Universität Kaiserslautern
AG Datenbanken und Informationssysteme

Seminar im Sommersemester 2002:

Multimediale Informationssysteme

(http://wwwdbis.informatik.uni-kl.de/courses/seminar/SS2002/)

SMIL und RTP/RTSP

Torsten Lenhart
(t_lenhar@informatik.uni-kl.de)

Inhalt

1. Einleitung

Am Anfang war das Wort – zumindest was die Evolution des Internet betrifft. Computernetzwerke wurden entwickelt, um Rechner an unterschiedlichen Orten zu verbinden und ihnen so die Möglichkeit zu bieten, Daten auszutauschen und miteinander zu kommunizieren. Anfangs handelte es sich bei den ausgetauschten Informationen ausschließlich um Text, später kamen auch andere diskrete Medien hinzu. Mit der Entwicklung immer effizienterer Übertragungstechnologien und den gleichzeitig stetig wachsenden Multimediafähigkeiten von Computern drängt sich die Frage auf, ob man das Internet nicht auch verstärkt als Plattform zur multimedialen Kommunikation nutzen könnte.

Die Erfahrungen haben leider gezeigt, dass die bisher angewandten Techniken und Protokolle nicht ausreichen, um eine zufriedenstellende Funktionalität in diesem Zusammenhang zu liefern. Neben anderen sind vor allem zwei Probleme aufgetreten, die von der gegenwärtigen Infrastruktur des Internet nicht bewältigt werden können. Auf der einen Seite fehlt ein allgemeingültiger Standard zur Synchronisation von unterschiedlichen Typen von Medien. Die existierenden Konzepte sind meist wenig verbreitet, uneinheitlich und sehr komplex. Auf der anderen Seite ist die dem Internet zu Grunde liegende TCP/IP-Protokollfamilie ohne Erweiterungen nicht auf die Erfüllung von Echtzeitanforderungen von Verbindungen ausgelegt, die aber zum Beispiel bei Online-Audio-Video-Konferenzen benötigt werden.

Im folgenden werden drei Konzepte vorgestellt, welche dabei behilflich sein können, die Multimediafähigkeit des Internets zu erhöhen:
Zunächst wird im Kapitel 2 die Synchronized Multimedia Integration Language (SMIL, ausgesprochen wie engl. "smile") vorgestellt. Es handelt sich hierbei um eine Sprache auf Basis der *Extensible Markup Language* (XML), welche die räumliche und zeitliche Synchronisation unterschiedlicher Medienströme in Multimediapräsentationen regelt.
Die nachfolgenden Kapitel behandeln den Protokollaspekt, wobei zunächst das Real-Time Transport Protocol (RTP) erläutert wird, welches für Verbindungen mit echtzeitabhängigen Daten verwendet wird.
Im Kapitel 4 wird schließlich das Real-Time Transport Streaming Protocol (RTSP) betrachtet. Hierbei handelt es sich um ein Protokoll auf Anwendungsebene, das zur Kontrolle von Medienströme verwendet wird.
Im letzten Kapitel werden schließlich die besprochenen Konzepte noch einmal zusammengefasst und ihre Beziehung zueinander erläutert.

2. Synchronized Multimedia Integration Language (SMIL)

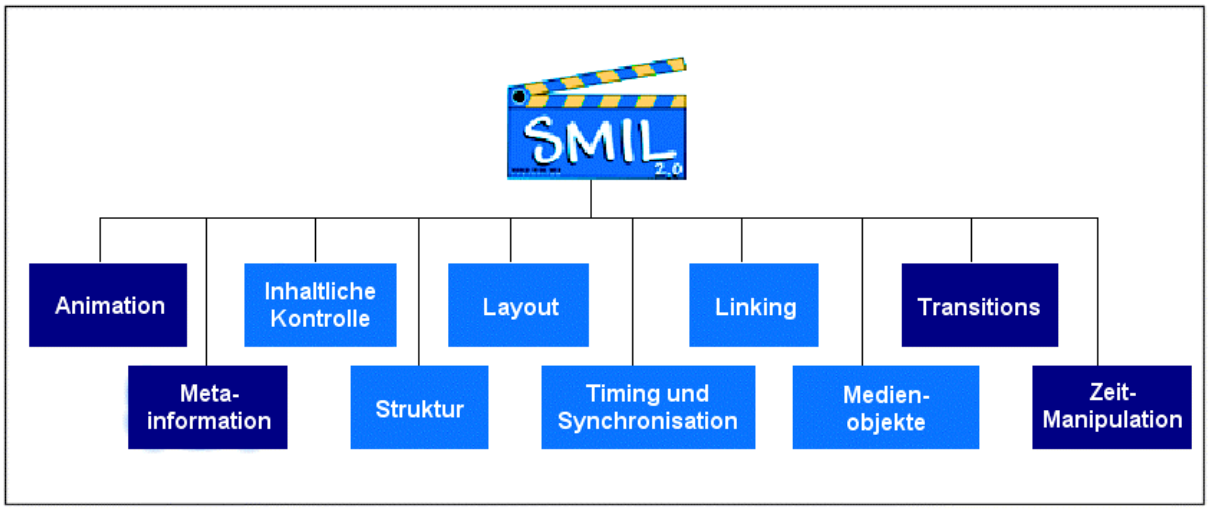
Bei SMIL handelt es sich um eine deklarative, XML-basierte Sprache zur Synchronisation und Integration unterschiedlicher Medienobjekte. Ihr Design ist so entwickelt, dass sie kompatibel zu anderen XML-verwandten Standards wie z.B. Cascading Style Sheets, XPointer und XLink ist, wodurch gegenseitige Einbettung ermöglicht und Wiederverwendung unterstützt wird. Die Entwicklung von Präsentationen mit SMIL gestaltet sich als vergleichsweise einfach. Man muss kein erfahrener Programmierer sein und es werden auch keine teuren Authoring-Tools benötigt, es genügt vielmehr ein einfacher Texteditor. Ähnlich wie bei HTML soll die einfache Handbarkeit zu einer weiten Verbreitung der Sprache führen.

2.1 Entwicklungsgeschichte

Im Dezember 1995 wurde beim *World Wide Web Consortium* (W3C) mit der Entwicklung einer neuen Sprache zum Erstellen von Multimedia-Präsentationen im Rahmen des Internet begonnen. Ziel war es, dem Wirrwarr an herstellerspezifischen und meist unnötig komplexen Techniken auf diesem Gebiet ein Ende zu bereiten. Zuvor erforderten Multimedia-Inhalte auf Webseiten entweder aufwändige Programmierung in Dynamic HTML oder die Erstellung von Skripten mit Sprachen wie JavaScript. Ansonsten blieb nur der Weg über spezielle Multimedia-Werkzeuge, deren Endprodukte lediglich über zu installierende Plug-Ins im Browser darstellbar waren.
Im November 1997 wurde schließlich die Version 1.0 der "Synchronized Multimedia Integration Language" [2] vom W3C verabschiedet und kurz darauf veröffentlicht. Die *W3C SMIL Working Group* arbeitete bei der Entwicklung eng mit Forschungseinrichtungen und führenden IT-Firmen zusammen, darunter unter anderem Microsoft, Real Networks, Intel und Phillips.
Im darauffolgenden März kamen die ersten, zum Teil kostenfreien Umsetzungen von SMIL innerhalb von Softwareplayern- und Browsern (GRiNS, SOJA, RealPresenter Basic) auf den Markt. Von Anfang an war klar, dass die in SMIL 1.0 verwirklichten Konzepte für die schnell wachsenden Anwendungsmöglichkeiten nicht ausreichen würden und es zeitnah einer Erweiterung des Sprachschatzes bedurfte. Im August 1999 wurde daraufhin der erste öffentliche Entwurf von SMIL 2.0 vorgestellt, welcher mit einigen Veränderungen und Erweiterungen versehen im August 2001 als endgültige Version [1] verabschiedet wurde. Die Länge des Spezifikationsdokuments hatte sich von etwa 30 Seiten bei SMIL 1.0 auf das fünfzehnfache (> 500 Seiten) erhöht.
Bei SMIL handelt es sich zwar um einen bislang außer in Fachkreisen wenig bekannten, aber dennoch weit verbreiteten Standard im World Wide Web. Eine Unterstützung von SMIL ist auf schätzungsweise 200 Millionen Rechnern weltweit zu finden (Stand 10/2001), was hauptsächlich aus der Implementierung der Sprache in Programmen wie RealPlayer G2, QuickTime ab Version 4.1 und Internet Explorer ab Version 5.5 resultiert. Für die Zukunft ist eine noch weitergehende Integration von SMIL auf Softwareebene geplant

2.2 Modulgruppen in SMIL

Es handelt sich bei SMIL also um eine Kollektion von XML-Elementen und Attributen, die man zur Beschreibung der räumlichen und zeitlichen Koordination von einem oder mehreren Medienobjekten innerhalb einer Präsentation benutzen kann. Generell kann jedes SMIL Element mit einem Attribut `id` versehen und dadurch eindeutig gekennzeichnet werden. Der Wert des Attributs wird in einer Zeichenkette festgelegt.
In SMIL 2.0 werden zehn wichtige funktionelle Gruppen von Modulen (Abb. 1) festgelegt. Die Module wiederum bestehen aus Elementen und Attributen. Dadurch wird unter anderem die Wiederverwendung von SMIL Funktionalität in anderen XML-basierten Sprachen auf der einen und die Einbindung dieser verwandten Sprachen in SMIL selbst auf der anderen Seite unterstützt. Die Wiederverwendungsmöglichkeit besteht also in zwei Richtungen, was sich in der praktischen Anwendung als extrem aufwandssparend erweisen kann.



(Abb. 1 – Modulgruppen in SMIL 2.0)

In Abb. 1 sind die verschiedenen Modulgruppen aufgeführt. Im folgenden sollen nun besonders relevante Gruppen und ihre wichtigsten Elemente und Attribute vorgestellt werden. Es sei gleich zu Anfang darauf hingewiesen, dass hier nur ein kleiner Ausschnitt der SMIL Funktionalität behandelt werden kann.

2.2.1 Struktur

SMIL-Dokumente bestehen auf hoher Abstraktionsebene betrachtet aus folgenden Komponenten:
Eingeleitet wird jedes SMIL-Dokument durch einen Tag `<smil>`, gegebenenfalls in Verbindung mit einem oder mehreren Attributen. Ein häufig verwendetes Attribut ist `xmlns`, mit dem man den XML-Namespace für das Dokument festlegen kann. Das Ende eines Dokuments wird durch das zum Anfangstag korrespondierende Endtag `</smil>`

gekennzeichnet, dazwischen existieren ein Head- und ein Body-Bereich. Der Head-Bereich enthält Informationen, die für das zeitliche Verhalten der Präsentation nicht von belang sind. Dazu gehören autorenspezifische Inhaltsdefinitionen, Meta- und Layoutinformationen. Im Body stehen hingegen die unter temporalen Gesichtspunkten und in bezug auf Vernetzung relevanten Informationen.
Die entsprechenden Elemente und Attribute sind im Modul Strukturmodelle spezifiziert. Ein SMIL-Dokument besitzt also grob folgende Struktur:

```
<smil xmlns="http://www.w3.org/2001/SMIL20/Language">
   <head>
      ...
   <head>
   </body>
      ...
   </body>
</smil>
```

2.2.2 Medienobjekte

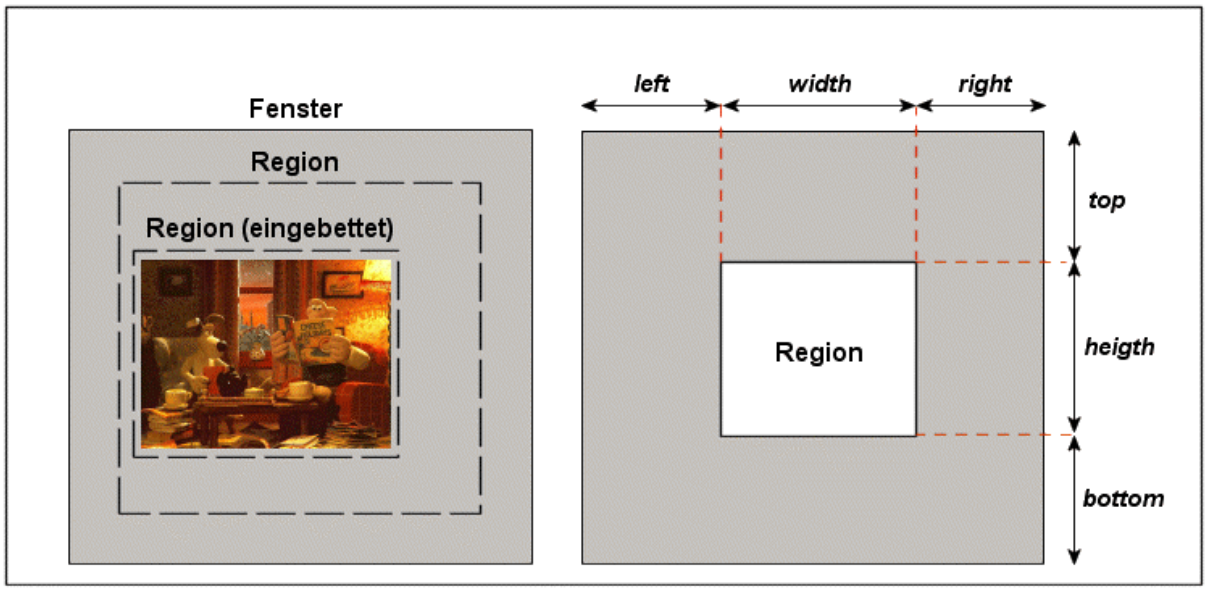
Die zum Erstellen der Präsentation miteinander zu integrierenden Medien und die Orte, wo sie zu finden sind, werden über Konstrukte der Modulgruppe Medienobjekte bestimmt. Die wesentlichen Elemente sind hier `ref`, `animation`, `audio`, `img`, `text`, `textstream` und `video`, wobei `ref` ein generisches Medienobjekt ohne nähere Festlegung des Inhalts referenziert. Zu jedem Element gehören die Attribute `src` und `type`. In `src` wird ein *Uniform Resource Identifier* (URI) angegeben, über den das Medienobjekt zugreifbar ist, in `type` wird festgelegt, in welchem Format es vorliegt. Das Attribut `type` ist dabei abhängig vom in `src` verwendeten Protokolltyp (FTP, HTTP, RTSP). Das `type` Attribut kann weggelassen werden, wenn gesichert ist, dass Informationen über den Dateityp vom Betriebssystem oder dem Server geliefert werden können. Eine Interpretation des Dateinamens ist nicht vorgesehen. Ein Beispiel für die Festlegung von Medienobjekten könnte so aussehen:

```
...
<img src="welcome.jpg" ... />
<audio src="http://www.w3c.org/SYMM/joe-audio.wav" … />
<video src="rtsp://www.cwi.nl/SMIL/video.rm" id="film"… />
...
```

2.2.3 Layout

Die Modulgruppe Layout enthält Konstrukte, um die räumliche Anordnung der auszugebenden Medienobjekte festzulegen. Hauptelement dieser Gruppe ist `layout`, in dem das Layoutschema für die gesamte Präsentation festgelegt wird. Unterelemente von Layout sind `region` und `root-Layout`. Mit den Attributen `width` und `heigth` im `root-Layout` Element kann die Größe des entsprechenden Fensters in Pixeln festgelegt werden,

mit dem Attribut `backgroundColor` die Hintergrundfarbe und mit `title`, wie der Name schon vermuten lässt, der Titel.
Jedes Präsentationsfenster kann eines oder mehrere `region` Elemente enthalten. Neben denselben wie bei `root-Layout` existieren für `region` Elemente noch einige zusätzliche Attribute. Dazu gehören `top`, `bottom`, `left` und `right`, welche die Position der Region im Präsentationsfenster bestimmen. In Abb. 2 wird die Bedeutung der Attribute veranschaulicht. Diese müssen nicht immer alle definiert sein, weil sich z.B. bereits aus `width`, `heigth`, `top` und `left` die eindeutige Position für eine Region ergibt. Es ist auch möglich, weniger als zur eindeutigen Platzierung notwendige Angaben zu machen, in diesem Fall werden default-Werte angenommen.
Mit dem Attribut `regionName` wird ein eindeutiger Bezeichner für eine Region gewählt. Über das Attribut `region` kann dadurch beispielsweise in den Medienobjekt-Elementen eindeutig die Region identifiziert werden, in der das korrespondierende Medium erscheinen soll.



(Abb. 2: Layoutkonzepte)

Innerhalb von Regionen sind hierarchische Anordnungen zulässig, sie können beliebig viele Unterregionen enthalten (vgl. Abb. 2). Dadurch ist es beispielsweise möglich Inhalte, die logisch zusammengehören, in einer gemeinsamen Region zu platzieren. Bei Veränderungen der Position dieser Region werden die in ihr enthaltenen Objekte automatisch mit verschoben, die räumliche Anordnung zwischen ihnen bleibt gleich.
Beispiel:

```
...
<layout>
   <root-Layout title="Beispiel" width="800" height="600"/>
   <region regionName="linkeHaelfte"
    top="5" left="5" width="295" heigth="590"/>
   <region regionName="rechteHaelfte"
    top="5" left="400" width="295" heigth="590">
      <region regionName="oberesViertel"
       top="5" left="5" width="295" heigth="290"/>
      <region regionName="unteresViertel"
```

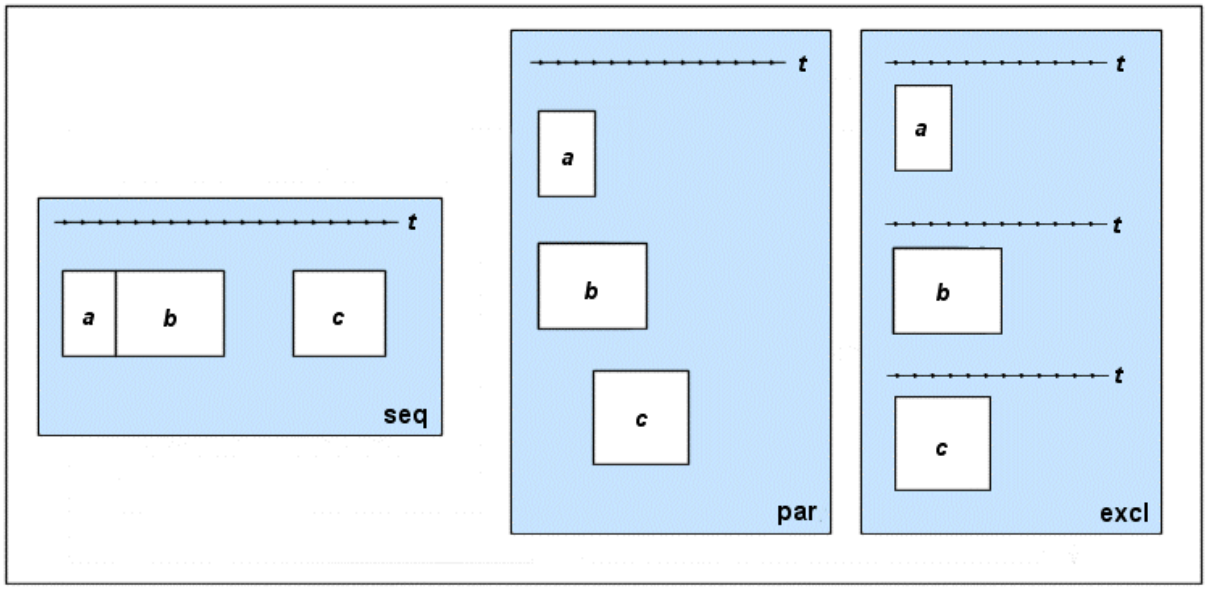
```
        top="305" left="5" width="295" heigth="290"/>
    </region>
</layout>
...
<img src="welcome.jpg" region="oberesViertel" ... />
...
```

2.2.4 Timing und Synchronisation

Bei der Gruppe "Timing und Synchronisation" handelt es sich um den Kern der SMIL Spezifikation, da hier die Elemente und Attribute zur zeitlichen Synchronisation der verschiedenen Medien innerhalb der Präsentation definiert werden.
Es existieren drei Synchronisationselemente, die unterschiedliche Ablaufszenarien unterstützen:

- Das `seq` Element startet die darin enthaltenen Medienobjekte eines nach dem anderen in Form einer Sequenz.
- Das `par` Element erlaubt hingegen parallele Abläufe. Es können mehrere Elemente gleichzeitig abgespielt werden.
- Das `excl` Element erlaubt zu einem Zeitpunkt nur ein Element im aktiven Zustand, gibt aber keinerlei Reihenfolge vor. Die enthaltenen Elemente sind meist mit Ereignissen verknüpft und sie dürfen erst präsentiert werden, wenn das zugehörige Ereignis eingetreten ist.

Die Unterschiede verdeutlicht Abb. 3, wobei besonders darauf zu achten ist, dass bei `par` und `seq` eine einzige Zeitleiste für alle Objekte existiert, bei `excl` aber jedes Objekt seine eigene besitzt.



(Abb. 3: Timing Container in SMIL)

Ähnlich wie Regionen können die Synchronisationselemente, die man auch "Timing Container" nennt, beliebig ineinander verschachtelt werden. Dadurch können nicht nur zwischen Objekten sondern auch zwischen Objektgruppen zeitliche Beziehungen eingeführt werden.
Neben den Timing Containern werden in dieser Modulgruppe Attributerweiterungen für Medienobjekte definiert. Zu den wichtigsten gehören `begin`, `dur`, `end` und `repeat`.
Im Attribut `begin` wird festgelegt, zu welchem Zeitpunkt nach aktiv werden des Elements mit dessen Präsentation begonnen werden soll. Dabei ist es möglich, eine bestimmte Sekundenzahl oder aber auch ein Ereignis anzugeben. Im letzteren Fall wird dann erst mit der Vorführung des Objekts begonnen, wenn das spezifizierte Ereignis eingetreten ist. Bei einem Ereignis kann es sich um das Ende oder den Beginn der Anzeige eines anderen Medienobjekts handeln, aber auch Einflüsse von Aussen wie z.B. ein Mausklick können als Ereignis definiert werden. Das Verknüpfen des Anzeigebeginns mit Ereignissen ist ein wichtiger Aspekt bei Timing Containern vom Typ `excl`.
Mit dem Attribut `dur` kann man die Dauer der tatsächlichen Präsentation eines Elementes bestimmen. Neben konkreten Zahlen kann der Wert des Attributs auch auf "*indefinite*" gesetzt sein, was bedeutet, dass die Präsentation des entsprechenden Objekts so lange fortgesetzt wird, bis ein übergeordneter Timing Container beendet wird. Das Ende der Dauer des aktiven Zustands kann mit dem `end` Attribut definiert werden, wobei dieses wie `begin` auch mit dem Eintritt von Ereignissen verknüpft werden kann. In `repeat` kann schließlich festgelegt werden, wie oft die Präsentation wiederholt werden soll.

Allein die Definition der Gruppe "Timing und Synchronisation" benötigt über hundert Seiten. Es werden noch weit komplexere Konzepte angeboten, aber für das grundlegende Verständnis sollten die hier dargestellten ausreichend sein.

Beispiel (korrespondierend zu Abb. 3):

```
...
<seq>
   <img dur="3s"  begin="0" src="a.jpg" ... />
   <img dur="6s"  begin="3" src="b.jpg" ... />
   <img dur="5s"  begin="0" src="c.jpg" ... />
</seq>
...
<par>
   <img dur="3s"  begin="0" src="a.jpg" ... />
   <img dur="6s"  begin="3" src="b.jpg" ... />
   <img dur="5s"  begin="0" src="c.jpg" ... />
</par>
...
<excl>
   <img dur="3s"  begin="x.close" src="a.jpg" ... />
   <img dur="6s"  begin="y.close" src="b.jpg" ... />
   <img dur="5s"  begin="z.close" src="c.jpg" ... />
</excl>
...
```

2.2.5 Inhaltliche Steuerung

Eine weitere erwähnenswerte Funktionalität von SMIL ist die Unterstützung einer dynamischen Anpassung der Darstellung von Medienobjekte. Dies wird durch das `switch` Element geleistet. In Kombination mit sogenannten Testattributen, die lokale Systemgegebenheiten beschreiben, handelt es sich um ein wirksames Konzept zur Optimierung des Präsentationsablaufs. Beispielsweise kann in Kombination mit dem Attribut `systemBitrate` festgestellt werden, ob die verfügbare Bitrate für das Abspielen eines Streaming-Video ausreicht oder ob alternativ eine Sequenz von Bildern angezeigt werden soll. Ausgehend von den lokalen Gegebenheiten kann so für jeden User ein angepasster und optimaler Ablauf der Präsentation gewährleistet werden. Dies verdeutlicht folgendes Beispiel:

```
...
<switch>
   <video src="rtsp://www.cwi.nl/SMIL/video.rm"
    systemBitrate="115200" />

   <seq systemBitrate="57344">
      <img src="http://www.cwi.nl/SMIL/img1.jpg" dur="5s"/>
      ...
      <img src="http://www.cwi.nl/SMIL/img1.jpg" dur="5s"/>
   </seq>
</switch>
...
```

Falls das entsprechende System eine Bitrate von 112 kbit oder höher gewährleisten kann, wird das Video abgespielt, falls es sich um weniger als 112 kbit, aber mehr als 56 kbit handelt, wird eine Bildersequenz angezeigt.
Einige Testattribute wie das im Beispiel verwendete werden von SMIL vorgehalten, in SMIL 2.0 gibt es jedoch für jeden Autor die Möglichkeit, eigene zu erstellen und dadurch das `switch` Element ganz individuell einzusetzen.

2.2.6 Linking

Das in SMIL angewandte Linking-Konzept ist eng mit dem von HTML verwandt, was schon bei einem Blick auf die verwendeten Elemente und Attribute auffällt. Aber auch hier liegen die Unterschiede im zeitlichen Bereich. In beiden Sprachen gibt es ein Element `a`, durch welches ein Link repräsentiert wird. Im dazugehörigen Attribut `href` wird bei SMIL die URI des Medienobjekts angegeben, auf das sich der Link bezieht. Während aber bei HTML lediglich in räumlicher Hinsicht an eine bestimmte Stelle des Zieldokuments gesprungen werden kann, z.B. durch Anker, ist es in SMIL außerdem möglich, direkt auf einen definierten Punkt entlang der Zeitachse zu verweisen. Die referenzierte Präsentation verhält sich dann genauso, als ob der gewählte Zeitpunkt im Ablauf auf konventionelle Weise erreicht worden wäre.
Links ins SMIL sind nicht nur im Bezug auf das Ziel zeitabhängig definierbar, der Link kann sich auch auf eine ganz bestimmte Sequenz innerhalb eines Videos beziehen. Die Verweise werden nur dann aktiv, wenn in der spezifizierten Zeitspanne auf das Video geklickt wird.

2.2.7 Sonstige Modulgruppen

Die vorgestellten Gruppen bilden wie oben erwähnt nur einen Ausschnitt aus den in SMIL spezifizierten. Des weiteren wurden hier lediglich die wichtigsten Elemente und Attribute eingeführt, um einen Eindruck davon zu geben, wie ein SMIL-Dokument aussehen kann. Die Mächtigkeit und funktionelle Vielfalt von SMIL lässt sich deshalb aus dem hier gegebenen Überblick lediglich erahnen.
Besonders hervorzuheben wären noch das leistungsfähige Animationskonzept, welches von SMIL angeboten wird. Animationen können zum einen als unabhängige Objekte von außen eingebunden werden, z.B. als Java-Applets, oder werden in SMIL selbst definiert und erstellt. Die Sprache liefert hierfür wirkungsvolle Elemente und Attribute, allerdings nimmt in den entsprechenden Modulen die für SMIL typische, relativ leichte Verständlichkeit ein wenig ab.

2.3 SMIL Profile

Es existieren verschiedene SMIL Sprachprofile, mit denen erreicht werden soll, das für verschiedene Anwendungsformen eine optimale und angepasste Form von SMIL zur Verfügung steht. Ein Profil lässt sich dabei grob als eine Sprache beschreiben, für die ein Browser oder Player entwickelt werden kann. Im Moment existieren lediglich vier wesentliche Profile:

- *SMIL 2.0 Language Profile (SMIL Profile):* Das Grundprofil, das fast alle Modulgruppen und fast alle Module enthält. Dieses ist für den universellen Einsatz vorgesehen.
- *SMIL 2.0 Basic Language Profile (SMIL Basic):* Dieses Profil ist vor allem für mobile Endgeräte mit knappen Ressourcen wie z.B. PDAs und in naher Zukunft auch Handys vorgesehen.
- *XHTML+SMIL:* Hauptgedanke hinter diesem Profil ist die Integration von SMIL Konzepten und Modulen, vor allem in Bezug auf Timing, in XHTML.
- *SMIL 1.0:* Durch die Rückwärts-Kompatibilität können Präsentationen, die für dieses Profil verfasst sind, auch in neueren Playern abgespielt werden.

Die Entwicklung von neuen Profilen, auch von Außenstehenden, ist dabei seitens des W3C ausdrücklich erwünscht und wird von diesem vorangetrieben.

2.4 Zusammenfassung SMIL

SMIL bietet umfangreiche Möglichkeiten zur Entwicklung von web-gestützten Multimedia-Präsentationen. Die besonderen Vorteile der Sprache liegen in ihrer relativ leichten Einsetz- und Erlernbarkeit, ihrem mächtigen Timing-Konzept und ihrer Abstammung von XML, was wesentlich zur massiven Unterstützung von Wiederverwendbarkeit führt. SMIL ist weder

abhängig von einer Plattform, noch an bestimmte Formate gebunden und kann dadurch kontinuierlich den wechselnden Gegebenheiten und Strömungen angepasst werden. Man kann davon ausgehen, dass ihr Einsatz in der Praxis in den nächsten Jahren kontinuierlich zunimmt, vor allem wenn man den steigenden Anteil von Multimedia-Inhalten im Rahmen der menschlichen Kommunikation betrachtet.

3. RTP (Real-Time Transport Protocol)

Das Real-Time Transfer Protocol (RTP) ist ein anwendungsintegriertes Protokoll, mit dessen Hilfe das Versenden von verzögerungssensitiven Inhalten, wie Video- und Audioströmen, über unterschiedliche Netzwerke unterstützt wird. Die von ihm dabei geleisteten Dienste sind neben anderen die Rekonstruktion der Reihenfolge von Daten, die Identifikation von Inhalten, die Anpassung der Übertragungsgeschwindigkeit und die Unterstützung von Sicherheitsaspekten. Eine mit Hilfe von RTP realisierte Verbindung besteht häufig zwischen mehreren Benutzern über Multicast, das Protokoll ist aber auch bei dedizierten Verbindungen zwischen zwei Kommunikationspartnern einsetzbar. RTP wird oft zur Realisierung von Audio- und Audio/Videokonferenzen über das Internet eingesetzt.
Obwohl es sich um ein relativ neues Protokoll handelt, ist es weit verbreitet und in vielen Anwendungen zu finden, darunter QuickTime, NetMeeting und verschiedenen Playern von Real.

3.1 Entwicklungsgeschichte

Die Wurzeln von RTP liegen in ersten Experimenten auf dem Gebiet "Voice over Networks" in den siebziger und achtziger Jahren. Die Network Research Group des Lawrence Berkeley National Laboratory verwendete 1991 innerhalb eines von ihr selbst entwickelten Tools für Audiokonferenzen ein Protokoll, welches später unter dem Namen RTP Version 0 bekannt wurde. Im Dezember 1992 wurde die erste offizielle Version veröffentlicht, aber zunächst noch nicht standardisiert. Nach mehreren neuen Entwürfen und grundlegenden Veränderungen verabschiedete die *Internet Engineering Steering Group* (IESG) RTP Version 2 schließlich im November 1995 als geprüften Internetstandard und veröffentlichte diesen im Rahmen der RFCs 1889 [8] und 1890.

3.2 Warum braucht man RTP?

Zunächst gilt es zu klären, warum die ursprünglich in der TCP/IP-Protokollfamilie definierten Transportprotokolle nicht ausreichen und was dementsprechend die Einführung eines eigenen Protokolls für Übertragungen mit Echtzeit-Anforderungen legitimiert.
Eine denkbare Alternative wäre es, ganz einfach das in der Transportschicht des Internet am häufigsten verwendete Transmission Control Protocol (TCP) zu benutzen. TCP gewährleistet eine zuverlässige Ende-zu-Ende-Übertragung, d.h. der Focus liegt darauf, dass keine Pakete verloren gehen. Bei Echtzeit-Anwendungen ist eine gewisse Verlustrate tolerierbar, viel wichtiger ist es hier, ob der Großteil der Pakete in festgelegten Zeitschranken ankommt. Insbesondere sind zu alte Pakete gänzlich nicht mehr zu gebrauchen und könnten verworfen werden. Würde man TCP als Transportprotokoll verwenden, wäre es möglich, dass diese nicht mehr relevanten Pakete die Auslieferung der relevanten maßgeblich verzögern könnten, was insbesondere im Lastfall zu ernsthaften Problemen führt. Deshalb und wegen seiner

unzureichenden Unterstützung von Multicast-Verbindungen ist TCP für den Einsatz in Echtzeitanwendungen nicht geeignet.
Das hinsichtlich der Übertragung unsichere Äquivalent User Datagram Protocol (UDP) erweist sich in diesem Zusammenhang als wesentlich kompetenter, insbesondere durch seine ausgeprägten Multiplexing- und Fehlererkennungsdienste. Allerdings fehlen hier einige für Multimediaverbindungen unabdingbare Anforderungen. Zum Beispiel werden keine Sequenznummern vergeben, obwohl es während des Transports zu Umordnungen kommen kann. Dies kann im konkreten Einsatz bei beispielsweise einer Audiokonferenz zu massiven Störungen in der Ausgabe der Audioströme führen.
Neben den genannten Schwächen der Alternativen bietet sich auch dahingehend die Einführung eines eigenen Protokolls an, da man auf diese Art einige zwar nicht dringend notwendige, aber sehr interessante Funktionalitäten verwirklichen kann. Einige davon werden im weiteren Verlauf des Textes noch vorgestellt werden.
RTP setzt gemeinhin auf UDP auf, um dessen Vorteile bei gleichzeitiger Beseitigung der Schwächen zu nutzen. Eine mögliche Verwendung von RTP in einem IP-Netzwerk hat die Einbettung der RTP-Pakete in UDP-Pakete zur Folge, welche wiederum in IP-Pakete verpackt werden (vgl. Abb. 4). Da aber RTP fast keine Annahmen über die Dienste der unteren Schichten macht, kann es auch in Verbindung mit vielen anderen Protokollen genutzt werden.

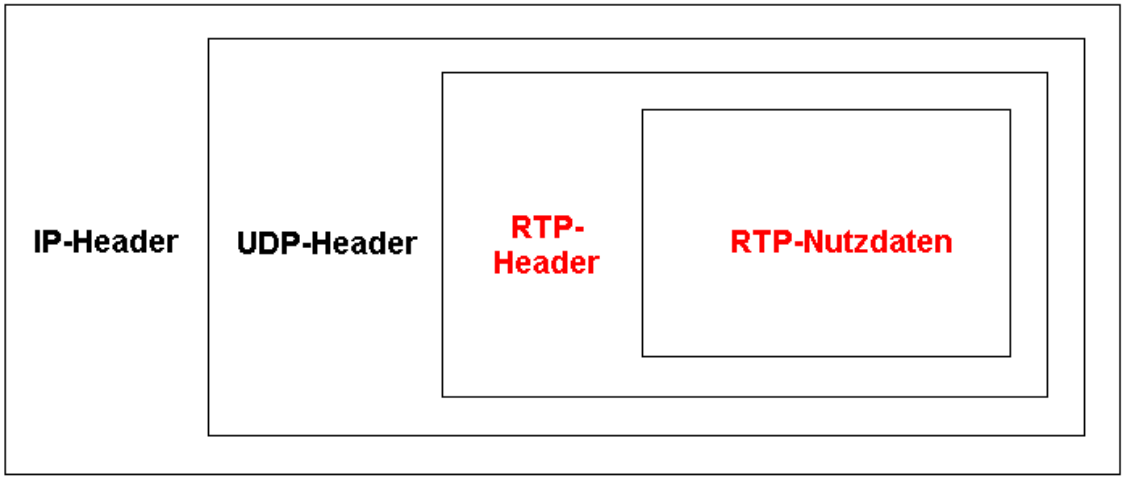


Abb. 4: Einbettung von RTP in die Internet-Protokollstruktur

3.3 Merkmale

Eine RTP-Sitzung ist eine Verbindung zwischen mehreren Teilnehmern (i.a. >2) mittels RTP kommunizieren. Handelt es sich bei den übertragenen Daten um Multimediainhalte, so muss jedes Medium eine eigene RTP-Sitzung eingerichtet werden, da pro Sitzung nur ein Nutzdatenformat spezifiziert werden kann. Multimedia-Verbindungen bestehen dann also aus mehreren einzelnen Sitzungen, die durch von RTP gelieferte Informationen synchronisiert werden können
Mit Hilfe von RTP wird eine Ende-zu-Ende-Verbindung aufgebaut, die speziell zum Transport und Austausch echtzeitabhängiger Daten gedacht ist. Es muss dabei betont werden, dass dabei keinerlei Mechanismen ergriffen werden, um Dienstgüte-Garantien zu gewährleisten. Es wird weder die pünktliche Ankunft eines Pakets garantiert, noch werden

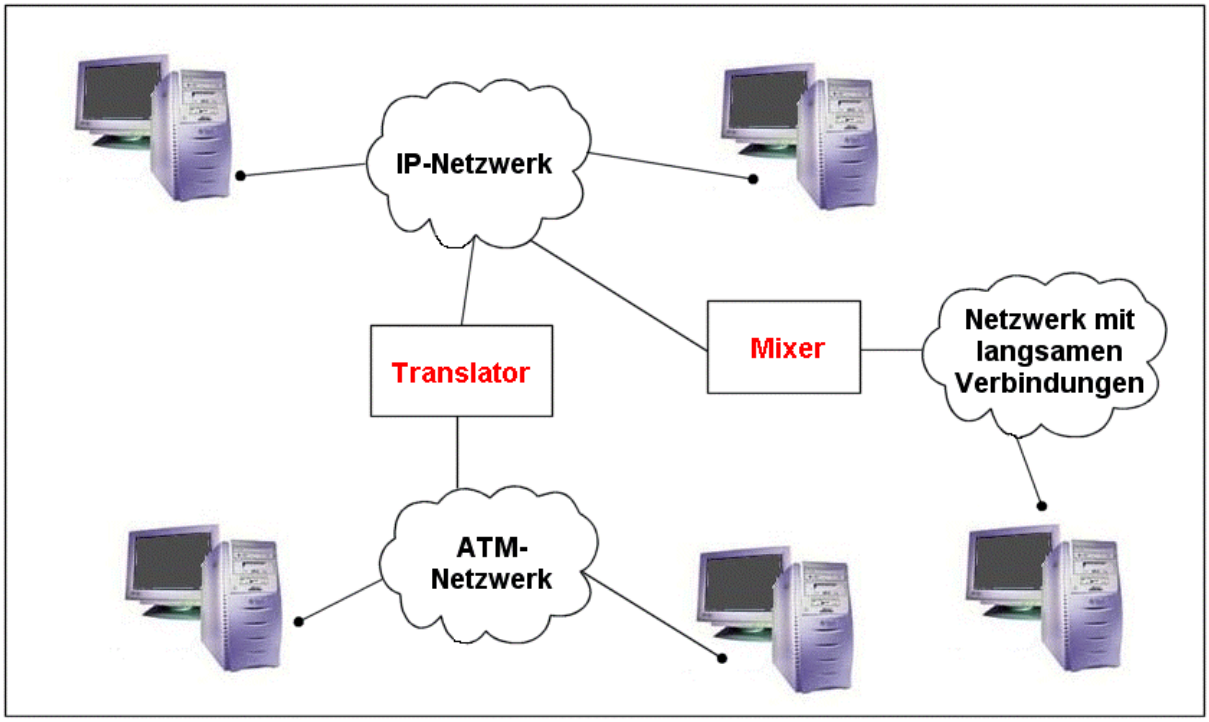
Flusskontrolle und Stauvermeidung in irgendeiner Form ergriffen. Auf den ersten Blick mag dies verwundern, aber bei genauerer Betrachtung kann kein Ende-zu-Ende-Protokoll die pünktliche Auslieferung eines Pakets garantieren, weil solchen Protokollen die Kontrolle über Ressourcen in Switchs und Routern verwehrt ist. Es hängt also von den Diensten der unteren Schichten ab, ob und wie Quality-of-Service erhöht und Garantien eingehalten werden.
Dennoch liefert RTP speziell auf Übertragungen von Real-Time-Daten zugeschnittene Funktionalitäten. Hier wäre zunächst vor allem die Vergabe einer Sequenznummer an jedes Paket zu nennen, wodurch nach Umordnungen die richtige Reihenfolge herausgefunden werden kann. Des weiteren werden durch das Vorhandensein von Zeitstempeln Kontrollmechanismen zur Synchronisation verschiedener Datenströme unterstützt.
Im Gegensatz zu anderen Protokollen ist es bei RTP üblich, dass es von jeder Anwendung separat implementiert wird und nicht wie sonst der Fall, global vom Betriebssystem. Oft wird RTP auch direkt in die Anwendungsausführung integriert, anstatt als eigene Schicht unterhalb der Anwendung zu fungieren. Diese enge Verknüpfung mit dem jeweiligen Programm wird durch folgendes Konzept weiter vertieft: Die Definition von RTP in RFC 1889 dient nur als Grundgerüst und muss um ein RTP-Profil erweitert werden. In einem solchen Profil können neben den möglichen Nutzdatenformaten auch Erweiterungen und Modifikationen festgelegt werden, die auf die entsprechende Anwendung zugeschnitten sind. Ein vorgegebenes und oft verwendetes Profil für Audio- und Videodaten ist in RFC 1890 festgelegt.

3.4 RTP Control Protocol (RTCP)

Im Rahmen von RTP wird neben oben genannten Paar aus Grundgerüst und Profil zusätzlich ein Hilfsprotokoll mit Namen RTP Control Protocol (RTCP) spezifiziert, wobei dessen Semantik ebenfalls vom Profil abhängen kann. Die wesentlichen Funktionen von RTCP sollen im folgenden näher erläutert werden.
Hauptaufgabe von RTCP ist es, Informationen über die zur Verfügung stehende Dienstgüte bereitzustellen, wobei diese Informationen durch periodisches Versenden von Kontrollpaketen gesammelt werden. Das dabei gewonnene Wissen ist für Sender und Empfänger wichtig, da entsprechend der Qualität der Verbindung die Art der zu sendenden Nutzdaten angepasst werden kann (Streaming Video mit hoher Qualität bei guter, Streaming Video mit geringerer Qualität bei mittlerer und eine Folge von Bildern bei schlechter Verbindung).
Eine weiterer Funktion von RTCP ist der Austausch von Informationen über die Teilnehmer einer RTP-Verbindung, insbesondere die Vergabe eines kanonischen Namens als global eindeutige Identifikation für eine Quelle. Es können aber auch andere Angaben wie der Name, die E-Mail-Adresse oder die Telefonnummer des Benutzers übergeben werden.
Die ersten beiden Aufgaben setzen voraus, dass alle Teilnehmer einer Verbindung kontinuierlich Kontrollpakete senden. Die Rate, mit der diese Kontrollpakete gesendet werden, kann dynamisch der Anzahl der Teilnehmer angepasst werden, damit auch bei vielen Benutzern der Anteil der Protokollpakete nicht über eine definierte Grenze ansteigt. Im allgemeinen liegt diese Grenze bei 5% des Gesamtaufkommens an Paketen.

3.5 Mixer und Translator

In RTP werden zwei zwischengeschaltete Geräte spezifiziert, die den Ablauf einer Verbindung unterstützen sollen: Mixer und Translator.
Bei RTP-Sitzungen zwischen vielen Benutzern tritt oft das Problem auf, dass Benutzer über Verbindungen mit unterschiedlicher Leistung angeschlossen sind. Das hat zur Folge, dass die zu übertragenden Daten an Teilnehmer mit geringer und hoher Bandbreite verteilt werden müssen. Will man den Echtzeitcharakter bewaren und keine Benutzer aufgrund von technischen Vorraussetzungen ausschließen, bleibt unter normalen Umständen nur die Alternative, die Größe der Daten und damit die Qualität an dem Teilnehmer mit der schlechtesten Verbindung anzupassen. Um dieses Problem zu beheben, sind in RTP sogenannte Mixer definiert.
Bei einem Mixer handelt es sich um einen RTP-Zwischenknoten, welcher die Pakete mehrerer Quellen (z.B. verschiedene Audioströme) empfängt, die Sendereihenfolge und das Timing rekonstruiert, auf die verfügbare Bandbreite abstimmt und die resultierenden Ströme zu einem Strom mischt. Dieser wird schließlich an einen oder mehrere Empfänger weitergeleitet. Ein Mixer passt also die Größe seines Ausgangsstroms an die Gegebeneinheiten der Ausgangsleitung zu den entsprechenden Empfängern an. Bei einer Audiokonferenz könnte das so aussehen, dass mehrere Ströme mit qualitativ guten Sprachdaten zu einem minderer Qualität gemixt werden, der dann aber auch entsprechend kleiner ist.



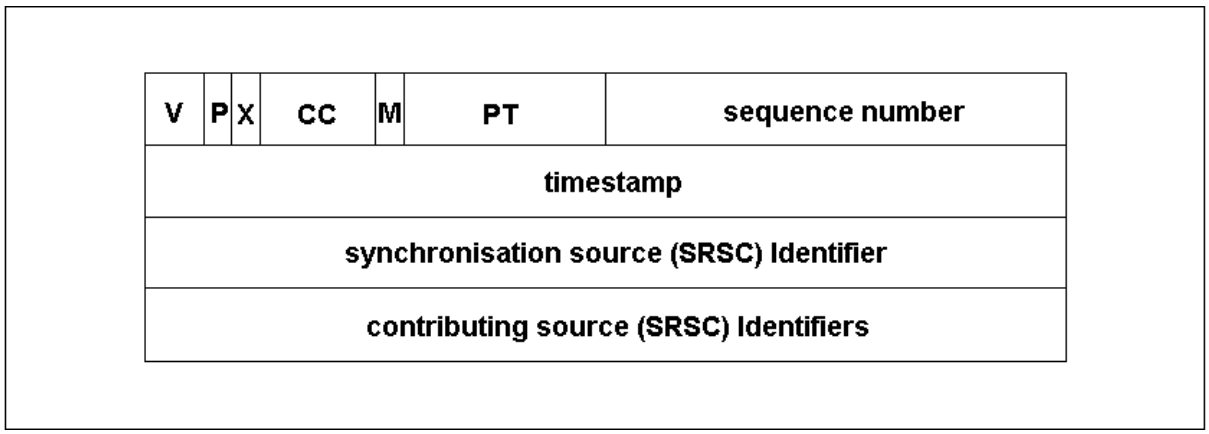
(Abb. 5: RTP-Session über verschiedene Subnetze)

Es könnte sein, dass manche Teilnehmer einer RTP-Sitzung nicht direkt erreichbar sind, weil sie sich in Subnetzen mit unterschiedlicher Protokollstruktur befinden. Um diese dennoch einbinden zu können, wird ein sogenannter Translator eingesetzt, der die RTP-Pakete entsprechend den unterschiedlichen Netzen anpasst. Außerdem können solche System eingesetzt werden, um Benutzer hinter einer Application-Level Firewall zu erreichen, was

dann durch die Tunneling-Technik realisiert werden kann. Des weiteren können über einen Translator Ver- und Entschlüsselungstechniken angewandt werden, wenn zum Beispiel sicherheitskritische Daten über eine unsichere Verbindung geschickt werden sollen. Translator haben im Grunde eine ähnliche Arbeitsweise wie Mixer, allerdings gibt es jeweils nur einen Ein- und Ausgabestrom und die Rohdaten werden, außer in seltenen Fällen, zur Typkonversion keiner Veränderung unterzogen.
Ein Beispiel für mögliche Einsatzgebiete einer netzwerkübergreifenden RTP-Session wird in Abb. 5 dargestellt.

3.6 Protokollheader

Im folgenden werden die Bedeutungen der verschiedenen Felder innerhalb des RTP-Headers vorgestellt. Als Überblick dient Abb. 6:



(Abb. 6: RTP-Protokollheader)

***version* (V), 2 Bit:** Verwendete RTP-Version (gegenwärtig 2)

***padding* (P), 1 Bit:** Ist dieses Bit gesetzt, enthält das Paket an seinem Ende eines oder mehrere Padding-Oktete, die nicht Teil der Nutzdaten sind.

***extension* (E), 1 Bit:** Fall gesetzt, enthält das Paket genau eine Headererweiterung

***CSRC count* (CC), 4 Bit:** Anzahl der nachfolgenden CSRC-Identifier

***marker* (M), 1 Bit:** Die Bedeutung des Markers wird durch die Profile definiert

***payload type* (PT), 4 Bit:** Mit Hilfe dieses Flags kann die Applikation die Nutzdaten und ihr Format identifizieren. Im Profil aus RFC 1890 sind einige Formate für Video- und Audioformate vordefiniert, zusätzliche Formate können von Benutzern selbst definiert werden

***sequence number*, 16 Bit:** Mit ihrer Hilfe kann die Sendereihenfolge von Paketen wiederhergestellt und ein eventueller Verlust registriert werden.

***timestamp*, 32 Bit:** Hier wird der genaue Zeitpunkt der Ermittlung des ersten Bytes der entsprechenden Nutzlast des Pakets angegeben. Dabei können mehrere aufeinanderfolgende Pakete denselben Zeitstempel aufweisen, z.B. wenn sie zum selben Videobild gehören. Deshalb kann der Zeitstempel die Sequenznummer

nicht ersetzen. Außerdem wird der Zeitstempel zur Synchronisation mehrerer RTP-Sitzungen benutzt.

***SSRC Identifier*, 32 Bit:** Bezeichner des Erzeugers des entsprechenden Paket. Es kann sich dabei sowohl um ein Programm auf dem System eines Teilnehmers oder um einen Mixer handeln. Der SSRC Bezeichner unterscheidet sich vom kanonischen Namen (s. RTCP) dahingehend, dass er für jedes Programm neu vergeben wird, der kanonische Namen hingegen ist für ein Teilnehmersystem immer gleich.

***CSRC Identifier*, 0-15 Elemente zu 32 Bit:** Dieses Feld ist nur belegt, wenn es sich bei der sendenden Quelle um einen Mixer gehandelt hat. Es handelt sich um eine Liste mit allen SSRC Bezeichnern der Quellen, die einen Beitrag zu diesem gemixten Strom geleistet haben. Die Anzahl der Elemente wird durch das CC Feld festgelegt, die Liste durch den Mixer gefüllt.

3.7 Zusammenfassung RTP

Das Real-Time Transport Protocol ist mittlerweile ein weit verbreitetes und oft verwendetes Protokoll zur Übertragung von echtzeitsensitiven Daten. Besonders in Anwendungsbereichen wie Online-Konferenzen in Ton oder Bild und Ton findet es breite Anwendung. Durch die Definition von Mixern und Übersetzern ist es hoch flexibel und kann auf unterschiedliche Benutzer zugeschnitten werden.

4. Real-Time Streaming Protocol (RTSP)

Das Real-Time Streaming Protocol (RTSP) wird zur Kontrolle von Strömen kontinuierlicher Medien eingesetzt. Es handelt sich dabei nicht um ein Content-Protokoll, d.h. die Daten selbst werden mit Hilfe eines anderen Protokolls versendet, lediglich die Regelung des Stromverkehrs wird von RTSP übernommen. Oftmals handelt es sich bei dem zugrundeliegende Protokoll, welches für den Transport der Nutzdaten verantwortlich ist, um das oben vorgestellte RTP, es kann sich aber auch um viele andere Transportprotokolle handeln.

4.1 Entwicklungsgeschichte

RTSP wurde basierend auf umfassenden Erfahrungen aus dem Bereich "Streaming Media" als Gemeinschaftsprojekt von Real Networks, Netscape Communications und der Columbia University entwickelt. Der erste Entwurf von RTSP wurde im Oktober 1996 eingereicht und im April 1998 als geprüfter Standard von der Internet Engineering Task Force (IETF) veröffentlicht (RFC 2326 [15]).

4.2 Merkmale

Wie oben erwähnt, handelt es sich bei RTSP um ein Kontroll-Protokoll, welches keine Inhalte transportiert. Deshalb liegt es nahe, dass man bei RTSP nicht von Paketen sondern von Nachrichten (Messages) spricht. Im Rahmen dieser Nachrichten wird das Verhalten eines oder mehrerer Medienströme geregelt, d.h. es wird bestimmt, wann ein Strom abgespielt, pausiert oder beendet wird. RTSP dient also, bildlich gesprochen, als "Netzwerk Fernbedienung" für Medienströme.
RTSP ist sowohl für Multicast- als auch Unicast-Einsatz vorgesehen. Verbindungen bestehen immer zwischen mindestens einem Medienserver, der die Daten kontinuierlich sendet und mindestens einem Medienclient, der diese konsumiert. Im Kontext von RTSP werden logisch zusammengehörige Ströme, zum Beispiel Video- und korrespondierender Audiostrom als Präsentation bezeichnet, Präsentationen mit mehreren beteiligten Servern und Clients heißen Konferenzen. Für jede Präsentation existiert serverseitig ein *Presentation Describtion File*, in dem Informationen über die Präsentation selbst und Angaben über die übertragenen Medien vorgehalten werden.
Generell sind drei verschiedenen Verwendungsszenarien für RTSP vorgesehen:

- Übertragung von kontinuierlichen Mediendaten: Der Client kann mediale Daten nachfragen und den Server zur Erstellung eine Verbindung und zum Senden der Daten auffordern.
- Einladung eines Medienservers zu einer Konferenz: Ein Server kann zu einer Konferenz "eingeladen" werden, um die von ihm vorgehaltenen Daten einzubringen oder neue Daten aufzuzeichnen

- Zusätzliche Daten: Ein Server kann seine Clients informieren, dass zwischenzeitlich zusätzliche Mediendaten eingetroffen sind.

Die Rolle, die das *Hypertext Transport Protocol* (HTTP) für HTML-Seiten einnimmt, füllt RTSP für Medienströme aus. Beide Protokolle sind sich in Syntax und angebotenen Dienste sehr ähnlich und ebenso wie bei HTTP wird bei RTSP eine Quelle durch eine URL identifiziert. Unterschiedlich ist, dass es sich bei HTTP um ein zustandsloses Protokoll handelt, RTSP aber sehr wohl verschiedene Zustände kennt und ein Medienserver diese entsprechend berücksichtigen muss. Außerdem sind bei RTSP sowohl der Server als auch der Client berechtigt, Anfragen abzusetzen, was bei HTTP nicht unbedingt der Fall ist.
Zur Realisierung der Dienste und Funktionalitäten sind Methoden definiert, von denen einige im folgenden vorgestellt werden:

- SETUP: Der Client fordert den Server auf, spezifizierte Daten zu senden.
- PLAY: Der Client fordert den Server auf, mit dem Senden der Daten über die zuvor über SETUP allokierte Verbindung zu beginnen.
- PAUSE: Der Client weist den Server an, den Strom kurzzeitig zu unterbrechen, ohne aber serverseitig Ressourcen freizugeben.
- TEARDOWN: Der Client fordert den Server auf, den Strom endgültig zu beenden und die belegten Ressourcen freizugeben.
- DESCRIBE: Die Beschreibung eines spezifizierten Medienobjekts wird an den Client geschickt.
- REDIRECT: Der Server informiert den Client, dass dessen Datenanforderung an einen anderen Medienserver weitergegeben werden muss.

4.3 Zusammenfassung RTSP

Das Real-Time Streaming Protocol (RTSP) ist eine wirkungsvolle Möglichkeit zur Kontrolle von Medienströmen. Seine Unabhängigkeit von den Protokollen der tieferen Schichten unterstützt eine breite Anwendung, ebenso die Implementierung auf vielen unterschiedlichen Betriebssystemen. Durch seine Ähnlichkeit mit dem wohl-bekannten HTTP wird ein schneller Einstieg in den Umgang mit dem Protokoll ermöglicht.
In der Praxis unterstützen Firmen wie SUN Microsystems, Apple und IBM den Einsatz von RTSP. Trotzdem könnte seine Zukunft als plattformunabhängiges Protokoll durch die ablehnende Haltung von Microsoft bedroht werden.

5. Zusammenfassung und Bewertung

Das Internet und andere herkömmliche Netzwerke sind zur Übertragung von textuellen Daten entwickelt worden und können nicht ohne weiteres für Multimedia-Kommunikation eingesetzt werden. Die vorgestellten Techniken bieten in diesem Zusammenhang eine Möglichkeit, die Lücke zwischen tatsächlich vorhandener und benötigter Funktionalität zu schließen. Mit Hilfe des Real-Time Transport Protocol (RTP) ist es möglich, echtzeitsensitive Daten adäquat zu übertragen, wobei die entstehenden Datenströme durch das Real-Time Streaming Protocol (RTSP) kontrolliert und gesteuert werden können. Schließlich kann man durch Verwendung der XML-basierten Sprache SMIL Präsentationen erzeugen, in denen die übertragenen Medien synchronisiert dargestellt werden.
Die beiden Protokolle und SMIL sind gut aufeinander abgestimmt und arbeiten problemlos zusammen, was nicht zu letzt daran liegen mag, dass an ihrer Entwicklung teilweise die gleichen Firmen und Personen beteiligt waren. Durch die Plattformunabhängigkeit von SMIL und die Flexibilität von RTP und RTSP in Bezug auf die Protokolle der unteren Schichten wird eine breite Anwendung wesentlich erleichtert. Es bleibt dennoch abzuwarten, ob sich diese Kombination für die Bewältigung der zukünftigen Aufgaben im Bereich der multimedialen Kommunikation durchsetzen wird.

Anhang A: Literaturangaben

SMIL:

[1] "Syncronized Multimedia Integration Language 2.0", W3C Recommendation, World Wide Web Consortium, August 2001
[2] "Syncronized Multimedia Integration Language 1.0", W3C Recommendation, World Wide Web Consortium, November 1997
[3] D. Bultermann, "SMIL 2.0 – Overview, Concepts and Structure", IEEE Multimedia, Oktober-Dezember 2001
[4] L. Rutledge, "SMIL 2.0: XML for Web Multimedia", IEEE Internet Computing, September/Oktober 2001
[5] C. Piemont, "Ein Lächeln fürs Web", c't Magazin für Computertechnik, 20/1998
[6] L. Rutledge, "Tonangebend", iX Magazin für professionelle Nachrichtentechnik, 10/1999
[7] K. Pihkala, "SMIL 2.0", X-Miles-Workshop, September 2001

RTP:

[8] H. Schulzrinne et al., "RTP: A Transport Protocol for Real-Time Applications", Internet Draft based on RFC 1889, November 2001
[9] H. Schulzrinne, RTP Homepage, http://www.cs.columbia.edu/~hgs/rtp/
[10] Prof. Dr. Reinhard Gotzhein, Skript zur Vorlesung "Kommunikationsanwendungen und –technologien", Universität Kaiserslautern, Sommersemester 2001
[11] Prof. Dr. Reinhard Gotzhein, Skript zur Vorlesung "Systemsoftware", Universität Kaiserslautern, Wintersemester 2000/2001
[12] Prof. Dr. Paul Müller, Skript zur Vorlesung "Multimediasysteme", Universität Kaiserslautern, Wintersemester 2001/2002
[13] C. Liu, "Multimedia over IP: RSVP, RTP, RTCP, RTSP", Internes Papier, Ohio State University, Juli 2000
[14] V. Hallivuori, "Real-Time Transport Protocol (RTP) Security", Seminar on Network Security, Helsinki University of Technology, 2000

RTSP:

[15] H. Schulzrinne et al., "Real Time Streaming Protocol", Internet Draft based on RFC 2326, Februar 1998
[16] H. Schulzrinne, RTSP Homepage, http://www.cs.columbia.edu/~hgs/rtsp/
[17] C. Liu, "Multimedia over IP: RSVP, RTP, RTCP, RTSP", Internes Papier, Ohio State University, Juli 2000