分散システム特論 レポート
RFC 2326
Real Time Streaming Protocol

情報工学専攻 荒木研究室
修士１年　鶴久 康治

1999年2月8日(月)

本レポートでは、RFC2326(Real Time Streaming Protocol)について述べる。また本レポート中で用いる用語については9節で述べる。

1 はじめに

近年ではインターネット上での連続メディア処理が、廉価なハードウェアと商用アプリケーションとで容易に行うことが可能となっている。連続メディアとは、一定時間間隔で符号化されてデジタルデータとして蓄積された、音声や動画のようなデータのことを指す。1992年にMBoneが出来て以来、マルチキャストプロトコルや、連続メディアデータの送信を支援するRTP(RFC1890)、RTPで送信される符号化形式の開発が進められてきた。またWWWの利用者の急激な増加に伴い、個人で作成した音声や動画のコンテンツをWWW上で公開することも増えてきた。

MBoneでは主に実時間で生成されるデータが送信され、これまで開発された技術を用いた応用の一つとして、インターネット上での会議システムがある。この実現のためのセッション管理用のプロトコルが必要である。また、WWW上では主に蓄積型データが扱われるが、本格的なVOD(Video on Demand)の実現のためには、HTTPだけでは機能が足りず、連続メディアデータに対する制御が不可能である。

これら両方の連続メディアデータ処理を統合し、不足していた機能を補うためのプロトコルとしてRTSPは提案された。

2 RTSPの概要

RTSPは複数の連続メディアストリームを開設し、制御する。RTSP自体は、連続メディアデータを送出するためのプロトコルでは無く、全く独立にして定義されたコマンド体系である。言い換えれば、RTSPはネットワークを介した、連続メディアデータに対するリモートコントロール機能を持つプロトコルである。主な特徴を以下に挙げる。

- アプリケーション層のテキストベースプロトコル
- 複数の連続メディアデータの伝送セッションを制御可能
- 連続メディアデータ用の伝送チャネルを問わない
- RTPベースの伝送機構を選択可能

RTSPは、表現方法や操作の面でHTTP/1.1に非常に類似している。これは、HTTPのセキュリティ機能やプロクシ機能、PICS (Platform for Internet Content Selection)などの下部構造を再利用するためである。また、今後のHTTPへの拡張の大部分が、そのままRTSPへも拡張することができる。

HTTPとの相違点を以下に挙げる。

- RTSPは新たなメソッドを採用し、異なったプロトコル識別子を持つ
- HTTPがステートレスであることに対し、状態を管理する必要がある
- RTSPは、サーバとクライアントの両方から要求を発行できる
- データは別のチャネル、プロトコルで伝送される
- ISO8859-1ではなくISO10646(UTF-8)を使用することを定義
- Request-URIは、常に絶対URIを含む

3 動作と状態遷移

連続メディアデータのストリームを受信するクライアントは、プレゼンテーション記述を取得する。取得する方法は、RTSPのDESCRIBEメソッド、HTTPや電子メールなどの他のプロトコル、コマンドラインからの入力などがある。その後、プレゼンテーション記述に示されたストリーム毎に、メディアサーバとRTSPメッセージを交換しセッションを開設する。さらにストリームを制御し、その後セッションを終了させる。(図1)

RTSPメッセージを交換するチャネルには、コネクションの概念が無い。その代わり、サーバはセッションID

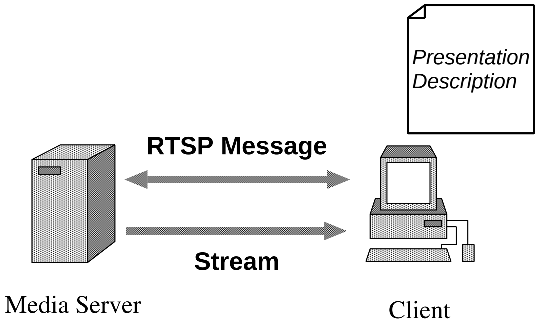

図 1: RTSP メッセージの交換

によってラベル付けされたセッションを管理する。クライアントは要求を発行する際に、以下のトランスポートコネクションを利用することができる。

① 持続的なコネクション

② 要求/返答のトランザクション毎のコネクション

③ UDP のようなコネクションレス型プロトコルの使用

但し③の場合、要求の再送処理は RTSP が行う必要がある。②のように、異なるトランスポートコネクションを流れる断続的な要求によって、ストリームを制御することが可能である。従って、セッションの開設/解放は、トランスポート層のコネクションの開設/解放に依存せず、セッションのステートマシンを管理する必要がある。

RTSP の状態遷移図は図 2 の通りである。サーバとクライアントの状態遷移は同一である。RTSP のメソッドのほとんどが状態遷移とは無縁のものであり、関係するのは SETUP、PLAY、RECORD、PAUSE、TEARDOWN のみである。

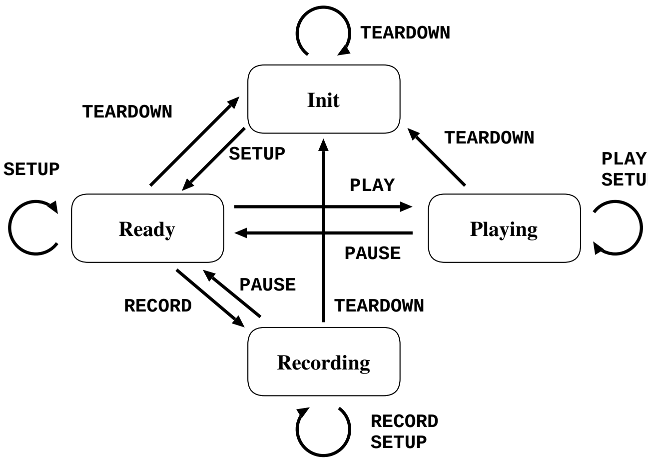

図 2: 状態遷移図

4 RTSP メソッド

以下に RTSP メソッドを挙げる。説明の冒頭の下線部は、方向 (S: サーバ / C: クライアント)、対象 (P: プレゼンテーション / S: ストリーム)、実装の必要性を示す。

OPTIONS

C↔S / P, S / 必須 (S→ C オプション)

標準でない要求を試すときに利用する。また URI に"*"を指定すると、サポートしているメソッドを Public ヘッダに含んだ応答を返す。

DESCRIBE

C→S / P, S / 推奨

要求した URI で識別される対象に関する記述を取り寄せる。サーバは要求された資源の記述を伴った応答を返す。

ANNOUNCE

C↔S / P, S / オプション

- C→S のとき
  要求した URI で識別される対象に関する記述を投げる。
- S→C のとき
  セッション記述を実時間的に更新。

新たなストリームがプレゼンテーションに加えられると、プレゼンテーション記述全体が再送される必要がある。つまり、プレゼンテーションからストリームの削除も可能である。

SETUP

C→S / S / 必須

ストリームに使われるトランスポート機構のセットアップや、途中でのパラメタ変更を行う。

PLAY

C→S / P, S / 必須

SETUP で指定した機構で、データ送出を開始する。SETUP の成功の応答があるまで、クライアントは要求してはならない。

PAUSE

C→S / P, S / 推奨

ストリームの配送を一時中断する。

TEARDOWN

C→S / P, S / 必須

ストリームの配送を停止し、サーバは資源を解放する。

GET_PARAMETER

SET_PARAMETER

C↔S / P, S / オプション

URI で指定したプレゼンテーションやストリームに関するパラメタの値を取得したり、設定したりする。

REDIRECT

S→C / P, S / オプション

クライアントに、他のサーバへ接続する必要があることを知らせる。Location ヘッダに新たな位置を示す。

RECORD

C→S / P, S / オプション

サーバにプレゼンテーションやストリームの記録を依頼する。URI は記録したデータの保管位置。

5 RTSP メッセージのフォーマット

RTSP のメッセージは、HTTP と同様に Request と Response の 2 種類である。フォーマットはそれぞれ以下の通りである。

```
Request  =  Request-Line
            *(  general-header
              | request-header
              | entity-header     )
            CRLF
            [ message-body ]
```

```
Response =  Status-Line
            *(  general-header
              | response-header
              | entity-header     )
            CRLF
            [ message-body ]
```

Request-Line には、メソッドと Request-URI、RTSP のバージョンが含まれる。Status-Line には、RTSP バージョンと、Status-Code、Reason-Phrase が含まれる。message-body には、要求のパラメタを表記したり、プレゼンテーション記述を含んだりするが、あまり使うことはない。

その他のヘッダは、ほぼ HTTP/1.1 と同義である。いくつかのヘッダの定義を削除し、RTSP 特有のヘッダを追加している。RTSP 特有の主なヘッダは、CSeq(要求/応答の組のシーケンス番号)、Session(セッション ID)、Transport(トランスポートパラメタ)、RTP-Info(RTP パラメタ) などである。

6 時間範囲の表現

PLAY や PAUSE メソッドで、Range ヘッダに時間を指定することができる。そこで用いられる時間範囲の表現には次の 3 通りがある。

① **SMPTE** 相対タイムスタンプ

SMPTE(Society of Motion Picture and Television Enginners: アメリカ映画テレビ技術者協会) が規定している相対タイムスタンプを用いて、各ストリームの先頭からの相対位置で表す表現形式。SMPTE の相対タイムスタンプは、最小単位が秒の時刻とフレーム番号で表す。

`"smpte="` smpte-time `"-"` [smpte-time]

例: `smpte=10:12:33:20-`
`smpte=10:07:00-10:07:33:05.01`

② ノーマルプレイタイム

ISO/IEC 13818-6 で規定されている時刻表現と同等。プレゼンテーションの始めを 0 として、そこからの時刻を秒とマイクロ秒で表現。

`"npt="` (npt-time `"-"` [npt-time])
| (`"-"` npt-time)

例: `npt=123.45-125`
`npt=12:05:35.3-`

③ 絶対時刻

ISO 8601 規格のタイムスタンプと同様に表現し、UTC を用いる。

`"clock="` utc-time ”-” [utc-time]

例: `clock=19970821T233420.25Z-`

これらのうち、②だけはサーバが必ずサポートしなければならない。

7 動作例

実際に RTSP のメッセージがどのように交換されるかを示す。Web サーバ上にあるプレゼンテーション記述をクライアントが取得し、メディアサーバからストリームを受信する例である。(C: クライアント / W: Web サーバ / A: 音声用サーバ/ V: ビデオ用サーバ)

(1) Web サーバからプレゼンテーション記述の取得

```
C→W:   GET /twister.sdp HTTP/1.1
       Host: www.example.com
       Accept: application/sdp

W→C:   HTTP/1.0 200 OK
       Content-Type: application/sdp

       v=0
       o=- 2890844526 2890842807 IN IP4 192.16.24.202
       s=RTSP Session
       m=audio 0 RTP/AVP 0
       a=control:rtsp://audio.example.com/twister/audio.en
       m=video 0 RTP/AVP 31
       a=control:rtsp://video.example.com/twister/video
```

(2) 音声用サーバのセットアップ

```
C→A:   SETUP rtsp://audio.example.com/twister/audio.en RTSP/1.0
       CSeq: 1
       Transport: RTP/AVP/UDP;unicast;client_port=3056-3057

A→C:   RTSP/1.0 200 OK
       CSeq: 1
       Session: 12345678
       Transport: RTP/AVP/UDP;unicast;client_port=3056-3057;
                                      server_port=5000-5001
```

(3) ビデオ用サーバのセットアップ

```
C→V:   SETUP rtsp://video.example.com/twister/video RTSP/1.0
       CSeq: 1
       Transport: RTP/AVP/UDP;unicast;client_port=3058-3059
V→C:   RTSP/1.0 200 OK
       CSeq: 1
       Session: 23456789
       Transport: RTP/AVP/UDP;unicast;client_port=3058-3059;
                                     server_port=5002-5003
```

(4) ビデオ用サーバへの再生要求

```
C→V:   PLAY rtsp://video.example.com/twister/video RTSP/1.0
       CSeq: 2
       Session: 23456789
       Range: smpte=0:10:00-

V→C:   RTSP/1.0 200 OK
       CSeq: 2
       Session: 23456789
       Range: smpte=0:10:00-0:20:00
       RTP-Info: url=rtsp://video.example.com/twister/video;
                                     seq=12312232;rtptime=78712811
```

(5) 音声用サーバへの再生要求

```
C→A:    PLAY rtsp://audio.example.com/twister/audio.en RTSP/1.0
        CSeq: 2
        Session: 12345678
        Range: smpte=0:10:00-

A→C:    RTSP/1.0 200 OK
        CSeq: 2
        Session: 12345678
        Range: smpte=0:10:00-0:20:00
        RTP-Info: url=rtsp://audio.example.com/twister/audio.en;
                                seq=876655;rtptime=1032181
```

(6) 音声用サーバへの終了要求

```
C→A:    TEARDOWN rtsp://audio.example.com/twister/audio.en RTSP/1.0
        CSeq: 3
        Session: 12345678

A→C:    RTSP/1.0 200 OK
        CSeq: 3
```

(7) ビデオ用サーバへの終了要求

```
C→V:    TEARDOWN rtsp://video.example.com/twister/video RTSP/1.0

        CSeq: 3
        Session: 23456789

V→C:    RTSP/1.0 200 OK
        CSeq: 3
```

8 最後に

発表では Real Networks 社の RTSP Reference Implementation を実装例として紹介したが、この RFC がインターネット・ドラフトの段階での実装であり、RFC の内容と違っている部分や足りない部分が多かった。商用アプリケーションでは、同社の RealPlayer G2 は、独自に開発してきた pnm プロトコルではなく RTSP を採用している。その他にもいくつかの RTSP 対応アプリケーションが出てきている。

この RFC では、ストリームとクライアントの 1 対 1 のセッションしか想定していない。しかし、RTSP の目標の一つである会議システムの実現のためには、シングルストリーム・マルチクライアントを考える必要がある。プロトコル自体はマルチクライアントに容易に拡張可能であると位置づけているが、どのクライアントが要求を出すのかといったポリシ制御の機能は定義されておらず、他の”social protocol” か”floor control mechanism” が必要であると記述がある。

9 用語

セッション ···

RTSP の完結したトランザクション。クライアントが SETUP メソッドによってトランスポート機構を確立してセッションが開始される。PLAY / RECORD メソッドによる操作の後、TEARDOWN メソッドによってセッションを終了する。

プレゼンテーション ···

複数のストリームの集合。

プレゼンテーション記述 ···

プレゼンテーションに関する情報。例えばストリーム毎の、符号化方式やネットワークアドレス、コンテンツの情報などである。フォーマットは規定されていないが、ストリームを識別するための URI は必ず含まれなければならない。SDP(Session Description Protocol：RFC 2327) によって表されることが多い。

URI (Uniform Resource Identifiers) ···

資源を一意に識別するための文字列。URL(Uniform Resource Locator) は URI の部分集合であり、資源の名前や属性で識別するよりもむしろ、アクセス機構 (ネットワーク「ロケーション」など) の表現によって資源を識別する下位集合である。