# 第 4 部

# リアルタイム通信

# 第 1 章

# はじめに

本章では、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 RT 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。

今まで，主に資源予約プロトコルに注目して，既存のプロトコルの問題点を補う新たなプロトコルの設計を目指していた．特に，資源予約および動的に QoS を変更するプロトコルにおけるプロトコル制御メッセージの実時間性を考慮した RtP や，ATM 上で RSVP を実現するための CSR 間の VC 確立プロトコルが挙げられる．今回は，既存のプロトコルの最新動向，および問題点をまとめる．

# 第 2 章

# リアルタイム通信プロトコルの現状

ここでは，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。詳しい紹介などは，前年度の報告書でもまとめてあるので，そちらを参照して頂きたい．今回は，この 1 年間でのプロトコルの進化を特に RSVP を中心として取り上げる．

## 2.1 RSVP

RSVP [1] は，Resource ReSerVation Protocol の略であり，ISI および PARC によって提案されているインターネット用資源予約セットアッププロトコルである．現在，Version 1 Functional Specification がインターネットドラフトとして提出されており，この 1 年間でもかなり更新されてきている．今後は，IPv6 のフローラベルによって実現できるフローを確立し，制御するためにインターネットでも広く利用される可能性が高い．

### 2.1.1 インターネットドラフト

現在，RSVP のもっとも最新のインターネットドラフトは，draft-ietf-rsvp-spec-12 である．これは，今年の 5 月に提出されたもので，ここ数ヵ月でも RSVP のドラフトはかなり頻繁に更新されている．

| | |
|---|---|
| rsvp-spec-12 | May 6, 1996 |
| rsvp-spec-11+ | April 15, 1996 |
| rsvp-spec-11 | March 18, 1996 |
| rsvp-spec-10 | Feb. 21, 1996 |
| rsvp-spec-9 | Feb. 12, 1996 |
| rsvp-spec-7 | July 7, 1995 |

表 2.1: RSVP のドラフト

実際にここ 1 年間でヘッダ構造なども改善されたり，2.2 節で挙げる問題点などを考慮す

るための拡張が数回行われている．現在の RSVP の共通ヘッダは簡単化され，図 2.1 のようになっている．オプションとして必要なデータは，この共通ヘッダの後にオブジェクトと呼ばれる単位で付加される．

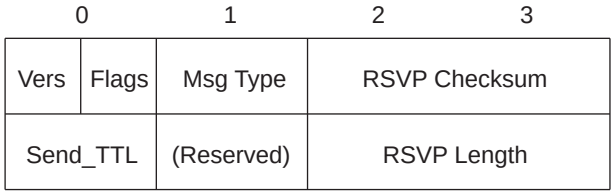


図 2.1: 現在の RSVP の共通ヘッダ

### 2.1.2　ドラフトの変化

表 2.1 に示すドラフトの変更点は，ほとんどの場合細かいバグフィックスである．ここでは，大きく変わっている場合を取り上げ，その変更点について述べる．

rsvp-spec-10 と rsvp-spec-11 で重点的に変わっている部分をドラフトから引用する．

- RSVP 層のフラグメント化機構が削除された．
  しかしながら，今後必要に応じてメッセージ長を 16 ビット以上に拡張できるように共通ヘッダは再構成された．
- IPv6 に関する記述を Introduction の章に追加した．
- ResvTear メッセージはサービスプリエンプションによってトリガーされる．
- トラフィック制御部は，更新された FLOWSPEC を返すことができる．(これにより，4 章の UPDATE TRAFFIC CONTROL 処理が大幅に変更された．)
- 2.3 節のディスカッションが書き直された．
- メッセージ処理規則が更新された．

rsvp-spec-09 と rsvp-spec-11 で重点的に変わっている部分をドラフトから引用する．

- 複数の POLICY_DATA オブジェクトの順番は問われなくなった．
- 共通ヘッダの length フィールドは，メッセージの全体の長さを表すよう変更した (3.1.1 節)．
- Message Id の定義をさらに具体化し，より完全な記述に変更した (3.1.1 節)．

- RSVP フラグメント化を新たに用意し，IP フラグメント化を許可しなくなった (3.1.1 節).
- 状態タイムアウトの粒度が規定されるようになった (3.6 節).

さらに，rsvp-spec-08 と rsvp-spec-09 で重点的に変更された部分をドラフトから引用する.

- 資源予約のエラー処理が根本的に変更され，second killer reservation problem を解決するようにした．ノードに対して，"blockade state" という新たな状態が導入された．この状態は， Error Code = 01 をもつ ResvErr メッセージによって生成され，資源予約更新メッセージを作成するマージ工程を制御する (2.6 および 3.4 節).
- SESSION オブジェクト内に二つのフラッグビットが設けられ，ある送信者への経路沿いに，RSVP をサポートしていないノードが存在することを受信者に示すことができる (2.9 および 3.7 節).
- オプションで INTEGRITY オブジェクトが導入され，共通ヘッダの後に設けられ，全てのフラグメントに含まれる (3.1 節).
- ERROR_SPEC オブジェクトには二つのフラッグビットが設けられた．InPlace と NotGuilty である (3.10 節).
- RSVP の実装は，メッセージ中のオブジェクトが任意の順番で配置されていてもできる限り受理し，メッセージ作成する場合は，BNF による順番に従うことを推奨するようにテキストが変更された.
- RSVP を使用する場合，TCP/UDP ポートフィールドが識別のために必要になる可能性があるので，一般的にはデータパケットの IP フラグメント化は不可能であるとテキストが変更された (1.2 節).
- 識別できないオブジェクトを処理する場合，そのオブジェクトを無視し，転送しないという，3 つ目の処理方法が導入された (3.9 節).
- 全ての一般トラフィック制御呼出は，インタフェース仕様を含むように変更され，Thandle がインタフェース特有になる.
- RSVP 用にインタフェースを無効にすることが可能となった (3.10.3 節).

ここで，注目したいことが，RSVP を利用した場合，IP フラグメント化ができないことである．現状の IPv4 に RSVP を導入した場合，フローを識別するために，上位層の識別子を利用するという手順をとったことになる．IPv6 には，フローラベルが導入され，フローを直接 IP の層で識別できるので，このような制約を持たせる必要はない.

## 2.2 RSVP の問題点

ここでは，今まで RSVP が発展していく中，問題点，もしくは短所として指摘されてきたことをいくつか取り上げる．現在のドラフトでは解決されているものもあるが，まだ取り上げられていないものもある．

### 2.2.1 One Pass

RSVP は，受信者指向資源予約セットアッププロトコルなので，受信者の数が増加してもスケーラビリティを維持することが可能であることが一つの長所であった．しかし，受信者側が発行する資源予約メッセージは，受信者から送信者に向かって (予め逆方向に伝搬された Path メッセージによって決定された経路に沿って) 伝搬されるのみであって，ST-2 や Tenet プロトコル体系で導入されている 2 パス (two pass) 機構と比較して，アプリケーションの要求する end-to-end サービスを実現することが困難であることが指摘された [2]．RSVP の機構は，2 パスに対して 1 パス (one pass) と識別するが，具体的には，1 パスの方法だと，end-to-end の遅延，およびジッタの上限を受信者が効果的に指定する方法がないことに問題がある．すなわち，資源予約メッセージが 1 パス，単一方向に伝搬されるだけでは，予約された資源によって，要求された end-to-end のサービスが得られるかは確実に分からない．しかし，当然 1 パス機構にも，長所はある．まず，マルチキャスト通信と組み合わせると効果的である．例えば，マルチキャストトリーを用いて，資源予約要求メッセージをマージすることによって，マルチキャストルーティングと同じ程度スケーラブルである．また，1 パス機構により，ルータによって管理される情報をソフトステート (soft state) 手法を用いて管理できる．すなわち，ルータに定期的更新メッセージなどを送信することによって，単純で強固な回復機構を実現することができる．

現在では，この 1 パス機構の欠点を克服するため，one-pass with advertising (OPWA) という機構が RSVP に導入されている．OPWA は，新たに ADV メッセージを導入し，リンク毎のサービス要求によって得られる end-to-end サービスを前もって受信者に知らせる (advertise: 宣伝する) 機構である．受信者に向かって下流に ADV メッセージが伝搬されると，ジッタや遅延などのパラメータを考慮し，その結果得られる end-to-end サービスが順々に求められる．ADV メッセージを受け取った受信者は，資源予約要求を発行する前に，この情報を参照することが可能となり，自分が得ることのできるサービスの目安として解釈できる．

最新のインターネットドラフトでは，ADV メッセージは，Path メッセージに統合されている．これにより，RSVP が必要とする制御メッセージの数を減らしている．Path メッセージには，Adspec という項目が設けられ，あるノードが Path メッセージを受け取ると，ローカルトラフィック制御機構に Adspec は渡され，更新された Adspec が返らせる．下流に転送される Path メッセージには，この更新された Adspec が含まれる．

### 2.2.2 Layer Violation

前述したが，RSVP では，必要に応じてトランスポート層 (TCP および UDP のポートなど) によって提供される情報を利用して逆多重化を実現している．このことにより，ルータは資源管理を行うために，IP 層からみるとデータ部に相当する特定なバイトを読み取れる必要がある．また，これにより，データの IP フラグメント化ができなくなってしまう．このように，RSVP はアドホックな手法を用いてフローのセットアップを実現している．今後，現在の提案されている RSVP が効果的にインターネット上で利用されるかは疑問に残る．

その他にも IPSEC によって保護されているデータストリームの取扱いも考慮しなくてはいけなくなった．RSVP が，トランスポート層の情報を必要とした場合，IPSEC を用いることによってこの情報は暗号化され，ルータ上の識別子として利用することができない．現在，セキュリティアソシエーション識別子を用いて逆多重化を行うことが提案されている．

### 2.2.3 Killer Reservation Problem

RSVP は，マルチキャストにおいて異なるサービス要求を発行することを可能にしているので，異種類の受信者をサポートしている．また，中継するルータなどで，パラメータによっては複数の要求はマージされ，上流へ伝搬されることがある．RSVP では，このときある要求によって他の要求が拒否される場合が生じる Killer reservation 問題が発生する可能性がある．

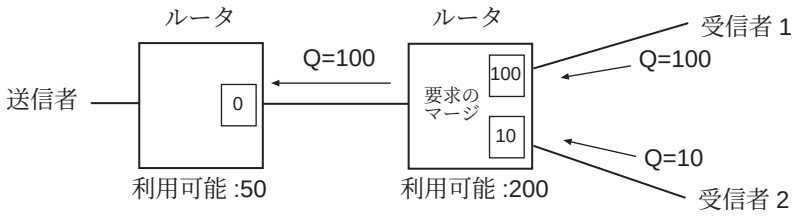


図 2.2: RSVP の killer reservation 問題

Killer reservation 問題は，2 種類ある．図 2.2 において，受信者 2 の要求した資源は予約済みで，受信者 1 が受信者 2 より大きな資源予約要求 (Q=100) を発行した場合，これらの要求は途中のルータでマージされ，上流に転送される．しかし，図 2.2 のような場合，マージされた要求はさらに上流にあるルータにおいて資源が足りない (資源が 50 までしか利用可能) ので拒否される．したがって，受信者 2 に対するサービスを維持することができない．この問題を解決することは極めて簡単で，ある要求に対するアドミッション制御が失敗した場合，既に予約されている資源はそのまま維持されるように処理される．

第二の問題は，受信者 1 のような大量の資源を確保するため要求を繰り返し発行し，失敗し続ける (左側のルータでは，利用可能な資源が 50 しかないので失敗する) ノードが存在するときに起きる．この場合，受信者 1 が要求を発行することを停止しない限り，受信者 2 の要求は満たされない．

このような問題を解決するため，新たに blockade state という状態が追加され，資源予約のマージ工程が変更された．ResvErr を受信すると，blockade state が作成され，上記の例の場合，受信者 1 が発行した大きな資源予約要求はマージされなくなる．

### 2.2.4 Application Characteristics

RSVP は多対多のマルチキャスト通信をサポートしている．多対多マルチキャストにおいて，資源を効率良く利用するためには，経路 (パス) に沿って資源を共有することが望ましい．RSVP もこのように資源の共有は実現している．しかし，多対多マルチキャストにおいて，複数の送信者が同時にデータを送信し，共有している資源が溢れてしまうことを考慮し，この際に受信者に対して具体的にどのようなサービスを提供するか規定しているプロトコルは RSVP を含めて存在しない．RSVP では，独自にフィルタという概念を導入し，受信者が特定な送信者を指定することができる．しかし，多対多のマルチキャスト通信のサポートを必要とするアプリケーションの性質を考えると，受信者が送信者を指定することは必ずしも容易に行えない．例えば，単純なビデオコンファレンスアプリケーションを考えた場合，必ず一人の送信者しかデータを送信しないというアプリケーションの性質を考慮するフレームワークが必要である．RSVP の場合，ある瞬間どの送信者からデータを受け取ればよいのかは受信者が判断できない．やはり，送信者側で協調し合う必要性もある．

このように RSVP は，受信者指向型のプロトコルとして開発されたが，送信者側の立場からアプリケーションの性質を考慮し，効率化を図ることは行われていない．動的に変化する環境に適用することは RSVP の特徴なのだが，その半面，パスが変化することなどによってアプリケーションが要求したサービスを必ず維持することができる保証はない．ここで示した多対多マルチキャストの例の場合でも同じで共有資源が溢れてしまった場合，必ずしもアプリケーションの要求したサービスは維持されない．

RSVP のこのような弱点は，RtMP というマルチキャスト資源予約プロトコルで解決しようとしている [3].

### 2.2.5 ATM との相性

RSVP は，受信者によって資源予約は行われるが，ソフトステートを導入しているので，ルータやスイッチなどで維持される情報は動的に変化する．それに対して，ATM は 2 パスで送信者側から VC を張るように設計されていて，さらに VC のパラメータは動的に変化することはない．これらの問題点は，[4] で詳しく取り上げられている．

## 2.3 RSVP の実装

RSVP のプロトタイプ実装は無償配布されている．最新バージョンは，3.2 で，RSVP の WWW ページから入手することが可能である．ここには，Sun が提供している Solaris 2.4 用の RSVP/CBQ[1]も入手することが可能である．これらは，両方ともに rsvp-spec-07 を元に実装が行われている．

RSVP 3.2 プロトタイプ実装は，SunOS 4.x で動作する Sun ワークステーション用に実装されたが，以下のプラットフォームにもポートされた．

- Sun ワークステーション Solaris 2.4
- SGI ワークステーション IRIX 5.3 (ホストサポートのみ)
- SGI ワークステーション IRIX 6.2
- PC 互換機 FreeBSD
- DEC Alpha ワークステーション Digital Unix version 3.0 (ホストサポートのみ)

また，以下のバージョンの IP マルチキャストをサポートしている．

- ルータ上で利用
  IP マルチキャスト pruning version をサポートしている．具体的には，PARC のリリース 3.5 に，95 年 6 月のアップデートを含んだカーネルおよび，mrouted リリース 3.6 のこと．
- ホスト上での利用
  どの IP マルチキャストのリリースでも使用できる．しかし，IP マルチキャスト 3.3 より以前のものの場合，UDP encapsulation が必要である．

RSVP のプログラミングインタフェースに関する文書も存在する [5]．これは，SunOS および BSD 系用の (元は，ISI の RSVP 3.1 から) 文書である．

また，プロトタイプの実装では，今後以下の機能が予定されているらしい．

- IPv6 サポート
- shared tree マルチキャストサポート
- アクセスコントロールおよびアカウンティング
- 予めスケジューリングする機構

[1]CBQ は，Class Based Queueing パケットスケジューラのことである．

## 2.4 RSVP の今後

今後 RSVP の仕様は以下のものをサポートするために拡張される予定である．

- アカウンティングおよびアクセス制御
- IP セキュリティを考慮したデータストリーム
- 診断 (diagnostic) メッセージ
- カプセル化用トンネル内の資源予約
- アクティブ QoS リンク層サポート
- 高性能ルーティング機能
- ユーザモデルの改良

ここでは，上記の項目のうち既にドラフトが提出されているものをいくつか紹介する．

### 2.4.1 アカウンティングおよびアクセス制御

バンド幅や物理資源などの要因によって発生する制約以外にアクセス制御を導入する方法についても考慮されるようになってきている [6]．ネットワークを運営している組織などでローカルに課金制度を導入してアクセス制御を行うことに利用される．このアクセス制御および課金は，現在のところ RSVP の仕様とは独立して考えられていて，ローカルポリシモジュール (Local Policy Module: LPM) によって実現されている．LPM により，RSVP メッセージ内に含まれる POLICY_DATA オブジェクトを元にして，通常境界ノードにおいてポリシアドミッション制御は実施される．

LPM アーキテクチャをサポートするためには，RSVP の仕様にいくつか変更点が必要である．以下に提案されている変更点を示す．

- LPM インタフェース (LPM 呼び出し，エラーコード，およびエラーに対する応答)
- API の変更
- 資源予約報告メッセージ (一般的な形式でも，LPM アーキテクチャに固有なものでも構わない)
- ポリシデータオブジェクト用のデフォルト処理

アカウンティングおよびアクセス制御を効果的に導入するためには，送信側と受信側用のポリシデータオブジェクト，およびアドバタイズ/フィードバック機構を実現する必要があるので，相方向でのやりとりが必要である．これらのデータを RSVP のオブジェクトとして実現する場合，RSVP メッセージ内にオブジェクトをカプセル化する方法が問題となる．上流方向メッセージには，Resv メッセージを利用できるが，下流に向かっては単純に Path メッセージを利用できない．Path メッセージの経路は，マルチキャストグループメンバシップ情報によって制御されるので，ある特定な次のホップに向かって正確に配送することができない．したがって，Path メッセージは，アカウンティングおよびアクセス制御用に利用されないであろう．そこで，[6] では，資源予約報告 (Reservation Report) メッセージを導入することを提案している．この資源予約報告メッセージは，ユニキャストを用いて下流に向かって送信されるので該当する次のホップのノードに正確に送られる．しかし，このように新たなメッセージを追加することによって，RSVP はスケーラビリティを維持することができるか疑問になってくる．このメッセージは他の用途のためにも利用できると，[6] には提案されているが，慎重に定めないと，スケーラビリティの問題が生じる．その他にも blockade state とポリシによってアクセス制御する方法を統合することなども考慮する必要がまだある．

### 2.4.2 診断メッセージ

[7] では，受信者から特定の送信者への経路上で維持される RSVP の状態に関する情報を収集するための診断メッセージが提案されている．現状の仕様だと，エンドホストが得ることのできる情報は，特定の要求が満たされなかったというエラーメッセージのみである．診断メッセージは，あくまでも情報を収集するための独立した RSVP 制御メッセージであり，ルータやホスト上の状態を変更するためには利用できない．

診断メッセージを導入することにおいて以下のことを目標にしている．

- パス状態が確立された後，パス上の全てのホップにおいて既に予約されている資源に関する RSVP 状態情報，またはまだ資源要求が発行される前の状態情報を収集することができること．
- 具体的には，パス上の各ホップにおけるマージされた資源予約要求に関する情報，タイマーの値，フロースペックなどの情報を収集することができること．
- パスが通過する非 RSVP クラウドのホップカウントの情報を収集することができること．
- packet implosion または explosion を発生しないこと．

診断パケットには，診断要求 (DREQ) と応答 (DREP) の 2 種類が定義されている．(共通ヘッダにおいて，DREQ は，Type=8 で，DREP は，Type=9 と提案されている.) パ

ケット発生数が莫大に増加しないように，診断パケットはユニキャストを使ってやりとりされる．また診断を要求するホストは必ずしもデータパスの受信側である必要はない．要求ホストは，パスのラストホップルータに DREQ パケットを送信する．DREQ パケットには，RSVP セッションとそのセッション内の送信者を特定する情報が含まれている．ラストホップルータは，該当するセッションに関する RSVP 状態情報を DREQ パケットに応答データブロックとして追加し，上流ノードにユニキャストする．DREQ パケットが送信者にたどり着くと，送信者はパケットタイプを DREP に変更し，DREQ を発行したノードに応答を返す．

この方式だと，パスが多数のホップから構成されていると，DREQ パケットに追加されていくデータも増加し，パス MTU サイズに納まらない可能性がある．IP のフラグメント/リアセンブリを利用すると，各ホップでその処理が必要なので，オーバーヘッドがかかる．そこで，デフォルトの MTU を定義し，DREQ パケットがこのサイズになることを途中ホップのルータが検出し，DREQ を発行した要求側に中間結果を返すという方法が提案されている．中間結果を返すと同時にルータは，上流に向かって縮小された DREQ を転送する．したがって，一つの DREQ 要求によって複数の DREQ パケットが発生する可能性がある．

DREQ パケットはパス状態が確立された後のみ転送される．パス状態が確立されていない場合は，traceroute などを用いてユニキャスト/マルチキャストのルーティングが正常に動作しているか確認できる．

## 2.5　Tenet Real-Time Protocol Suite

カルフォルニア大学バークレィ校で研究および開発されている Tenet Real-Time Protocol Suite [8] についても昨年度報告書で簡単に紹介した．ここでは，Tenet Real-Time Protocol Suite の現状，および Tenet グループが行っている関連研究について解説する．

## 2.6　Suite 1 のリリース

Tenet プロジェクトは， Tenet Protocol Suite 1 のソースコードを配布[2] している．このリリースには，データ転送用に RTIP と RMTP，またチャネル確立/削除用に RCAP といった 3 つのプロトコルが実装されている．また，これらのプロトコルはインターネットプロトコルと共存できるように設計されている．RCAP は，ユーザモードで動作し，RMTP および RTIP はカーネルの中に含まれている．現在 Ultrix 4.2A, Irix 4.0.5f, and BSD/OS 2.0 用のリリースが用意されていて，OSF-1 用のリリースも準備されている．

[2] ftp://tenet.cs.berkeley.edu/pub/src/tenet-suite1-0.8.tar.Z

## 2.7 Suite 2

Suite 1 に続いて，現在 Tenet プロジェクトでは Scheme 2 を設計し，これを Suite 2 として実装している．Suite 2 は，以下の点を考慮し，Suite 1 を拡張したプロトコル体系である．

- 効率の良いマルチパーティ通信を実現するための抽象概念および機構を提供する
- 柔軟なクライアントサービスインタフェースを提供する

ここで，新たなに導入した抽象概念は，ターゲットセットと呼ばれ，IP マルチキャストのホストグループと似ている．受信者は，要求する転送サービスに応じてターゲットセットに参加する．チャネルは，データのソースからターゲットセットの各メンバに対して確立される．RSVP と同じようにマルチキャスト，および資源を共有することによって効率化を図っている．また，クライアントサービスインタフェースの入力パラメータは，クライアントの要求する値と最悪値の対を指定することが可能であり，やりとりする必要回数を減らすことができる．

Suite 2 は，異種型インターネットワークにおいてマルチパーティリアルタイムネットワークアプリケーション実験を可能とする基板環境を提供する．また，複数のトラフィック仕様モデル，およびパケットスケジュール機構をサポートし，サードパーティクライアントに全てのアクションを開始することを許可することにより位置独立性も実現している．上記の機構は全て RCAP デーモンによって管理される．Suite 2 の RCAP デーモンは，オブジェクトセット (ターゲットセット，チャネル，位置およびネームサーバなど) および二階層メッセージディスパッチャからなる．現在のプロトタイプでは，チャネルは複数のデスティネーションに対して確立することが可能であり，デスティネーションは，ターゲットセットと対応するチャネルに動的に参加/脱退ができる．Suite 2 では，RSVP のようにチャネルを受信者からも確立することが可能となっている [9].

このプロトタイプは，資源の共有もサポートしている [10]．資源を共有する機構は，マルチパーティアプリケーションを効率良くサポートするため利用される．資源共有機構を導入することによって，アプリケーション固有の情報を利用して，ネットワーク資源をチャネル間において共有しながら，クライアントの必要とするサービスを保証することができる．チャネルグループという概念が提案され，種々のチャネル間関係を指定することのできる統合機構を提供する．これによると，資源の共有から得る利益はネットワークトポロジ，およびルーティングポリシに依存する．またシミュレーションでは，資源の共有機構を導入することによって，チャネルを受け入れる確立が上がることも分かっている．

## 2.8 ST-2+

ST-II の発展型，ST-2+ [11] は，送信者による資源予約機構に加えて，受信者指向機構も新たに導入している．今まではインターネットドラフトだったが，RSVP より先に RFC になっている．

## 2.9 まとめ

簡単に 3 つのプロトコルについて解説したが，今後，インターネットでは RSVP が広く使われる可能性が大きいと思われる．RSVP は，ソフトステートをルータに導入することにより，フロー用の資源予約を実現しているのに対して，他のプロトコルはハードステートを導入しているところに大きな違いがあると思われる．ソフトステートによる実現の方がインターネットのように動的に変化し続ける環境には適していると思われる．

最後に，ここでとりあげたプロトコルの最新情報は，以下の URL から入手できる．

- RSVP - http://www.isi.edu/div7/rsvp/rsvp.html
- Tenet プロトコル体系 - http://tenet.berkeley.edu/

# 第 3 章

# 今後の予定

RT ワーキンググループ自体は，3 月の合宿で活動を停止し，その中のいくつかのメンバによって，新たに具体的な目的をもった rt-bone WG が開始された．
以下に rt-bone の研究計画を示す．

現在，いくつかの資源予約プロトコルが提案されており，実装されているものも存在する．しかしながら，これらのプロトコルは，まだ広域ネットワークにおいて実証されておらず，ATM などの技術と効率良く統合する必要がある．そこで，我々は，広域網で実際に資源予約を利用するリアルタイム通信用のプロトコル，または機構を実現することを目指する．
以下に，簡単に予定している研究計画を示す．

1. まずはデータリンクに ATM を利用する．BSDOS に Efficient Networks, Inc. の ATM ドライバを組み込む．また RSVP を IPv6 上に実装する．
2. 上記の環境を用いて RSVP/ATM について検証する．
3. 資源予約に必要なキュー管理機構の検討および実装．RSVP に代わる新たな資源予約機構を提案し，プロトタイプを実装する．
4. 広域網での実験を行う．

さらに，メンバが係わっている実時間通信プロトコルのプロジェクト RtP [12] および RtMP [3] 関係からのアイディアを取り込み，その有効性を検討する．以下に，予定している項目を示す．

- RtP: プロトコル制御メッセージの実時間性
- RtMP: アプリケーションの性質を考慮した資源の共有機構

先に述べた RtMP は，RSVP とは異なった目的をもって設計されてきた．RSVP は動的に変化する環境に柔軟に適用する思想に基づいて設計されているが，RtMP は必ずアプリ

ケーションの要求するサービスをフローが維持される間は保証するという思想に基づいて設計されている．送信者側でアプリケーションの性質を考慮し，共有資源を効率良く利用するためのフレームワークなども提供している．しかし，これは送信者リストを各送信者上で分散管理することによって行われる．これにより，広域環境での効率，および送信者の数に対するスケーラビリティが問題となる．今後の実験ではこのような問題点も解決していきたい．