# Interactive Applications

- Structure of video conferencing systems
- Protocols for video conferencing
- Media transport

# Interactive Conferencing Applications

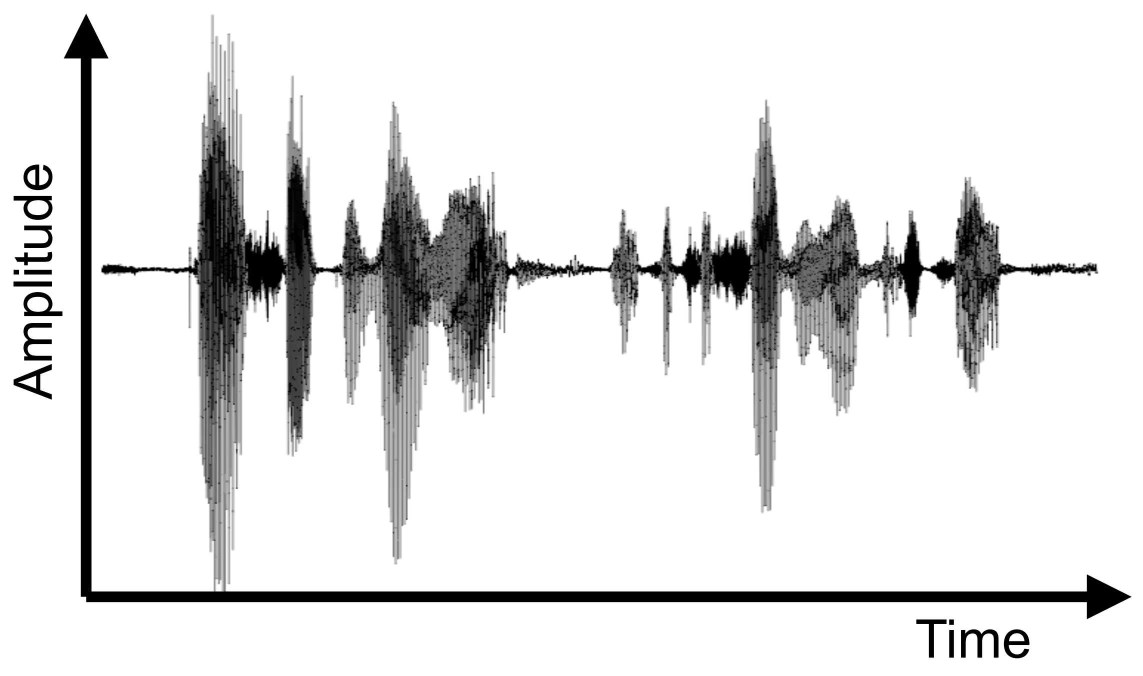
- What are interactive applications?
  - Telephony
  - Voice-over-IP (VoIP)
  - Video conferencing
- Long history of research and standardisation:
  - Network Voice Protocol (NVP)
    - Packet voice experiments over the ARPAnet
    - RFC 741 published in 1976
  - Modern standards development from mid-1990s
    - Mbone conferencing tools
    - SIP, SDP, and RTP protocols
    - Adopted by 3GPP as basis of mobile telephone standards
  - Browser-based conferencing from mid-2010s
    - WebRTC

Multi-party video conferencing over IP, 1998


CU-SeeME (source: Wikipedia)

# Requirements on Timing and Data Rate

- Interactive conferencing has tight latency bounds
  - One-way mouth-to-ear delay ~150ms maximum for telephony
  - Video conferences want to lip-sync audio and video
    - Audio should be no more than 15ms ahead, or 45ms behind, video
  - User experience degrades gracefully
- Data rates depend on media type and encoding options
  - Speech coding typically operates on 20ms packets
    - Captures one frame of speech every 20ms, encodes, transmits
      - Data rate ~tens of kilobits per second
      - Background noise during quiet periods encoded at lower quality, packets sent less often
  - Video frame rate and resolution highly variable
    - High definition video encoding using H.264 is around 2-4Mbps
    - Frame rates between 25 and 60fps commonly used
    - I-frames → tens of packets; P-frames → single or few packets



# Reliability Requirements

- Speech data is highly loss tolerant
  - Loss concealment can hide 10-20% random packet loss without noticeable loss in quality
  - Burst losses are less well concealed
- Video packet loss hard to conceal
  - No scene changes to reset decoder state to known good value
  - Retransmissions possible in some cases; forward error correction more typical

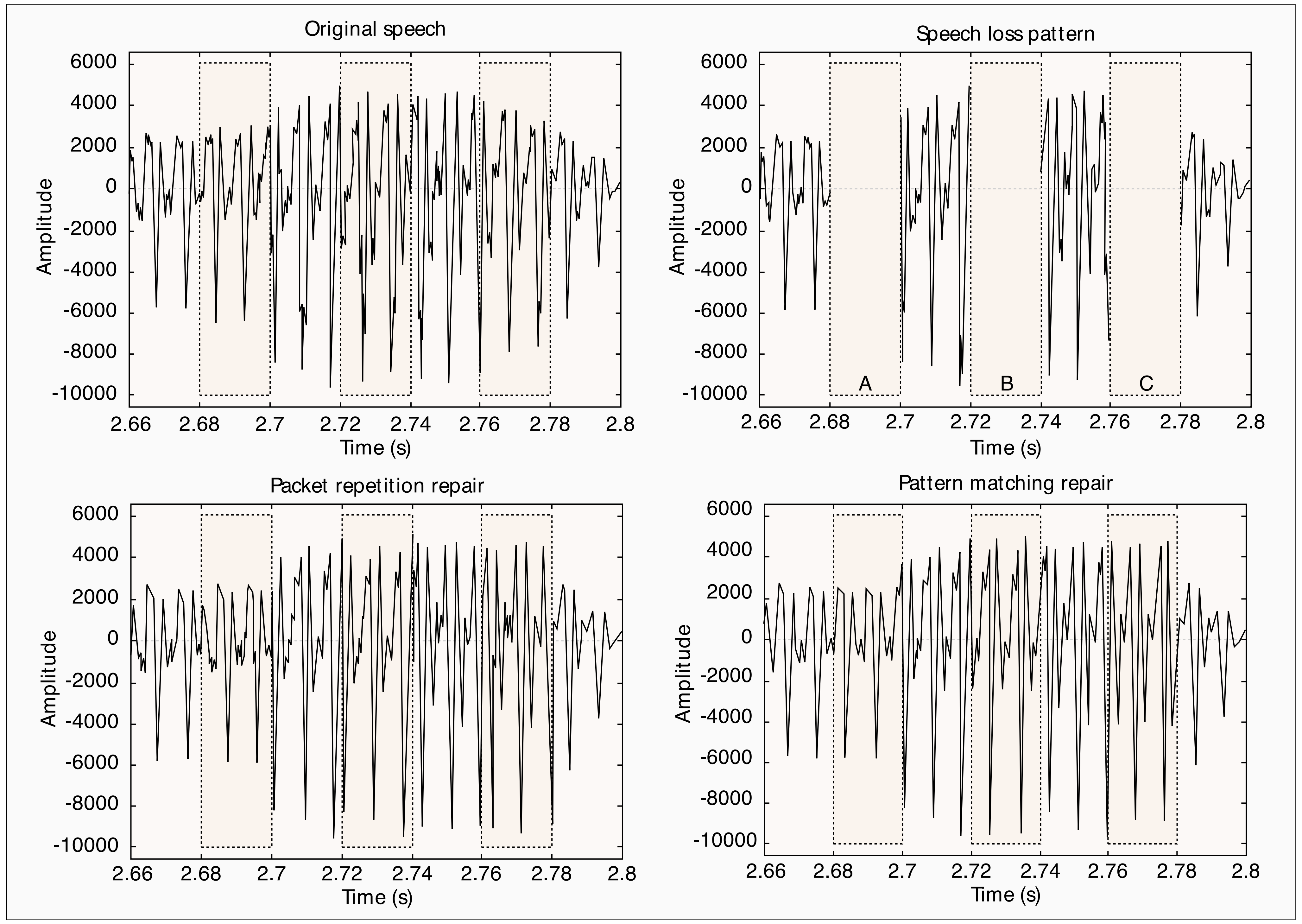
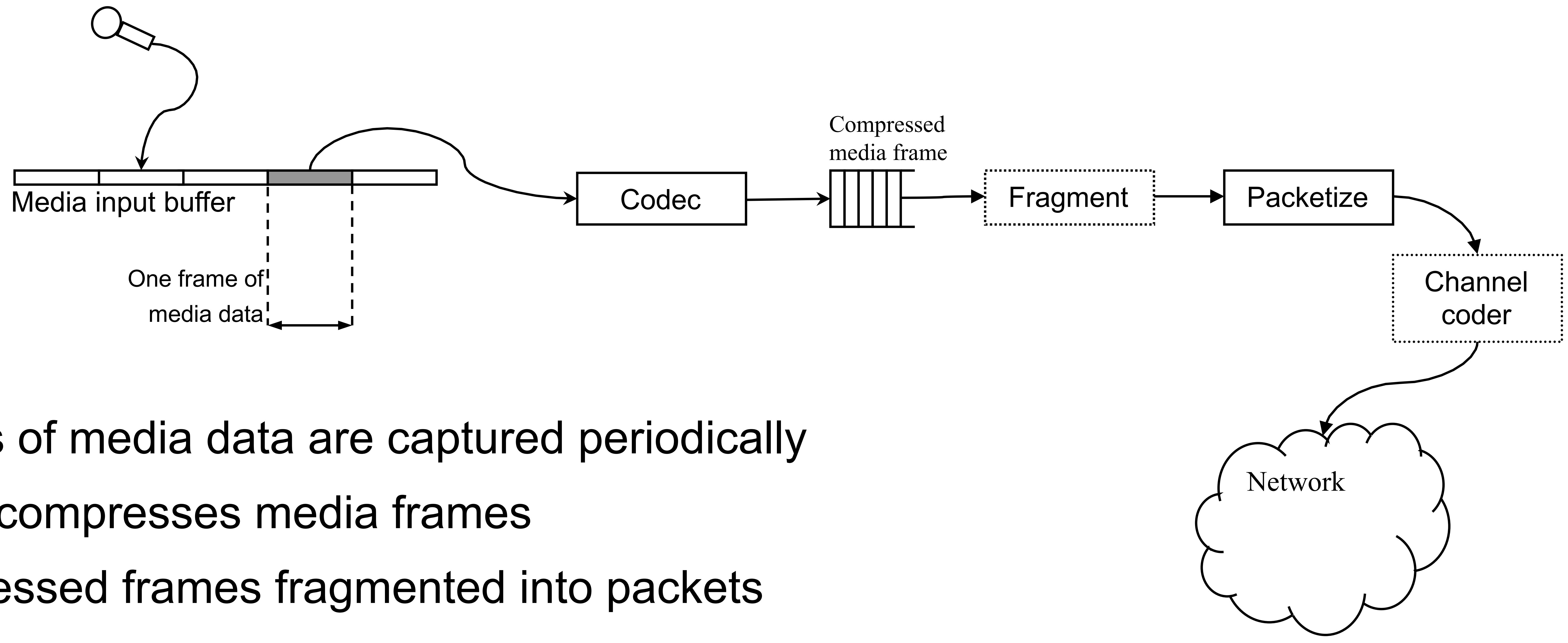



**Figure 9.** *(a) Sample error concealment techniques: original audio signal; (b) sample error concealment techniques: the loss pattern; (c) sample error concealment techniques; packet repetition; (d) sample error concealment techniques: one sided waveform substitution.*

C. S. Perkins, O. Hodson, and V. Hardman, A Survey of Packet Loss Recovery Techniques for Streaming Media, IEEE Network Magazine, September 1998. DOI:10.1109/65.730750

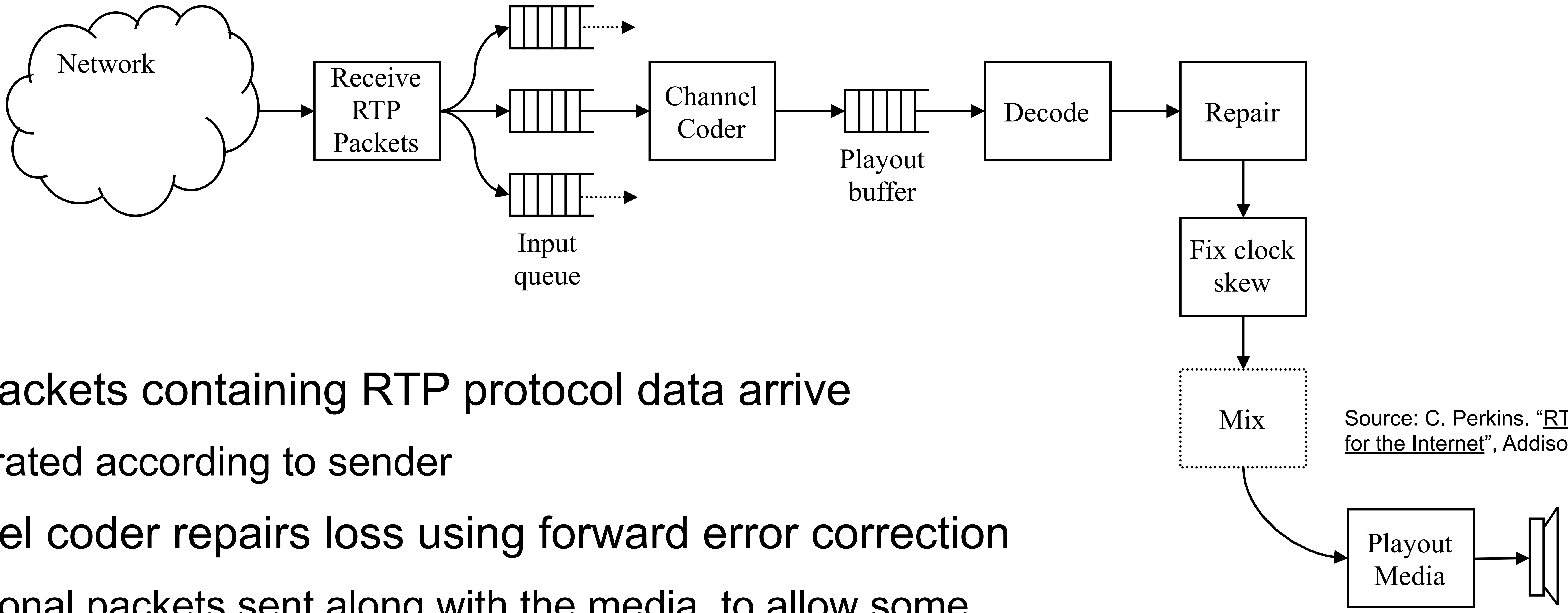
# Interactive Applications: Media Transmission Path




- Frames of media data are captured periodically
- Codec compresses media frames
- Compressed frames fragmented into packets
  - Transmitted using RTP inside UDP packets
  - RTP protocol adds timing and sequencing, source identification, payload identification
- Transmitted over the network

Source: C. Perkins. "RTP: Audio and Video for the Internet", Addison-Wesley, 2003

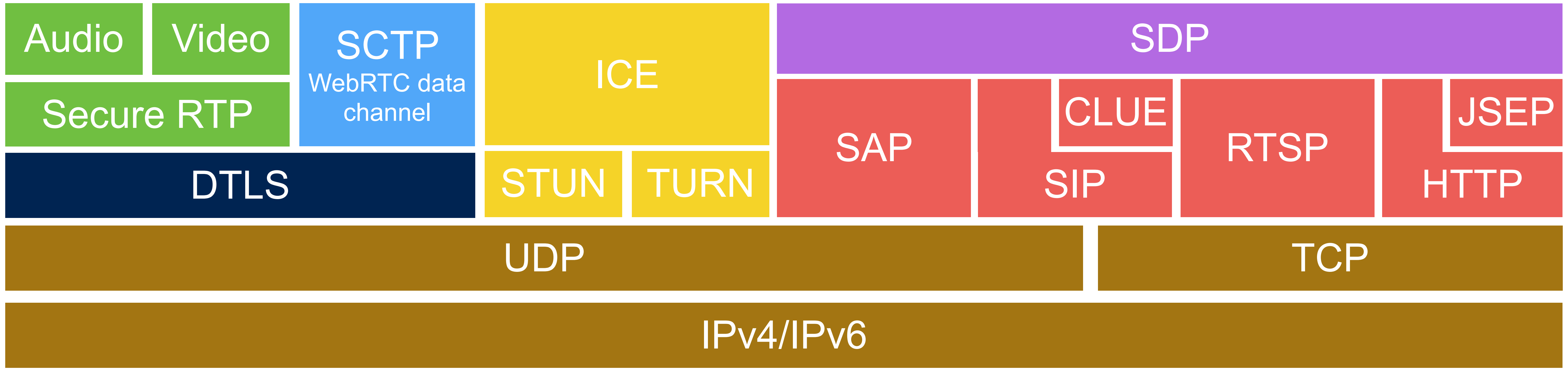
# Interactive Applications: Media Reception Path




Source: C. Perkins. "RTP: Audio and Video for the Internet", Addison-Wesley, 2003

- UDP packets containing RTP protocol data arrive
  - Separated according to sender
- Channel coder repairs loss using forward error correction
  - Additional packets sent along with the media, to allow some repair without needed retransmission
- Playout buffer used to reconstruct order, smooth timing
- Media is decompressed, packet loss concealed, and clock skew corrected
- Recovered media is rendered to user

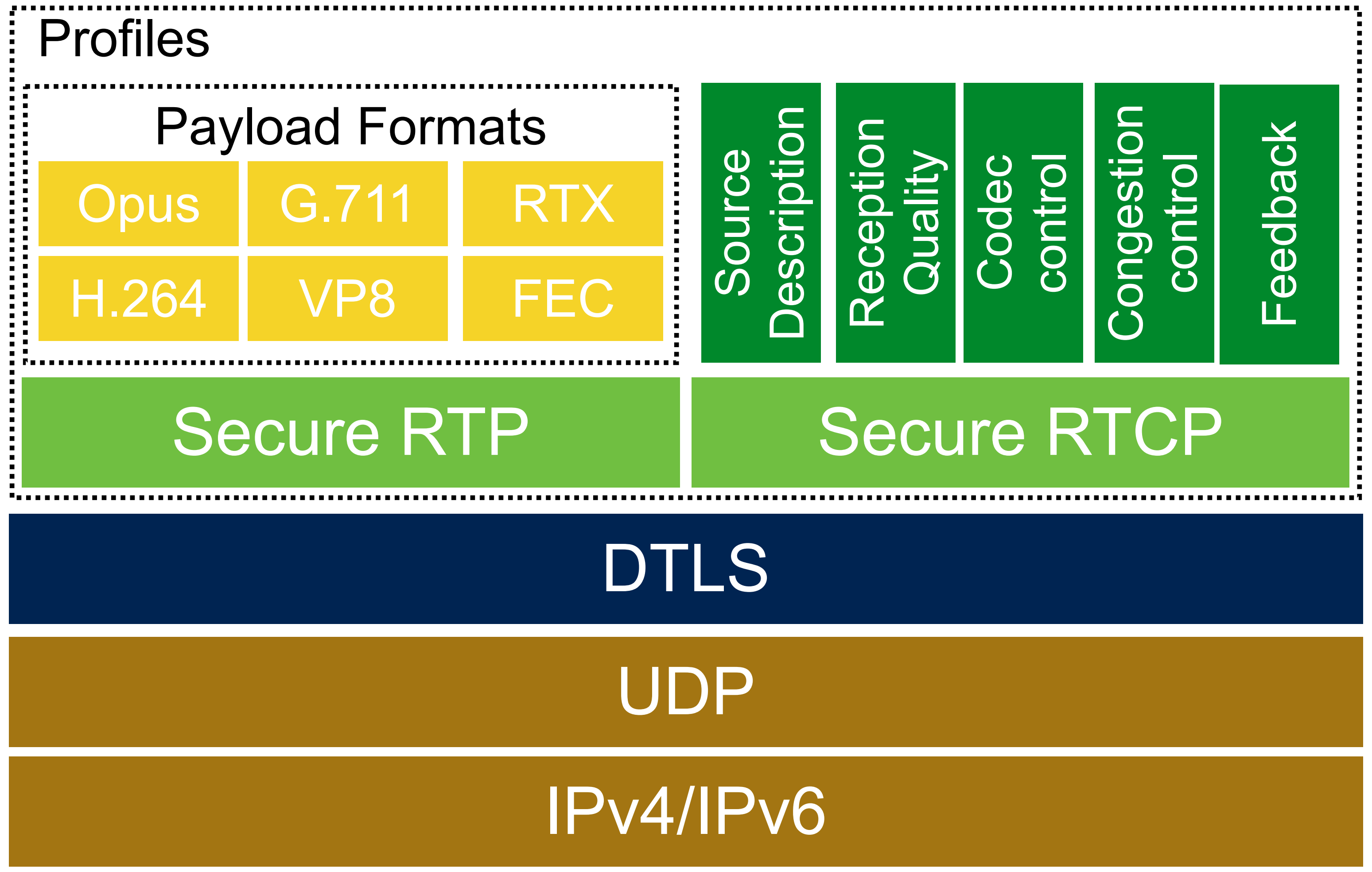
# Internet Multimedia Standards




- Media: secure RTP, WebRTC data channel
- Path discovery and NAT traversal: ICE, STUN, TURN → Lecture 2
- Session descriptions: SDP
- Signalling protocols for different purposes
  - Announcing multicast sessions: SAP – *obsolete*
  - Control of streaming media: RTSP
  - Control of interactive conferencing: SIP
  - Control of telepresence: CLUE – *not widely used*
  - Control of web-based interactive media: JSEP (WebRTC)

# Media Transport: RTP

- Separate data and control channels
  - RTP – media payload formats
  - RTP Control Protocol (RTCP)
    - Source description and caller identity, reception quality, codec control
- Payload formats
  - Codec-specific packet formats; application level framing; robust, but complex
  - Each frame packetised for independent use for low latency
- Datagram TLS handshake – usual TLS handshake but within UDP packets
- Extensions
  - Reception quality and user experience monitoring
  - Codec control and other feedback
  - Circuit breakers and congestion control



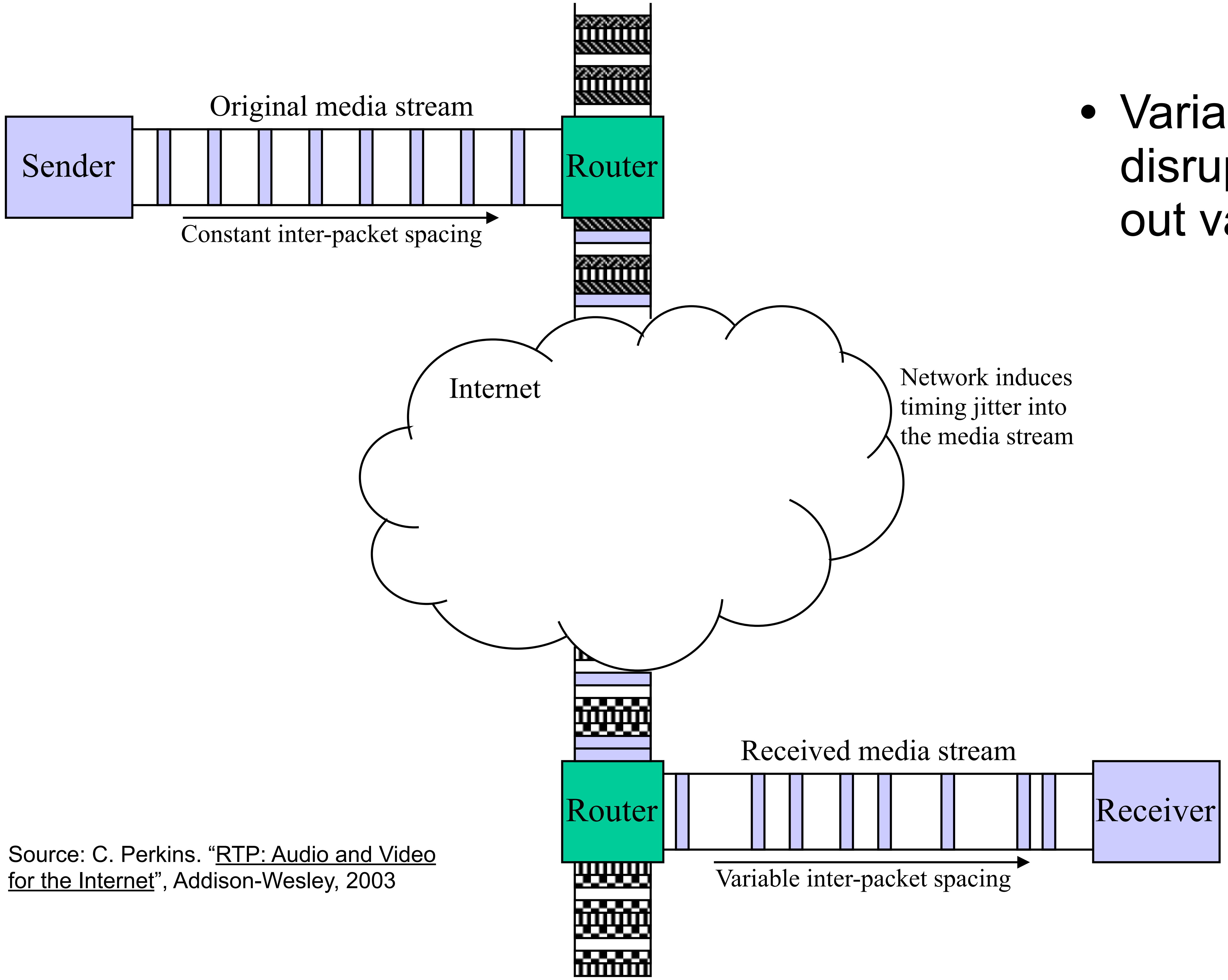
# Media Transport: RTP

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|V=2|P|X|  CC   |M|     PT      |       sequence number         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                           timestamp                           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|           synchronisation source (SSRC) identifier            |
+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
|                     [CSRC identifier list]                    |
|                        (4 * CC octets)                        |
|                        CC may be zero                         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ -+
|     defined by signalling     |    header extension length    |  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+  |
|                       header extension                        |  | OPTIONAL
|                  format defined by signalling                 |  | (if X=1)
|                                                               |  |
|                                                               |  |
|                                                               |  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ -+
|                            Payload                            |
|           (variable format and length, depends on PT)         |
|                                                               |
|                                                               |
|                                                               |
|               +-------........---------------+---------------+
|               |Padding (PadCnt octets, if P=1)|PadCnt (if P=1)|
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

https://datatracker.ietf.org/doc/rfc3550/

- RTP data packets carried within UDP
- Header information carries:
  - Sequence number and timestamp
    - Allows receiver to reconstruct ordering and timing
  - Source identifiers
    - Who sent this packet – needed for multiparty calls
  - Payload format identifier
    - Does the packet contain audio or video?
    - What compression algorithm is used?

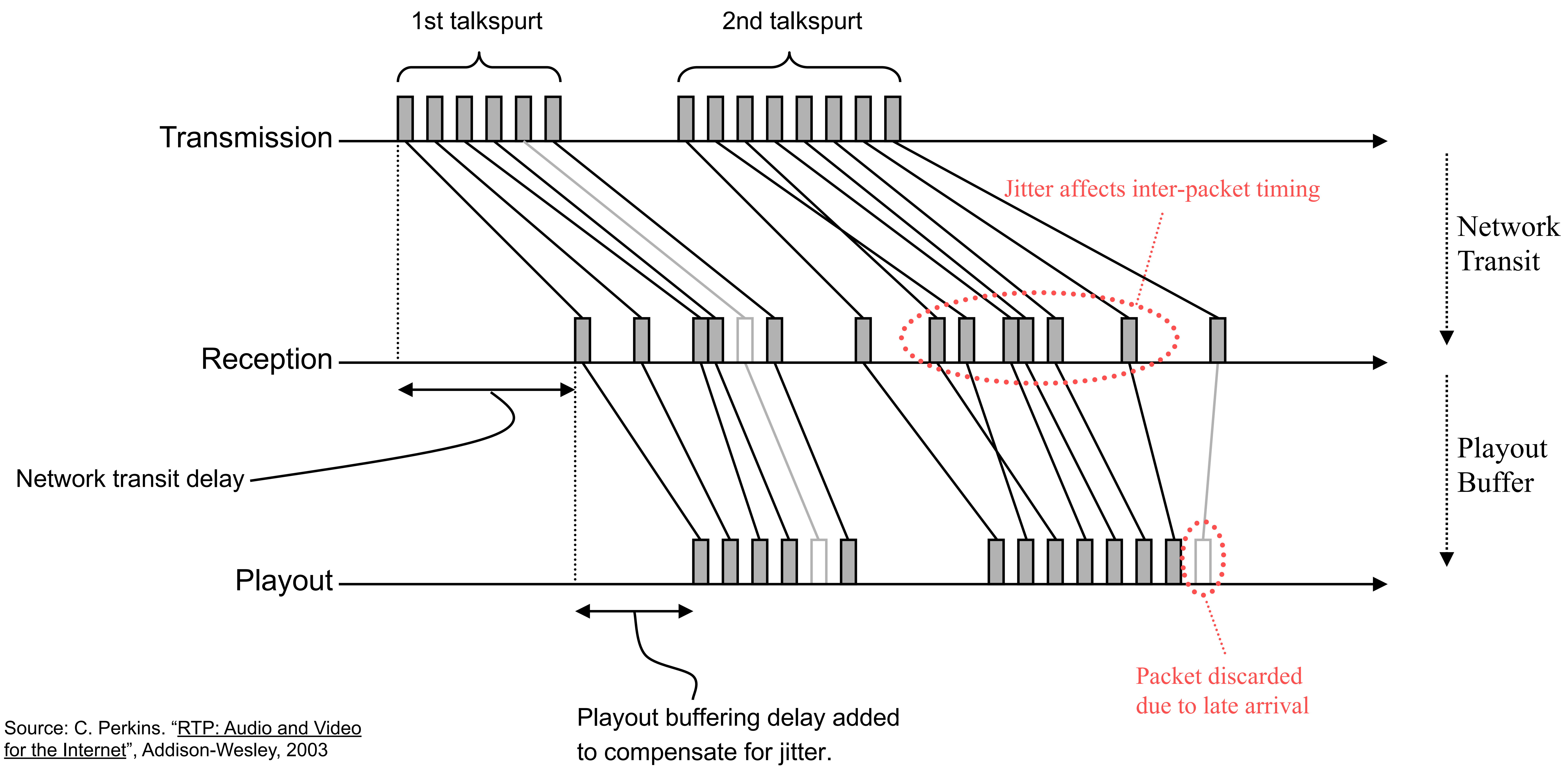
# Media Transport: Timing Recovery (1/3)




Source: C. Perkins. "RTP: Audio and Video for the Internet", Addison-Wesley, 2003

- Variable queueing delays in the network disrupt timing – receiver buffers to smooth out variation

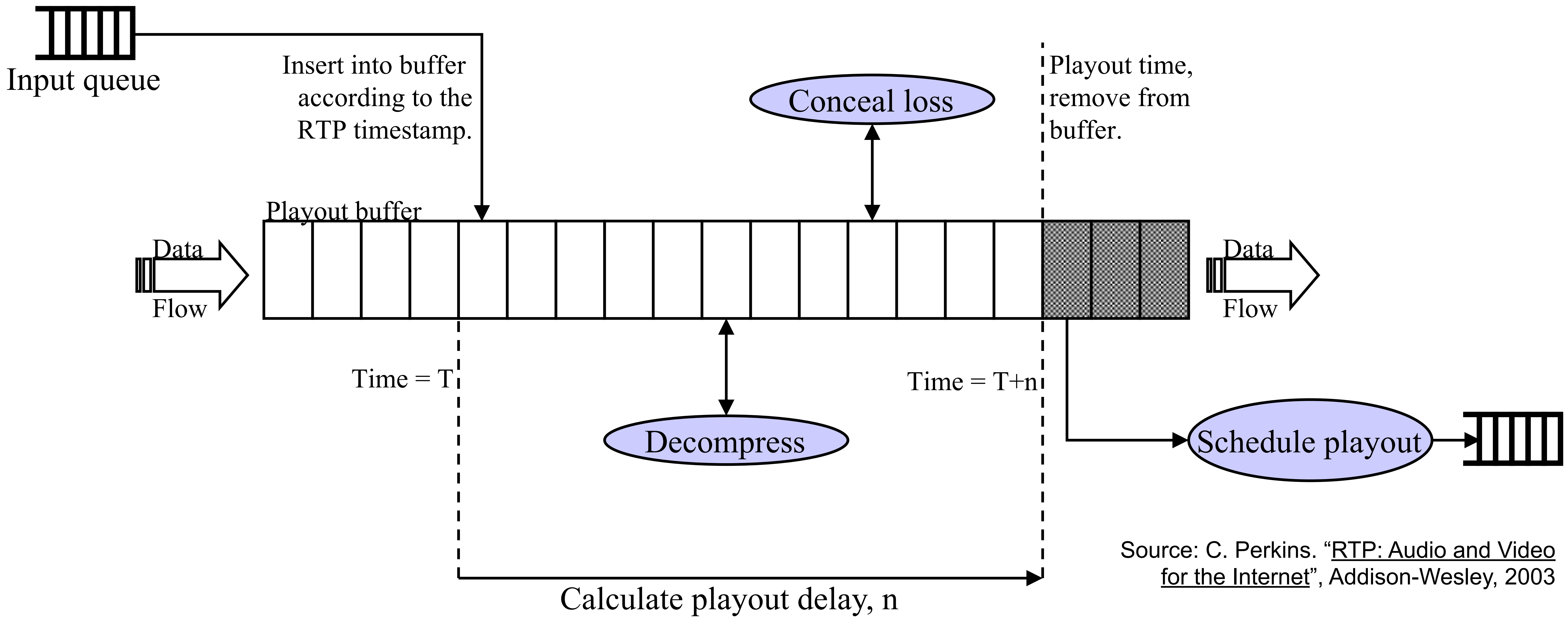
# Media Transport: Timing Recovery (2/3)




Source: C. Perkins. "RTP: Audio and Video for the Internet", Addison-Wesley, 2003

# Media Transport: Timing Recovery (3/3)

- If packets played out immediately on arrival, variation in timing leads to gaps
- Delay playout by more than typical variation in inter-arrival time, to allow smooth play back




Source: C. Perkins. "RTP: Audio and Video for the Internet", Addison-Wesley, 2003

# Media Transport: Application Level Framing

- Packet loss is possible, so receivers must make best use of packets that do arrive
- RTP **payload formats** define how compressed audio/visual data is formatted into RTP packets
  - Goal: **each packet should be independently usable**
  - If a packet arrives, it should be possible to decode all the data it contains – not always possible, but desirable
  - Naïve packetisation can lead to inter-packet dependencies where a packet arrives but can't be decoded because some previous packet, on which it depends, was lost

Architectural Considerations for a New Generation of Protocols

David D. Clark and David L. Tennenhouse
*Laboratory for Computer Science, M. I. T.*

**Abstract**

The current generation of protocol architectures, such as TCP/IP or the ISO suite, seem successful at meeting the demands of todays networks. However, a number of new requirements have been proposed for the networks of tomorrow, and some innovation in protocol structuring may be necessary. In this paper, we review some key requirements for tomorrow's networks, and propose some architectural principles to structure a new generation of protocols. In particular, this paper identifies two new design principles, Application Level Framing and Integrated Layer Processing. Additionally, it identifies the presentation layer as a key aspect of overall protocol performance.

**1 Introduction**

Historically, there has been a sharp distinction between the network architectures used in universal access environments, such as the telephone network, and the architectures developed within the computer communications community. We believe that both communities would benefit from a unified architecture that addresses new communication requirements. We have identified three key requirements that will stress the existing protocol architectures. Future networks will have considerably greater capacity, will be based on a wider selection of technologies, and will support a broader range of services.

The principle design goal for many research projects, and some commercial products, is gigabit operation, both in aggregate over trunks and to individual end points. As networks proceed to higher speeds, there is some concern that existing protocols will represent a bottleneck, and various alternatives have been proposed, such as outboard protocol processors [1, 11] and new protocol designs [2, 14, 12]. Since there is no clear consensus on the real source of protocol overhead, it is hard to validate these various solutions.

Another requirement is that the protocols of tomorrow operate over the range of coming network technology. An obvious example of emerging technology is Broadband ISDN [19, 6] which is based on Asynchronous Transfer Mode (ATM). This network technology provides a switching mode somewhat different from classic packet switching in that the data to be transferred is divided up into small (currently 48 byte) fixed size elements for transmission. A more futuristic network technology is wavelength division over fiber.

Finally, there is the requirement for service integration. In the future, a single end system will be expected to support applications that orchestrate a wide range of media (voice, video, graphics, text) and access patterns (interactive, bulk transfer, real-time rendering). This cross-product of media and access will generate new traffic patterns with performance considerations, such as delay and jitter tolerance, that are not addressed by present architectures.

In summary, future networks must exhibit a significant degree of flexibility and be based on an architecture that admits a wide range of application demands and implementation strategies. In the remainder of this paper, we identify some key principles for the elaboration of such an architecture.

**2 Structuring Principles for Protocol Architectures**

Most discussions of protocol design refer to the ISO reference model [9], or some other model that uses layering to decompose the protocol into functional modules. There is, however, a peril in using only a layered model to structure protocols into components, which is that layering may not be the most effective modularity for implementation.

It is important to distinguish between the architecture of a protocol suite and the engineering of a specific end system or relay node. The architecture specifies the decomposition into functional modules, the semantics of the individual modules and, the syntax used to effect the protocol. There should be no *a priori* requirement that the decomposition of the engineering design of a given system correspond to the architectural decomposition of the protocol. In the case of layered architectures, practical experience [4, 17] provides strong support for alternative engineering designs. Unfortunately, the structure of the protocol architecture may unnec-

*This research was supported by the Defense Advanced Research Projects Agency, monitored by the National Aeronautics and Space Administration under contract No. NAG2-582, and by the National Science Foundation under grant NCR-8814187.

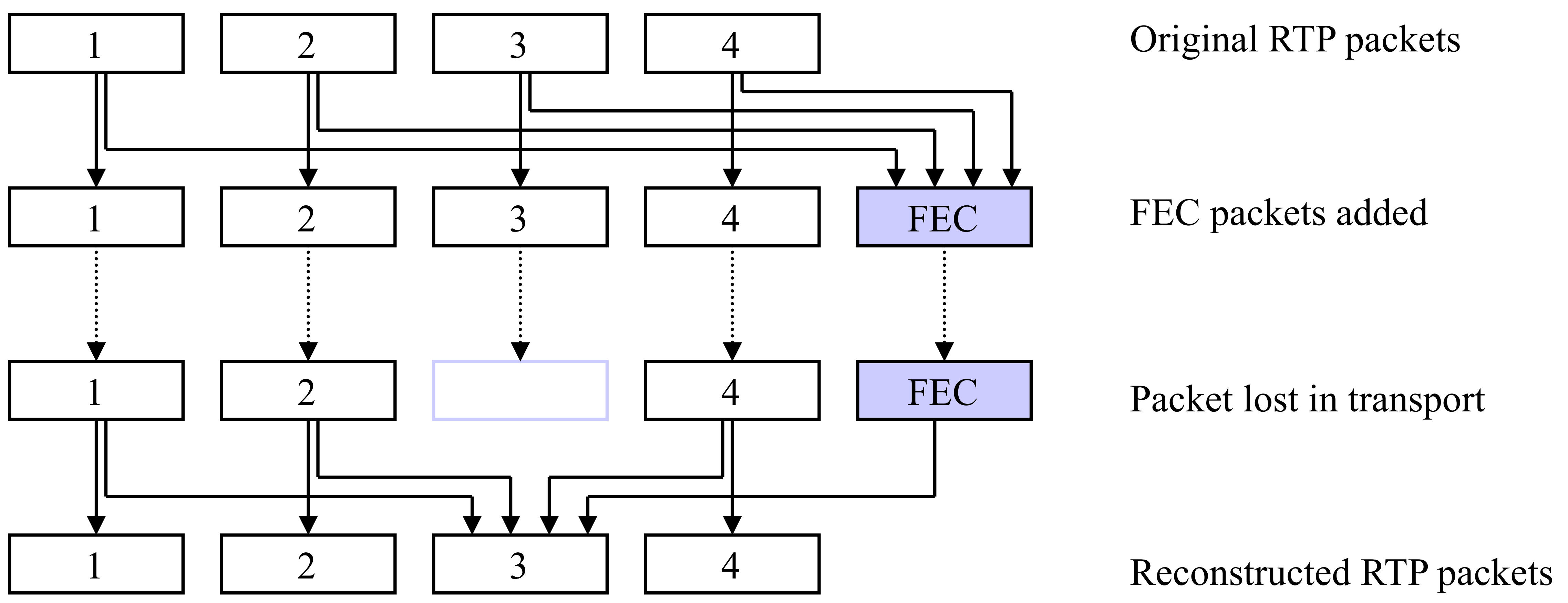
Permission to copy without fee all or part of this material is granted provided that the copies are not made or distributed for direct commercial advantage, the ACM copyright notice and the title of the publication and its date appear, and notice is given that copying is by permission of the Association for Computing Machinery. To copy otherwise, or to republish, requires a fee and/or specific permission.
© 1990 ACM 089791-405-8/90/0009/0200...$1.50

200

D. Clark and D. Tennenhouse, "Architectural considerations for a new generation of protocols", ACM SIGCOMM Conference, September 1990. https://dx.doi.org/10.1145/99508.99553/

# Media Transport: FEC vs. Retransmission

- Retransmission possible, but often takes too long – packet should have been played out before retransmission arrives
- Forward error correction (FEC) often used instead




- Additional FEC packets are sent along with the original data
  - Contain error correcting codes
  - e.g., the Exclusive-OR (XOR) of the original packets – many different FEC schemes
- If some original packets are lost but the FEC packets arrive, original data can be reconstructed

# Interactive Applications

- Architecture for video conferencing
- Multimedia standards
- Media transport