

# Multimedia over IP

The idea of multimedia communications is very appealing—text, graphics, sound, and video all combined to enrich the ways in which we communicate and to transform communications from a one-dimensional to a multidimensional experience. Many believe that multimedia content will drive the continued growth of the Internet.

Traditionally, voice has been carried on circuit-switched lines. During the past decade, work on VoIP has grown, for a variety of reasons that will be discussed in Section 4.1. More recently, interest in multimedia over IP is rapidly rising. With increasing availability of high-bandwidth links and advances in QoS technologies, a future is often envisaged in which multimedia telecommunications carried over IP networks becomes an ordinary way of communicating.

Our concept of multimedia over IP includes the more familiar concept of VoIP as a special case. Multimedia over IP, as compared to a single media over IP, adds one other complication—the various media streams need to be synchronized. However, whether the subject is a single medium like voice or multimedia, we are dealing with real-time traffic.<sup>1</sup> Real-time traffic has stringent requirements on the playback of the voice and video at the receiving side. Even if a stream is broken into segments (a requirement for transmission in packet form, as is the case for multimedia over IP), the playing of the voice or video needs to be “smooth.” In other words, the time relationships between when the segments are played needs to match the time relationships between when the segments were recorded on the sending side. Therefore, variance in the time sequencing of consecutive segments of the voice or video stream (variance, that is, from the time sequencing on the recording side) can be little tolerated.

Furthermore, real-time traffic can be divided into *conversational* and *streaming* traffic [1]. Conversational traffic, such as voice or video telephony, involves an interactive exchange of streaming data. Humans want to perceive the voice and/or video arriving instantaneously or almost instantaneously in order to carry on a normal conversation. Therefore, the end-to-end delay from recording the voice and/or video on

1. Of course, an exception is the case that multimedia content is downloaded for later playback, an example being the transfer of files containing multimedia content over peer-to-peer file sharing networks, and another example being the using of FTP to transfer files, some of which may happen to have multimedia content. For the purposes of this book, however, we are more interested in the cases where real-time playback of multimedia is in effect.

one side to playing it on the other side(s) must be very low. (Studies have one-way delays of less than 150 ms as best, whereas up to even 400 ms may be tolerable in certain conditions, such as when people are physically across the globe and psychologically prepared for small delays [2].) On the other hand, some kinds of streaming real-time multimedia, such as streaming movies from a video server, do not have as tight delay constraints as conversational traffic. Since the traffic is mostly one way, some delay can be tolerated before the multimedia traffic starts playing. However, once it does begin, the usual constraints for real-time traffic apply. Since our focus in this book is on conversational traffic, we will not discuss protocols like Real-Time Streaming Protocol (RTSP), which are applicable for streaming multimedia. Note that RTSP makes use of Real-Time Protocol (RTP) for stream transport. We do discuss RTP in this book, but for RTSP you may refer to the RFC for more information [3].

In Chapter 2, we make a distinction between the Internet and other IP-based networks. Similarly, here we make a distinction between voice over the Internet and VoIP. The global Internet is an existing network of networks that will provide only best-effort packet delivery for the foreseeable future. As such, it cannot provide high-quality transport for VoIP services, much less video over IP services. VoIP, on the other hand, refers more generally to VoIP-based networks. This includes private IP-based networks that can be engineered with the desirable QoS because they are controlled by one organization or a small group of organizations that can agree on QoS controls. With voice over the Internet, each organization only has control over a small piece of the entire network. It is virtually impossible to get every organization using the Internet to agree on a desired QoS solution and to cooperate in implementing it.

## 4.1 Motivation

We first present a series of possible reasons for development of multimedia over IP and VoIP, and then discuss which of these are primary motivations, and which are secondary.

### 4.1.1 Efficient Digital Voice Coding

The public switched telephone network (PSTN) previously used only analog transmission of voice. However, the circuits between switches in the PSTN have by now mostly been replaced with digital circuits, where voice is encoded at 64 Kbps using pulse code modulation (PCM). Most lines to homes and offices are still analog lines, although some are also digital, such as integrated services digital network (ISDN) lines. Of course, 64 Kbps is not a magic number, and digital voice coding schemes have been created with other rates as well. For example, in GSM, a 13-Kbps voice coder is used. Interestingly, in GSM, despite the highly efficient 13-Kbps encoding used, there is a transcoder rate adaptation unit (TRAU) in the radio access network to convert between this 13/16-Kbps coding and regular 64-Kbps PCM, because the

circuits between the MSC and other switches use standard 64-Kbps PCM. Transporting multimedia over IP could allow the benefits of efficient voice coding schemes to be more fully realized.

#### 4.1.2 Support for Different Levels of Quality and Different Media

With the PSTN, there is no room for different levels of voice quality. It is not a trivial matter to add video telephony due to the design of the circuits that support 64-Kbps PCM voice signals. Higher-rate voice coding schemes could not be easily supported. Even if a better voice coding scheme to “fit” in the 64-Kbps circuits existed, it would cause either wasted bandwidth (it might not need all the 64 Kbps) or other problems in conversion between the A-law PCM used in some parts of the world and the  $\mu$ -law PCM used elsewhere. (With  $\mu$ -law PCM, switches may assume incorrectly that signals are encoded in a certain way, although they may not even use PCM.) Besides, the PSTN does not accommodate negotiating voice coders, in contrast to SIP in the VoIP case (as will be seen in Chapter 5).

#### 4.1.3 Network and Service Integration

In the past, voice telephony was the main form of network traffic. People might have used voice modems (for data, using the telephone lines for transmissions) to do data communications. Although not very efficient, that was acceptable in the past. As data networking, and in particular, IP networking, has grown, there is less and less need for the PSTN to access networks like the Internet, as various broadband alternatives have emerged. Looking ahead, the volume of data traffic will dwarf the volume of PSTN traffic.

Rather than maintaining parallel PSTN and data networks, might it make sense to integrate them? Given that IP packet-switching technology works very well for general data traffic, could we integrate more “connection-oriented” voice and video traffic over IP as well? With integration, tremendous cost savings could be enjoyed. This is the benefit of network integration. Meanwhile, it might be easier to provide integrated services like unified messaging, or Web and telephony integration, using IP rather than a separate network. Innovative integrated services could be created in such an environment. This is the benefit of service integration.

#### 4.1.4 Statistical Multiplexing

Consider the traditional phone system and the connection between a residential phone and the central office. Lines from several homes in a neighborhood may all be connected to a pole and groups of lines are spliced (consolidated) into larger and larger bundles moving towards the central office. This system is analogous to a river, where the river mouth corresponds to the connection to the central office and the sources of water correspond to the phones. The small streams consolidate into larger streams that flow into the main river as tributaries. One major difference is

that in the case of the phone system, not every phone is in use most of the time. At some stages of consolidation, especially closer to the central office, there is not enough capacity to handle all or most of the phones being used *simultaneously*. Hence, it is possible that if many phones are in use, and a call is to be delivered to, or originated from, yet another phone in the same region, that call cannot be completed. Such an uncompleted call is said to be *blocked*.

Clearly, blocked calls are annoying to their victims, so why is not more capacity provided? Suppose that the capacity needed to completely prevent blocking is called full-connection capacity. In this case, there is enough capacity to handle all the phones in the region in use simultaneously. Making some simple assumptions about the duration of phone calls and about the interval of idleness between phone calls, the capacity can be significantly decreased from the full-connection capacity, while maintaining a low probability of call blocking (e.g.,  $10^{-5}$ ). This is called statistical multiplexing, or the multiplexing of shared resources based on the statistical reasoning that blocking is rare. VoIP arguably provides much better statistical multiplexing than circuit-switched voice sessions, because for circuit-switched sessions, the circuits are tied up even when neither party is speaking, whereas with VoIP, packets are sent only when there is speech to encode and send.

#### 4.1.5 Assessment

Network and service integration is the primary motivation for investment in voice and multimedia over IP. The PSTN is very good for voice, but it is a stand-alone and monolithic network, and mostly a one-medium network as video telephony has not taken off. As the volume of data traffic eclipses and then dwarfs the volume of PSTN traffic, it makes much less sense to maintain two networks just for historical reasons. Integrating the networks will result in cost savings in the end. Also, service integration leads to all kinds of exciting possibilities. As the second important motivation behind network and service integration, I would rank support for different levels of quality and different media. This would fit in nicely with the new service possibilities with service integration. Some bandwidth efficiency may be achieved using more efficient voice coding schemes, as well as through statistical multiplexing, but these efficiencies would be less significant in the future.

Despite the real and perceived benefits of multimedia over IP, QoS and transport challenges remain. The fact remains that the circuit-switched PSTN provides good quality voice using dedicated circuits. With multimedia over IP, the best-effort nature of IP packet delivery is a real concern. If it turns out that acceptable-quality voice and video cannot be delivered over IP networks, then the potential benefits of this service do not matter. So we first look at the requirements in Section 4.2, and then discuss the issues and challenges in providing multimedia over IP in the light of these requirements (in Section 4.3). Next, we look at how some IP transport protocols might or might not meet the requirements (in Section 4.4).

## 4.2 Requirements

First, we consider the transport requirements for multimedia over IP streams and the coding requirements. Next, we consider what other network elements may be required for multimedia over IP. Finally, we examine signaling requirements.

### 4.2.1 QoS and Transport Requirements

As mentioned at the start of this chapter, two of the biggest challenges to providing sufficient QoS for multimedia over IP are related to delay: the end-to-end delay needs to be less than 400 ms, and the delay variance needs to be small as well. The delay variance is often called the jitter, referring to the small (hopefully!) fluctuations in arrival time of packets at the destination.

Unlike some other kinds of traffic (e.g., file transfer), voice and video traffic can survive occasional dropped packets. The encoding schemes are designed so that the occasional dropped packet results in a momentary slight degradation of quality and not catastrophic loss. For voice or video traffic, because they are real-time, retransmission of dropped or lost packets does not make sense, because by the time the retransmissions occurs, the arriving packet would be useless. (The stream would have moved on, so it would not make sense to play the few milliseconds of voice/video contained in the retransmitted packet.) This is quite different from the requirements for file transfer, for instance, where large end-to-end delay and arrival-time jitter can be tolerated as long as all packets eventually arrive at their destination.

Furthermore, for multimedia, a third requirement is that the streams can be synchronized at the receiver. Otherwise, the human at the receiver would be disconcerted by streams playing back in an out-of-sync manner. Therefore, the playback application must be able to synchronize packets from the different streams.

These requirements apply in both wired and wireless networks. However, they are generally harder to meet in wireless networks, where delay may be higher, and where higher error rates lead to higher rates of dropped packets, as we will discuss in Section 4.5.

### 4.2.2 Coding Requirements

As discussed earlier, even the PSTN has been switching over to digital encoding of voice, rather than using analog encoding, at least in the backbone of the network between switches. However, the PSTN is constrained to its 64-Kbps circuits, unless a massive redesign of the PSTN allows other rates. For multimedia over IP, there is no artificial 64-Kbps limitation. The question is whether significant bandwidth savings can be achieved for only modest loss of signal quality. The standard G.711, used in the PSTN, has a mean opinion score (MOS) of 4.3. The MOS is a scale from 1 (bad) to 5 (excellent) and is used to rate voice coders. We note that the G.729 coder, while scoring only 4.0 on the MOS test, uses only 8 Kbps of coding! The tradeoff is the added delay, 15 ms in the case of G.729 (it uses 10-ms frames and

includes a 5-ms look-ahead buffer). In general, the most bandwidth-efficient coding schemes (like G.729) will need to do frame encoding. Frames typically range from 10 to 30 ms.

### 4.2.3 Other Network Elements

In many cases, only the two end hosts need to understand VoIP protocols and functions; the intermediate nodes in the path of the VoIP traffic need not add special functions for this support. They may have capabilities that can be used for VoIP support, such as QoS capabilities, but these are not specifically for VoIP support alone, unlike the functions in some of the new network elements we will now introduce. This is a good example of the end-to-end design principle introduced in Chapter 1. However, there are usage cases where it is helpful for intermediate nodes to perform special functions specifically to support VoIP applications. Examples of such functions and network elements include mixers, translators, and PSTN gateways.

#### 4.2.3.1 Mixers

Imagine a conference call between two locations, in each of which two or more people speak to their colleagues at the other side through a speakerphone, and listen through the same speakerphone to what their colleagues say. Would you expect a separate connection to exist between each possible pairing of colleagues at the two locations? Of course not—the voices of everyone at one site are mixed and carried on one line to the other side, where the same thing happens in reverse.

Mixers help provide an analogous service in the case of VoIP telephony. Mixers combine two or more streams into one stream, where ideally the resultant stream is a superposition of the original streams. Mixing is a powerful concept with the flexibility to be used in a variety of scenarios. Unlike the speakerphone analogy, there need not be any notion of geographical proximity of the source streams. Like the speakerphone analogy, a mixer may be used to consume network resources more efficiently (than to transmit streams separately). This may not matter as much for high-bandwidth users, who can afford to receive several separate streams without a problem, but could be essential for low-bandwidth users, for example, users accessing the network through wireless links. If the low bandwidth is a severe problem, the encoding scheme of the output of the mixer could be chosen to optimize bandwidth usage. Mixers could also take the shape of other creative ideas, such as cleverly combining video scenes of different individual people to simulate a video scene of a group of people [1].

#### 4.2.3.2 Translators

One of the usage scenarios of a mixer involves low-bandwidth users. Streams from other machines can be combined (mixed) into one. Now, imagine again the same low-bandwidth users, wanting to transmit to other users in the conference. If they had to transmit separate streams to each of the other users, they would again run into bandwidth problems. A mixer cannot help in this case, since the issue is not

mixing multiple streams into one, but to transmit one stream and split it into multiple streams. However, a translator, placed on the other side of the low-bandwidth link, can help by taking one stream from such a low-bandwidth user and replicating it to transmit to multiple recipients. This translator usage is a kind of multicasting, in the sense of being an efficient way of transmitting from one to many. It is a kind of dual of the mixer going from many streams to one.

In general, though, a translator is not used solely for one-to-many translation. A translator can also take one input stream and produce one output stream, perhaps with a different encoding scheme. A useful rule of thumb to differentiate between mixers and translators is that given multiple input streams, the translator will process all of them separately, producing one or more output streams corresponding to each input stream, whereas the mixer would mix groups of two or more of the input streams to produce one output stream per group.

#### 4.2.3.3 Combining Mixers and Translators

In general, there may be cascades of one or more mixers, and one or more translators, between the communicating end hosts. Figure 4.1 shows an example of mixers and translators in action. Notice that the symbols used for mixers and for translators indicate their functions. The various streams (represented by arrows) are mixed together (come together) in the center, for mixers, and not for translators. In mixer A, three streams are mixed into one, whereas in mixer B, two streams are mixed into one. In translator C, there are two input streams, each of which is translated into an output stream (perhaps to change the encoding scheme or other parameters). In translator D, we see translation from a multicast address to unicast streams.

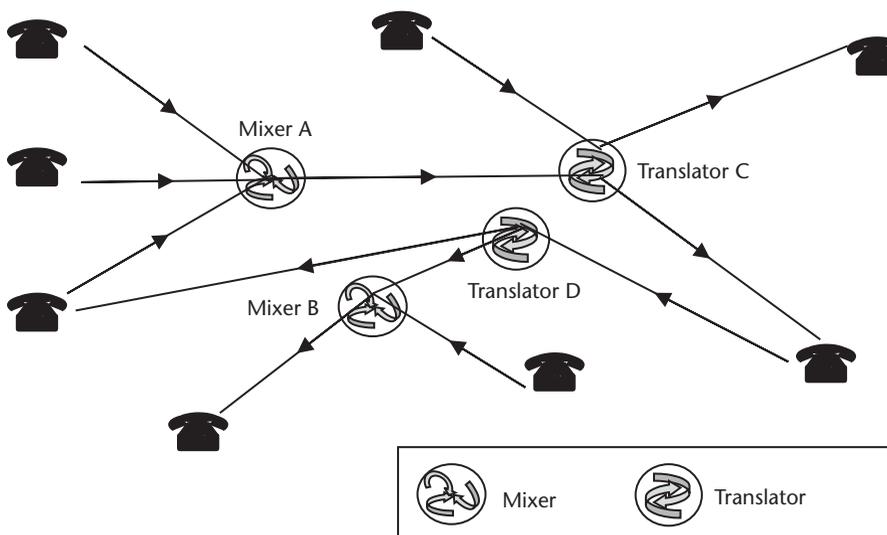


Figure 4.1 Illustration of mixers and translators in action.

### 4.2.3.4 PSTN Gateways

The traditional PSTN is going to be around for a long time, at least in some parts of the world. Many people will only be reachable through the PSTN, so being able to complete calls to the PSTN, or receive calls from it, is a very attractive system feature.<sup>2</sup> Various PSTN gateways have been designed for this purpose. The current leading standards-based approach is the media gateway approach, itself a consolidation of several proposals and currently standardized as H.248 (formerly H.GCP) by the ITU [4]. In this approach, two gateways are used for PSTN interworking: a *media gateway* for converting between signaling on both sides (e.g., between SIP and ISUP), as well as traffic encoding on both sides, and a *signaling gateway* for converting between signaling transport used on the IP side (e.g., SCTP) and signaling transport used on the PSTN side. The two gateways complement each other, one focusing on signaling and the other on signaling transport. The use of these two gateways, instead of just one integrated PSTN gateway, allows for more flexibility in deployment, and is a good example of modular design.

The media gateway itself is often split into a *media gateway controller* and a *media gateway*. The media gateway does the actual media conversion (e.g., between G.711 on the PSTN side and any of a number of possibilities on the IP side), while the media gateway controller controls the media gateway using H.248 signaling. Again, the split between media gateway controllers and media gateways is an example of modular system design. It allows each component to be upgraded separately, and each focuses on what it does and doing it well. Media gateways without control functions are cheaper, and multiple media gateways can be controlled by each media gateway controller. The arrangement of these network elements is shown in Figure 4.2. On the left side is the IP network, where the signaling traffic (e.g., SIP over UDP) and media streams (e.g., encoded with G.723) are split and go between the phone and media gateway controller, and between the phone and media

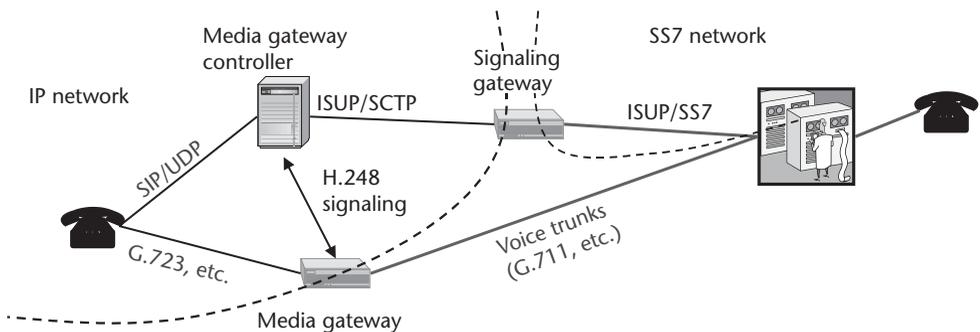


Figure 4.2 Interworking with the PSTN.

gateway, respectively. The signaling protocol conversion between SIP and ISUP signaling is done at the media gateway controller and the transport for the signaling changes from IP to the SS7 network, as it enters the SS7 network at the signaling gateway. Meanwhile, the media streams are translated (e.g., between G.723 and G.711), and changed from IP transport to placement on voice trunks in the PSTN, at the media gateway. H.248 signaling is used for control between the media gateway controller and the media gateway.

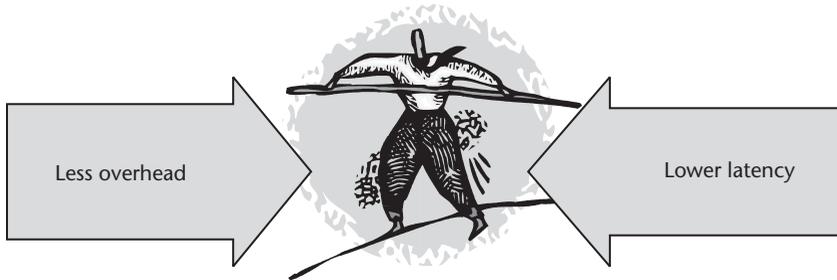
#### 4.2.4 Signaling Requirements

Signaling between the two parties is required so that agreement can be reached on various questions, such as whether a session is desired in the first place and what kind of traffic it involves (e.g., video or audio and with what coders and decoders). There is also signaling involving other network elements, such as gateways to the PSTN when one or both parties are PSTN phones. In the early days of VoIP, the ITU's H.323 was the prominent signaling protocol. However, recently the IETF's SIP has grown in popularity and is replacing H.323 as the dominant signaling protocol for controlling VoIP sessions. One reason for this phenomenon is that SIP is "lightweight," specializing in session control signaling and doing it well, in the spirit of the piecewise specification approach mentioned in Chapter 1. Meanwhile H.323 is a broader specification and more cumbersome than SIP. We will discuss SIP and signaling requirements in detail in Chapter 5.

### 4.3 Issues and Challenges

Most of the challenges related to multimedia over IP have to do with QoS. Since the Internet provides only best-effort service, meeting the QoS requirements for multimedia traffic is a big challenge. Techniques for engineering an IP network for QoS differentiation can help, as will be discussed further in Chapter 7. With careful design, the end-to-end delay could be kept below 400 ms (or 300 ms or 250 ms, depending on the quality desired). Jitter could be handled by some buffering at the receiver, at the expense of slight increases in end-to-end delay.

*High packetization overhead* is another challenge. Because VoIP requires low latency, the transmitter cannot afford the luxury of waiting for many bits to accumulate before sending them together as a large packet. In fact, voice is typically sent in individual frames that are encoded frame by frame to achieve high compression efficiency. Each frame is typically 10 ms or 20 ms of speech, with the actual number of bits per frame depending on the bit rate. As explained in Section 4.2.2, the frame size and look-ahead buffer size add to the overall delay ( $10 + 5 = 15$  ms for G.729, for example). Given the great savings in bandwidth usage, designers generally tolerate the delay for one frame. However, waiting for multiple frames to send together is a problem. Now, if we send frame by frame, the G.729 frame has only 80 bits, for example. The IP header is at least 20 bytes (160 bits) long, and it gets worse with IPv6 (40 bytes). Add to that the transport headers (UDP, RTP), and we see that the



**Figure 4.3** Is there an ideal packet size for VoIP?

bulk of the packet is header overhead. Figure 4.3 illustrates the dilemma: On the one hand a heavy pressure is exerted to reduce the end-to-end delay by reducing the packet size. On the other hand, a heavy pressure push is exerted to increase the packet size to reduce the high overheads.

One approach to solving the packetization overhead problem is *RTP multiplexing*. RTP is a transport protocol optimized for real-time traffic [5, 6] (see Section 4.4.3). RTP multiplexing refers to sharing an RTP header between multiple sessions that are transmitted in the same packet.

Another approach to solving the packetization overhead problem is RTP header compression. RTP header compression can compress the IP/UDP/RTP header down to two to four bytes, from 40 bytes [7]. Its usefulness is limited by the fact that it works only for the single-link, unicast case. Otherwise, more header information is needed. Nevertheless, this is useful over a wireless link, where bandwidth is often scarce. In wireless links where packet loss is a problem, and where very high compression efficiency is required (because of limited bandwidth, for example), robust header compression may be used as an alternative to RTP header compression [8]. More recently, an enhancement to RFC header compression, Enhanced compressed RTP, has been developed, which is more robust for links where the delay is high and where packet loss and reordering may be encountered [9].

Additional issues and challenges related to multimedia over IP in wireless networks will be discussed in Section 4.5.

## 4.4 Transport Protocols

We briefly examine the suitability of traditional transport protocols like TCP and UDP for real-time multimedia applications, and then discuss at some length a transport protocol, RTP, designed with such applications in mind.

### 4.4.1 TCP

TCP is the main transport protocol used in IP networks. However, TCP was designed for traditional data network applications, like file transfer, with different transport requirements from multimedia applications, as explained in Section

4.2.1. For example, TCP uses ACKs so that it can retransmit packets not received by the receiver. However, for real-time multimedia traffic, this is not useful, since the application would have moved on to play the most recently received packets by the time a retransmitted packet arrives, and so would not have use for retransmitted packets.

#### 4.4.2 UDP

Whereas TCP is a complex protocol with several mechanisms to support traditional applications like file transfer, UDP is a more lightweight protocol that is more suitable for real-time multimedia applications. For example, UDP neither detects unreceived packets, nor arranges for them to be retransmitted. However, while not retransmitting unreceived packets is good for real-time multimedia applications, it would be helpful to detect them, as well as provide some time-stamping capabilities. So UDP could be used, if it could be augmented in some ways.

#### 4.4.3 RTP

RTP was designed from the ground up to carry real-time traffic. Therefore, it has features that help in providing the QoS needed for multimedia over IP. These features include time stamping, the ability to synchronize different streams, and the ability to support a mixer. The companion protocol, Real-time Control Protocol (RTCP), is used with RTP to distribute reports on the performance of the real-time data transport to the participants in the conference.

RTP is typically run over UDP (i.e., RTP packets, including the RTP headers, are put into UDP packets as the payloads), where RTP uses even port numbers, and the corresponding RTCP packets use odd port numbers that are one higher than the port numbers used by RTP. We will first explain how RTP works and then describe RTCP. RTP over UDP can be thought of as augmenting the UDP header information with additional information useful for real-time traffic, and placing these additional bits of information in the front portion of the UDP payload. What are these bits of information, and how do they help? We examine the RTP header, shown in Figure 4.4 and highlight the fields most pertinent to facilitating real-time traffic. In particular, we will discuss:

- Time stamp;
- Sequence number;
- Synchronization source (SSRC) identifier;
- CSRC identifiers and CSRC count.

We first note that two notions of packet sequencing are needed. In order to facilitate synchronization of different media, and to facilitate jitter computations (for jitter monitoring by RTCP and possible correction by the application), highly accurate time stamping is needed. This is analogous to a postal worker stamping the date on the stamp on an envelope, where the recipient of the letter will know when

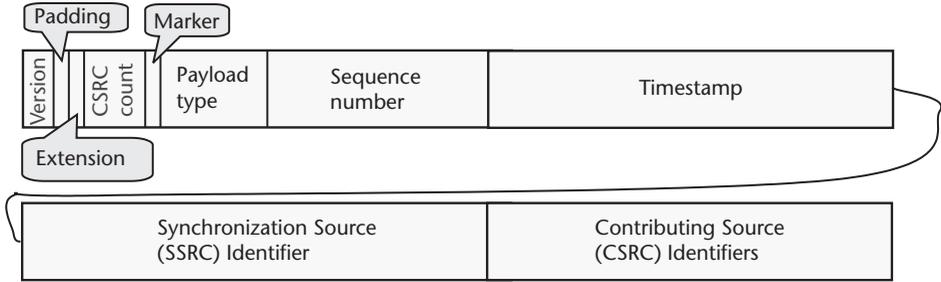


Figure 4.4 RTP header.

the letter was sent. In the case of RTP time stamping, the time resolution is of course finer than in the postal analogy, and the value is contained in the *timestamp* field (an example of the use of the time stamp will be shown shortly). The second notion of packet sequencing arises from the observation that it would not necessarily be trivial for the receiver to figure out if any packets have been lost, by just looking at the time stamps of the received packets. Hence, a *sequence number* is also used, where the sequence number is incremented by one each time an RTP packet is sent. If there is a break in the consecutive sequence numbers received, a receiver can detect that packets have been lost. The sequence number also allows a receiver to correctly reorder packets that have arrived out of sequence.

What if there are multiple streams from different sources? How would the receiver tell them apart? The SSRC field is used to allow differentiation of packets associated with different synchronization sources, where a synchronization source may be, for example, a microphone or camera. Sequence numbers and time stamps have relevance for packets from the same synchronization source. In other words, packets from the same synchronization source need to be played back in sync, and the SSRC together with the time stamps and sequence numbers allows an application to do so. There may be multiple streams, each with their own SSRC identifier, in one RTP session, an example being multiple video cameras, each producing a stream. Furthermore, there can be one or more RTP sessions in a general multimedia session.

The CSRC list is useful only when the stream has gone through a mixer. The CSRC list is the list of contributing sources, in other words, the SSRCs of the streams that went into the mixer. The SSRC of the packets coming out of the mixer is the SSRC of the mixer. Hence, the CSRC list allows receivers to identify the contributing sources despite the streams having been mixed. An application could then display the list of speakers.

Why does not RTP do more to guarantee timely delivery? Since it is a transport protocol and not a lower-layer protocol, the uses of RTP are limited, as with any other transport protocol. RTP does provide tools (such as time stamps) for an application to play media streams properly synchronized with segments in the right sequence and time order. Further tools for applications to monitor overall performance are provided by the companion protocol, RTCP. Table 4.1 summarizes what RTP does and does not do.

**Table 4.1** Summary of Capabilities of RTP

<i>What RTP Does Do</i>	<i>What RTP Does Not Do</i>
End-to-end transport	Guarantee timely delivery
Provides tools like time stamping, sequence numbers, SSRC that multimedia applications can use	Deal with resource reservation

#### 4.4.3.1 RTCP

RTCP distributes reports on the performance of the real-time data transport provided by RTP. The reports from receivers can be used by senders to dynamically adjust their encoders, or for flow and congestion control. Also, a network service provider might set up a node to receive RTCP packets to monitor the network for network problems. Such a node, or any other node that receives the various RTCP feedback reports, could assess the scale of problems. For example, if the quality of real-time data transport received is very bad only at a couple of nodes clustered together, but not at other nodes, then the problem may be suspected to be local. However, if all nodes report roughly the same performance problems, a global problem is more likely.

RTCP provides for both sender reports (SRs) and receiver reports (RRs). SRs contain statistics for *both* transmission and reception. In that case, why are RRs necessary? RRs are used by nodes that are not active senders. The concept of active sender in this context applies only to the time interval since a node issued its last report. If a node has sent RTP packets since then, it issues an SR. Otherwise the node issues an RR. The other use of RRs is in the case that an active sender is reporting on packets received from more than 31 sources. The first 31 packets will be reported in the SR and the rest in one or more additional RRs sent with the SR. The main difference between the SR and RR packets is that the SR includes a 20-byte sender information section. This sender information section details the packet count transmitted by the sender (a running count for as long as the same SSRC identifier is used). An NTP time stamp is included too, so that when other receivers send their reports, round-trip propagation can be estimated.

What information do the RRs contain, and how might they be useful? The reports are organized by SSRCs. For each SSRC, fraction lost, cumulative number of packets lost, highest sequence number received, interarrival jitter, the last SR time stamp received, and the time since that last SR, are sent. The fraction lost is an estimate of the proportion of packets lost to received packets, during the most recent reporting period, whereas the cumulative number of packets lost is an absolute number. A jitter estimate can be computed, as specified in the RTP specification [5]. The SR time-stamp information will assist the sender in computing round-trip propagation time. Note that the fraction lost and the jitter estimate provide estimates of network congestion that could be useful to the application.

## 4.5 Wireless Multimedia over IP

If one or more of the links in the communication path are wireless links, additional challenges arise. Wireless links have lower bandwidths than wired links, making schemes for compression of RTP headers more necessary in the wireless case. Wireless links generally have higher error rates than wired links. The “raw” error rates (without error control coding) can be quite high, and even after the application of forward error correction (FEC) codes, the error rates are still higher than in wired links. The high error rates have implications on multimedia over IP in several ways. First, header compression is very important for multimedia over IP applications, as explained earlier, because of the short packets and large header overheads of uncompressed headers. However, compressed headers are very sensitive to errors, so special header compression algorithms should be used. Second, certain voice codecs perform very poorly, and are therefore less suitable for use, in high-error environments like wireless links. Special, more robust algorithms like 3G-324M should be used over wireless links.

While we will discuss QoS in greater detail in Chapter 7, we note here that wireless links can add significant amounts of additional delay to the end-to-end delay. For delay-sensitive applications like conversational voice and video, there is already a challenge providing satisfactory delay characteristics when the entire network path is wireline. With wireless links in the path, the challenge is increased. Another phenomenon in mobile wireless networks is handoffs, as mobile nodes move from one connection point to another. Handoffs typically result in additional latency, even if care is taken to avoid packet loss through buffering schemes. Schemes that buffer packets and retransmit them after the handoff is complete may work well for regular data traffic, but the latency is a more serious problem for multimedia over IP. We will see some schemes for reducing handoff latency later in this book.

Another issue is battery power consumption. Wireless devices, being small and mobile, rely on batteries for power, so it is desirable to minimize power consumption, and thus to maximize the length of time the device can operate before needing to be recharged. Traditional cellular phones come with sophisticated power-saving mechanisms. For example, as we saw in Chapter 3, location update and paging concepts work together to effect power savings. GSM also has a variety of other mechanisms for power conservation. GPRS similarly has power-saving mechanisms, including the use of three mobility management states (discussed in Chapter 11). However, 802.11 WLAN does not have the same range of power-saving mechanisms, because it is a link-layer protocol. Thus, it cannot take advantages of mechanisms, such as the state-dependent location update frequency in GSM, that make use of cross-layer interactions. The standard 802.11 has no understanding of the traffic it carries, whereas GSM can associate the application-level concept of being in a call or not being in a call to the frequency of location updates as performed in the network layer. Therefore, in the case of voice over WLAN, vendors must implement their own non-standard power conservation schemes. In fact, vendors are doing so, although one could argue that a standardized solution might have been preferable [10].

## 4.6 Summary

This chapter and the next focus on technologies that directly support conversational voice and video applications, which are expected to play an important role in the future wireless Internet. This chapter deals with transport aspects and RTP in particular. After discussing the motivations for multimedia over IP, we surveyed the requirements (e.g., QoS and transport requirements and coding requirements) and discussed various network elements helpful for delivery of multimedia over IP traffic, such as mixers, translators, and PSTN gateways. We discussed issues and challenges for transporting multimedia over IP and over wireless links in particular. The basics of RTP and RTCP were explained.

## References

- [1] 3GPP TS 23.107, “Quality of Service (QoS) Concept and Architecture (Release 5),” V5.9.0, June 2003.
- [2] Hassan, M., A. Nayandoro, and M. Atiquzzaman, “Internet Telephony: Services, Technical Challenges, and Products,” *IEEE Communications Magazine*, April 2000.
- [3] Schulzrinne, H., A. Rao, and R. Lanphier, “Real Time Streaming Protocol (RTSP),” RFC 2326, April 1998.
- [4] ITU Recommendation H.248, “Gateway Control Protocol,” 2002.
- [5] Schulzrinne, H., et al., “RTP: A Transport Protocol for Real-Time Applications,” IETF RFC 3550, July 2003.
- [6] Schulzrinne, H., and S. Casner, “RTP Profiles for Audio and Video Conferences with Minimal Control,” IETF RFC 3551, July 2003.
- [7] Casner, S., and V. Jacobson, “Compressing IP/UDP/RTP Headers for Low-Speed Serial Links,” IETF RFC 2508, February 1999.
- [8] Bormann, C., et al., “Robust Header Compression (ROHC): Framework and Four Profiles: RTP, UDP, ESP, and Uncompressed,” IETF RFC 3095, July 2001.
- [9] Koren, T., et al., “Enhanced Compressed RTP (CRTP) for Links with High Delay, Packet Loss and Reordering,” IETF RFC 3545, July 2003.
- [10] Wexler, J., “Preserving Wireless VoIP Handset Battery Life,” <http://www.nwfu-sion.com/newsletters/wireless/2004/0419wireless1.html>, 2004.

