション毎に専用の DCH を作成することも可能である。

また、RtP を利用するために NPS の内部には、RtP スタックが新たに追加されている。現時点では、RTS のアプリケーションは、RtP を直接呼び出すようになっていて、トランスポート層プロトコルは用意されていない。従来の NPS のアプリケーションに対するインタフェースは、そのまま利用できるので、アプリケーションプログラムは大幅に変更する必要はない。

3.1.7 評価

ここでは、RtP Version 1 の簡単な評価を示す。今回のデータは、Intel i486DX2-66 を装備した PC-AT 互換機に Ethernet として 3com Etherlink II/16 また FDDI に Network Peripherals の NP-EISA を搭載して、測定した結果である。

表 3.2 には、Ethernet 上のデータ転送を UDP/IP スタックと比較したデータを示す。

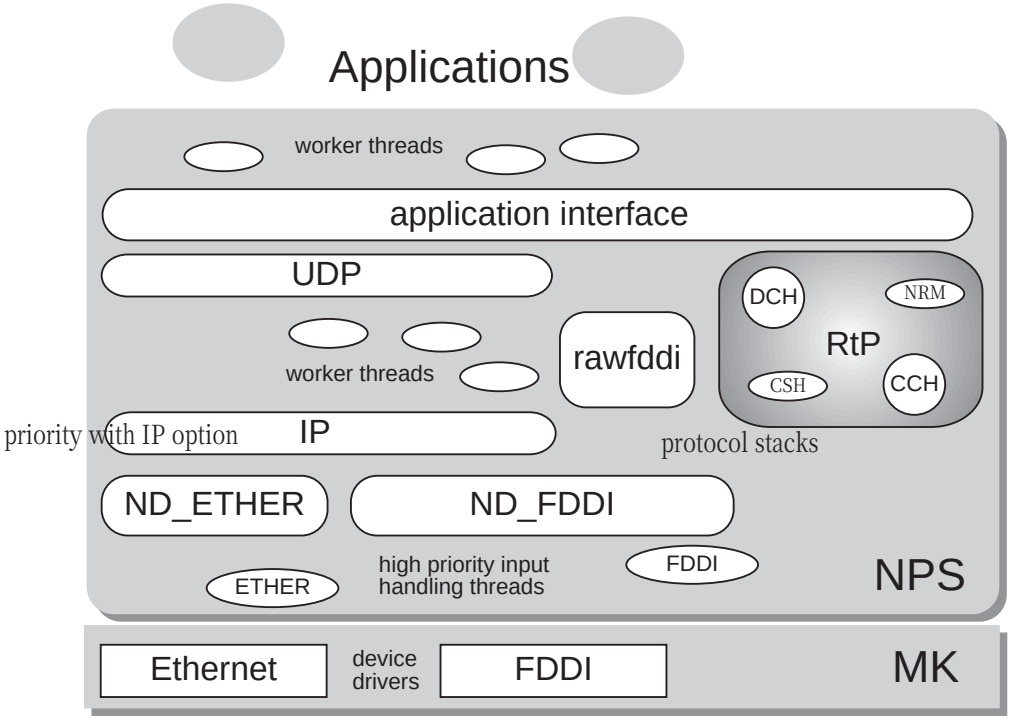

図 3.5: RtP の構成

Data bytes	RtP/Eth	UDPIP/Eth
4	6.36	5.22
512	9.21	8.25
1024	11.96	10.28

表 3.2: RtP 評価 Ethernet の往復転送時間 (ms)

表 3.3 には、FDDI 上のデータ転送を UDP/IP スタック、およびポートの機能を実現している何も処理しない rawfddi プロトコルを比較したデータを示す。

Data bytes	RtP/FDDI	UDPIP/FDDI	rawFDDI
4	6.69	5.63	3.95
512	7.91	7.04	5.16
1024	9.05	8.21	6.35
2048	11.54	10.79	8.71

表 3.3: RtP 評価 FDDI の往復転送時間 (ms)

これらの二つの表は全て、RTS のエンドアプリケーション同士において測定した数値になっている。通常のシステムの値に比べて全体の数値がかなり大きなものになっているが、従来の NPS にも FDDI をサポートするため変更が成されているため、バッファ管理の効率が落ちているのが原因である。RtP の評価は他のプロトコルスタックと比較して相対的に検討したい。

RtP のモジュールのなかではインタフェースを選択する部分などルーティング機能 (現時点ではスタティックなルーティングテーブルを用いている) などが付加されているため少々のオーバーヘッドが生じている。しかしながら、VDC を介してデータ転送する部分に関しては、プロトコルのオーバーヘッドが従来に比べて僅かである。

参考のため、表 3.4 には、Ethernet と FDDI のセグメントに接続しているゲートウェイノードを介した場合の往復転送時間を示す。

Data bytes	MachTS	Rate Mon.
4	10.02	9.06
512	13.72	12.53
1024	17.20	16.23

表 3.4: RtP 評価 FDDI と Ethernet の 2 つのセグメントを介した往復転送時間 (ms)

表 3.5 には、VCSP の性能評価を示す。

表 3.5 には、VCC を介して VDC を確立するための性能を表す。この表は、単一セグメント内の VDC の性能を示す。

表 3.7 には、VCC を一つのゲートウェイを介した場合の性能を示す。

VCC を確立するために要する時間 (表 3.5) と VDC を確立するために要する時間 (表 3.6) を比較すると、やはり使用するスケジューリングアルゴリズムに大きく作用される。

	Ether	FDDI
Mach Time Sharing	9.77	10.66
Rate Monotonic	10.92	10.69

表 3.5: RtP 評価 VCSP の場合 (ms)

	Ethernet	FDDI
Mach Time Sharing	9.61	10.36
Rate Monotonic	7.65	8.70

表 3.6: RtP 評価 VDC の確立時間 (単一セグメント) (ms)。

RtP Version 1 では、 VCSP の処理は、CSH によって実装されており、これは CCH よりも低いプライオリティで実行している非同期スレッドである。CCH の方は周期が短い高プライオリティスレッドである。したがって、明らかにスケジューリングアルゴリズムの影響が出ており、VDC の確立は、CCH によって処理されるので、レートモノトニックアルゴリズムを用いた方が安定して、僅かに高速に処理される。

3.1.8　検討事項

VCC と VDC の処理用に資源をどの程度の割合で確保するべきであるかという問題がある。これは利用されるアプリケーションにも依存し、確定することは困難である。現在のところでは具体的な手法は提案されていない。CCH をメタレベルの存在として取り扱い、このようなパラメータのとるべき値を常時検知する実体として実現する解決方法などが挙げられる。また、VDC 間の優先度、および VCC を介してながれる制御メッセージ用の優先度機構の導入の検討も必要である。

現在のところ規定時間を求めるためには、資源を予約する手続きおよびネットワークの通信に要する時間の予測値を規定時間として前提としている。特に、ネットワークの通信は、VCC の往復時間の測定メッセージを定期的に送出することによって予測値を求めてい

MachTS	Rate Mon.
13.89	12.27

表 3.7: RtP 評価 VDC の確立時間 (2 セグメント FDDI/Ethernet) (ms)。

るが、この方法が最適であるかを検討し、他の手法を考慮する必要があるかもしれない。

また、分散マルチメディア環境において、重要であるマルチキャスト通信の必要な処理を考慮しなければいけない。VCC 網を利用することによって、マルチキャスト用の VDC のトリーを確立する機能などが考えられる。

最後に、現在はリアルタイム・ネットワーク・アーキテクチャの第 1 ステップとして、RtP を開発しているが、今後 RtP で得た実験結果を基にし、プロトコル体系全体を再検討する方向で研究は進めてられている。

3.2 RSVP on IP over ATM

3.2.1 IP over ATM のモデル問題

現在、IP over ATM のモデルには Classical IP over ATM[20] と Conventional IP over ATM[21] の 2 つがある。これらは、それぞれ ATM の機能を発揮するために異なった仕様とモデルを提供している。

- Classical IP over ATM Classical IP over ATM のモデルは ATM のセル通信を活かすために Logical IP Subnet(LIS) という仮の大きなサブネットを作り、ルータを介さずに LIS 内のホスト同士が通信を行ない、LIS 間の接続のみルータを介すものとなっている。この LIS を用いたモデルを Large Cloud(LC) モデルとよばれる。しかし、このモデルは今までのインターネットのモデルであった CATE-NET モデルと異なっているため既存のプロトコルを変更しなければならない問題点がある。

- Conventional IP over ATM

  もう一方の Conventional IP over ATM は CATE-NET モデルを継承し、既存のインターネットと同様にサブネットまたはネットワークをルータによって接続するものである。しかし、従来の ATM ルータではパケットをフォワードするときに、入ってきたセルを一旦パケットに構成し、フォワード先を決定してから、またパケットをセルに分解して送出する二度手間になり ATM のセルによる高速通信を発揮することができない。そこでこのモデルでは従来のルータとは異なる Cell Switch Router(CSR) と呼ばれるルータを提供しする (図 3.6 を参照)。CSR とは何らかの方法であらかじめパケットの経路を決定したすることで、送られてくるセルをパケットまで構成せず、セルのまま通信を行なう機能をもったルータである。このモデルでは異なった (サブ) ネットに跨ったホスト間の通信は必ずルータを通るためルータにおいてパケットのフィルタリングなどが実現できるものと期待される。

- RSVP on IP over ATM

  RSVP は Hop-by-Hop に資源予約が行なわれるものである。よって ATM 上のインターネットに RSVP を実装するに当たっては Conventional IP のモデルを採用し、研

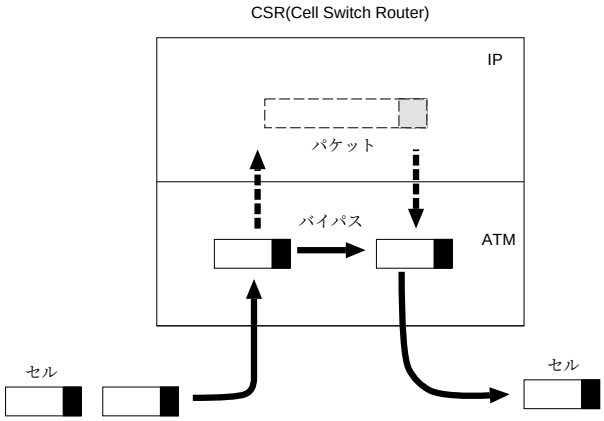

図 3.6: CSR の概略図

究を始める。ATM 上での資源予約の実現方法は RSVP からの予約情報に沿った VC を張ることによって実現されると考える。現在 RSVP の仕様ではトランスポート番号毎に、また、マルチキャスト通信いおいてはセッション毎に資源予約を行なうことが可能であるため、トランスポート番号毎にまたはセッション毎に、Next-Hop 間で VC を張ることで帯域が保証される (図 3.7参照)。

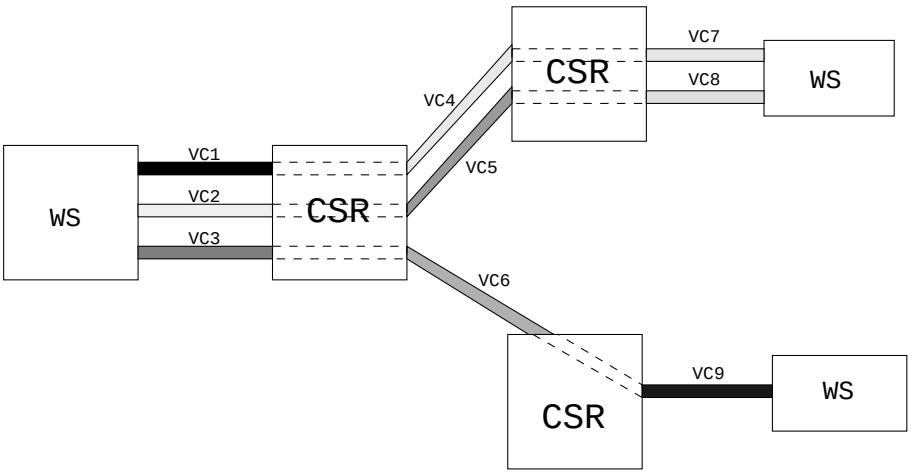

図 3.7: 接続モデルの概略図

RSVP は資源予約のフロー仕様の情報交換を行なうのみであり、実際に VC を張ることを行なわない。そこで、CSR 間で RSVP から得るフロー仕様を基に VC を張るための CSR 間の VC 接続プロトコルが必要となり、CSR 間、WS-CSR 間の VC を管理するための VC Allocate daemon を用意する (図 3.8 参照)。

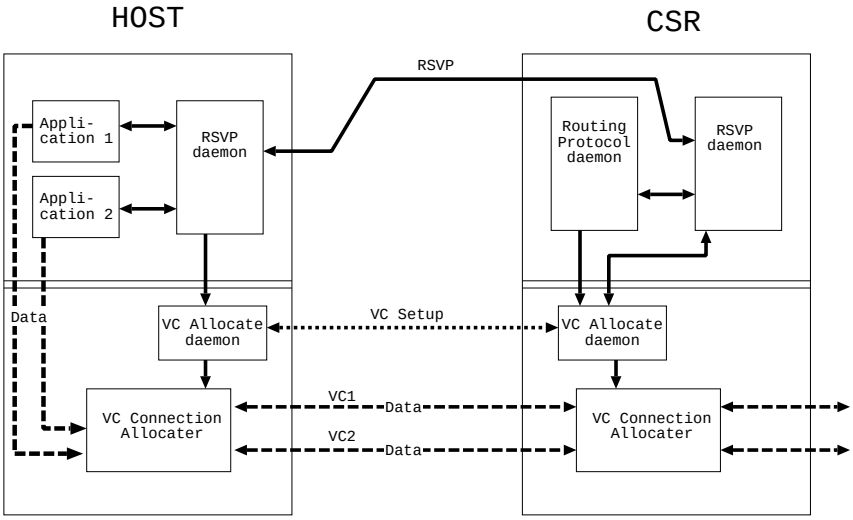

図 3.8: ホストとルータ間の構成図

3.2.2 実装と目標

現在、FORE,ENI,Interphaze 等から提供されるボードに附属する ATM ドライバの実装は RFC1577 による Classical IP over ATM の実装であり、また、ホスト間の接続はあらかじめ設定した 1 本の VC で張ることしかできない。このドライバでは上で述べた RSVP を使用した ATM の VC による帯域保証を行なうことが不可能である。そこでホストから送信先のホスト間においてアプリケーション毎（トランスポート番号毎、またはセッション毎）に VC が張れるような ATM ドライバが必要であり、今後、BSD/OS および FreeBSD での実装を試みる計画である。

また、CSR 間での VC を張るための接続プロトコルの開発および実装を行ない、RSVP からの要求によって WS-CSR-WS 間の VC による帯域確保を可能とし ATM 上でデータリンク層での資源予約通信の実現を計り、有効性を示す。

実装計画

開発環境を構成する機器としては当初 ATM スイッチには ATOMIS-5、マシンには PCI バスを搭載した AT 互換機、ATM ボードには Interphaze 社のボードを予定している。

- RSVP Version 1 Functional Specification 05 の実装
- VC 管理のための VC Allocate daemon の設計と実装

- IP アドレスとトランスポート番号を合わせ VC と対にし VC を張るための VC Allocator の実装

3.3　帯域資源予約機構を利用しない新たなリアルタイムプロトコルに関する研究

3.3.1　現在の問題点

現在の規模までに至ったインターネットの歴史を振り返る上で、各ネットワークのポリシ決定が、そのネットワーク管理者自身に委ねられていた事は、インターネットの発展に大きく影響したと言える。絶対的な命令権を持つ者によって指導されながら発展したのではなく、各ネットワークごとに適した管理方法も採用できたのである。同時に、インターネットプロトコルという統一的な通信規約を利用することによって、広域相互通信が可能となった。このように、各ネットワークごとにある程度の我儘 (わがまま) が通る自由な通信を確保出来た。

その延長上として、まずローカルエリア内における太い回線を利用したリアルタイム通信が、端末性能の向上などに伴い汎用されるようになってきた。何故ローカルエリア内なのかといえば、一般的に対外ネットワーク回線と内部ネットワーク回線の利用可能な帯域幅を比較した場合、対外ネットワークの方が回線幅が狭く、要求される品質のリアルタイム通信を行なうための充分な通信帯域幅を確保出来ないためである。それに対して、ローカルネットワークの場合、Ethernet などを利用していれば、10Mbps 程度は利用可能なので、対処も不可能ではなかった。

しかし、テレビ会議のようなことが汎用のネットワークで可能となることで、その利用要求は広がってきた。社内などローカルなネットワークだけでなく、物理的にもネットワーク的も離れた場所とリアルタイム通信を行ないたいというものである。それを現実とするためには、対外ネットワークから、通信相手が接続されているネットワークまでの、通信経路や通信帯域幅とその情報を処理するための CPU 資源などの確保が必要となってきた。これはまた、膨大な情報量を必要とする映像・音声通信などの利用増大に伴なったローカルネットワークの混雑にも対応でき、内外を問わない必要性が生じた。

そこで「帯域資源予約 (Session Reservation)」という通信技術が提唱されてきた。原理的にこの通信技術を用いれば、利用者は通信帯域を確保することが可能となり、広域なリアルタイム通信が実現される。実現されれば、より多数の利用者が広域に渡って多用途目的に利用することが安易に予想できる。

しかし、それに反してインターネットの物理的に利用可能な帯域幅は限られたものである。利用者の要求に応じた十分な帯域確保を行なうことは難しい。そのようなインターネットの状態のもとで、この帯域資源予約技術をインターネット全体規模で用いることが汎用的に行なわれる事態が発生すると、通常の通信にまで悪影響を及ぼすばかりか、管理用の

通信、情報配送などの通信にまで支障を来すことが予想される。

3.3.2 目標

以上のような事実、及び予期される事項を鑑みると、特に確立的な広域に渡るネットワーク帯域資源予約については、慎重に扱わなければならないことが分かる。そのため、まず技術・仕組みを正しく理解するためのサーベイを行ない、その利点・問題点を列挙する。その結果から、適切な利用方法を見出し提案出来ることを目標とする。

スケーラビリティや広域ネットワークにも対応した帯域資源予約、或は末端ユーザがそれと同じような効果を得られることが可能となるポリシ、技術、アプリケーションなどを、可能ならば実装という形で提唱出来るようにしたい。

3.3.3 研究進行予定

本目標達成に向けた実証的検証を行なうに際しては、ネットワーク媒体を特定することなく、またオペレーティングシステムに深く依存することなく、明示的な広域に渡る資源予約を考察する。その結果を踏まえて、末端的なアプリケーション・インタフェースの提案、設計・実装なども行なう。

第 4 章

最後に

前章の解説にも含まれていたが、インターネット社会で今後最も注目されることが、資源予約を一般ユーザに許すことである。資源予約という機構を導入することによって、今までのインターネットで一般に築き上げられた思想、および概念などを見つめ直さなければいけなくなってしまう。これが、果たして、良いことであるか、悪いことであるか、また一般ユーザおよびシステム管理者などの全てのインターネットの利用者の望んでいることであるかは、まだ答えがみつかっていない難しい問題である。

Real-Time ワーキンググループは、このような重要な「ながれ」を把握し、あらゆる観点から、インターネット社会に適したリアルタイム通信の実現を目指している。そのうち、資源予約の必要性を検討することは興味深い点であり、グループの方針としては、比較検討のため資源予約機構をベースにしたプロトコル体系の構築、および資源予約機構を利用しない体系の両極端からこの問題を叩くように定まった。要するに自由で open minded な姿勢で目標に向かって研究を進めていくことである。

その他には、やはり具体的にさまざまな通信媒体を利用したリアルタイム通信の実現、またその基盤の上での QOS 制御に関する実験を考えている。媒体としてはいろいろ考えられるが、現在では、まだ高速通信媒体としてしか見なされていない FDDI や ATM をリアルタイム通信に適した形で利用する形態を検討中である。近い将来としては、さまざまな現存するプロトコルの実装を試み、それぞれの比較検討などを考えている。