[19] 中华人民共和国国家知识产权局

[51] Int. Cl.
H04L 12/00 (2006.01)
H04L 12/56 (2006.01)
H04L 1/18 (2006.01)

[12] 发明专利说明书

专利号 ZL 200510091911.0

[45] 授权公告日 2009 年 12 月 23 日

[11] 授权公告号 CN 100574199C

[22] 申请日 2005.8.3
[21] 申请号 200510091911.0
[30] 优先权
[32] 2004. 9. 3 [33] US [31] 10/934,823
[73] 专利权人 微软公司
地址 美国华盛顿州
[72] 发明人 李 谨
[56] 参考文献
US2003028623A1 2003.2.6
US2002049760A1 2002.4.25
US20030204602A1 2003.10.30
审查员 王 瑞

[74] 专利代理机构 上海专利商标事务所有限公司
代理人 沈昭坤

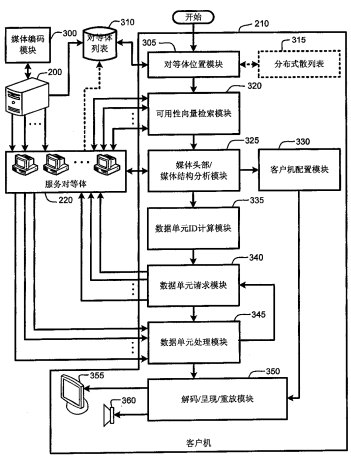
权利要求书 6 页 说明书 33 页 附图 8 页

[54] 发明名称

用于可缩放媒体的分布式流传送的系统和方法

[57] 摘要

“PeerStreamer”提供用于松散耦合的 P2P 网络的接收器驱动对等(P2P)媒体流传送。 该网络中的对等体只执行简单的操作、可以高速缓存流媒体的全部或一部分、不与其他对等体协作、可能是不可靠的、以及可以在任何给定的流传送会话期间脱机或联机。 该网络中的客户机进行实时操作，以便协调对等体、将媒体从多个对等体流传送、执行负载平衡、处理对等体的联机/脱机状态、以及执行对该流媒体的解码和呈现。 在一个实施例中，PeerStreamer 使用高速率擦除弹性编码来允许多个服务对等体保持部分媒体而无冲突，以便客户机仅仅检索固定数量的擦除编码块，而不管在哪里检索和检索什么特定块。 在另一个实施例中，PeerStreamer 使用嵌入编码媒体来根据可用的服务带宽和客户机队列状态而改变流传送比特率。



1. 一种用于在松散耦合的对等网络中提供多媒体数据包的客户机驱动流传送的方法，所述方法包括：

将已编码的媒体文件分成媒体头部和媒体主体；

在一个或多个服务对等体上高速缓存所述媒体头部和所述媒体主体的至少一个或多个数据包，从而使得每个数据包被高速缓存于至少一个服务对等体上；

使用客户机计算机来检索所述服务对等体的列表；

使用所述客户机计算机来从所述服务对等体列表上的服务对等体中的一个或多个中检索可用性向量；

使用所述客户机计算机来基于所检索的可用性向量从所述对等体群集中检索构成所述媒体头部的数据包；

使用所述客户机计算机由所述媒体头部来为所述媒体主体的每个数据包计算数据 ID；以及

使用所计算的数据 ID 来向一个或多个特定服务对等体请求所述媒体主体的特定数据包的传输。

2. 如权利要求 1 所述的方法，其特征在于，高速缓存在每个服务对等体中的数据包的相对比例 p 与所述对等体的服务带宽成正比，与所述媒体的比特率成反比，并服从 1.0 的最大相对比例。

3. 如权利要求 1 所述的方法，其特征在于，所述媒体是嵌入编码媒体，并且，所述数据包对应于已编码媒体的不同的比特率层，并且其中，高速缓存在每个服务对等体中的所述数据包的一相对比例 p 与素数对等体的服务带宽成正比，与所述数据包的比特率成反比，并服从 1.0 的最大相对比例。

4. 如权利要求 1 的所述方法，其特征在于，还包括：响应于所述数据包传输请求，提供所接收的数据包的基于实时客户机的解码，以提供从一个或多个所述服务对等体到所述客户机的流媒体传输。

5. 如权利要求 1 所述的方法，其特征在于，还包括：为已编码的媒体文件生成媒体结构伴随文件。

6. 如权利要求 1 所述的方法，其特征在于，在至少一个所述服务对等体上高速缓存一个或多个数据包包括：将所述数据包分成固定长度的数据单元，并在所述一个或多个服务对等体上高速缓存所述数据单元。

7. 如权利要求 6 所述的方法，其特征在于，在将所述数据单元高速缓存在所述服务对等体上之前，每个数据单元被映射到 ID 空间中的唯一 ID，所述 ID 空间对应于所述客户计算机所计算的数据 IDs。

8. 如权利要求 5 所述的方法，其特征在于，所述媒体结构伴随文件定义已编码媒体文件的特征，至少包括已编码媒体文件的每个数据包的时间标记和长度，并且其中，所定义的特征被所述客户机用来计算所述数据 IDs。

9. 如权利要求 1 所述的方法，其特征在于，检索所述服务对等体的列表包括以下任何一种：从所述服务对等体之一中检索该列表；从服务器计算机中检索该列表；以及执行分布式散列表（DHT）查找，以识别所述服务对等体。

10. 如权利要求 1 所述的方法，其特征在于，每个服务对等体的可用性向量包括由每个对应的服务对等体保持的已编码媒体文件的确切部分的简洁说明。

11. 如权利要求 1 所述的方法，其特征在于，所述客户机与每个服务对等体之间的通信，包括客户机传输请求和服务对等体传输，是使用 TCP 通信协议来实现的。

12. 如权利要求 1 所述的方法，其特征在于，所述客户机与每个服务对等体之间的通信使用自动重复请求（ARQ）协议来重发丢失或延迟的数据包。

13. 如权利要求 1 所述的方法，其特征在于，所述已编码媒体文件是被嵌入编码的。

14. 如权利要求 1 所述的方法，其特征在于，在向所述客户机的流媒体传输期间，周期性地更新所述服务对等体的列表。

15. 一种用于在松散耦合的对等网络中将媒体流传送到一个或多个客户机的方法，包括使用计算设备以便：

在一个或多个服务对等体上存储已编码媒体文件的一个或多个数据包，以便每个数据包被高速缓存在至少一个所述服务对等体上，所述已编码媒体文件包括媒体头部和媒体主体；

在每个服务对等体上，确定至少定义该服务对等体上的可用的所存储的数据包的可用性向量；

识别所述服务对等体的列表，并在客户机请求之后来将所述列表提供给客户机计算机；

使用提供给所述客户机计算机的服务对等体列表来从每个列出的服务对等体下载可用性向量，并进一步将所述媒体头部文件从服务对等体下载到所述客户机；

在所述客户机计算机上，从所述媒体头部中计算关于已编码媒体文件的每个数据包的数据 ID，并使用所计算的数据 ID 来向一个或多个特定服务对等体请求所述媒体主体的特定数据包的顺序传输；以及

响应于所述数据包传输请求对所接收的每个数据包进行解码和呈现，以便在所述客户机计算机上提供实时流媒体重放。

16. 如权利要求 15 所述的方法，其特征在于，还包括：计算至少包括已编码媒体文件的每个数据包的时间标记和长度的单独的媒体结构伴随文件，并且其中，使用所定义的特征来计算所述数据 ID。

17. 如权利要求 15 所述的方法，其特征在于，使用所下载的媒体头部来对所述客户机计算机上的解码器和呈现器进行初始化，用于被传送到所述客户机计算机的已编码媒体文件的数据包的顺序解码和呈现。

18. 如权利要求 15 所述的方法，其特征在于，在一个或多个服务对等体上存储已编码媒体文件的一个或多个所述数据包还包括：将每个数据包分成固定长度

数据单元，并在所述一个或多个服务对等体上存储所述固定长度数据单元。

19. 如权利要求 18 所述的方法，其特征在于，在将所述数据单元存储在所述服务对等体上之前，每个数据单元被映射到唯一 ID 空间中，所述 ID 空间对应于所述客户机计算机所计算的数据 ID。

20. 如权利要求 19 所述的方法，其特征在于，向一个或多个特定服务对等体请求已编码媒体文件的特定数据包的顺序传输包括：向一个或多个所述对等体请求构成每个媒体数据包的特定数据单元，并使用所述客户机计算机来再结合所请求的数据单元，以便在对每个数据包进行解码和呈现之前重建每个媒体数据包。

21. 如权利要求 15 所述的方法，其特征在于，将所述服务对等体列表提供给所述客户机计算机包括以下任何一种：从所述服务对等体之一中检索该列表；从服务器计算机中检索该列表；以及执行分布式散列表 DHT 查找，以识别所述服务对等体列表。

22. 如权利要求 15 所述的方法，其特征在于，所述客户机与每个服务对等体之间的通信，包括客户机传输请求和服务对等体传输，是通过使用 TCP 通信协议来转发丢失或延迟的数据包来实现的。

23. 如权利要求 15 所述的方法，其特征在于，被高速缓存在每个服务对等体中的数据包的一个相对比例 p 与所述对等体的服务带宽成正比，与所述媒体的比特率成反比，并服从 1.0 的最大相对比例。

24. 如权利要求 15 所述的方法，其特征在于，所述媒体是嵌入编码媒体，并且，所述数据包对应于已编码媒体的不同的比特率层，并且其中，被高速缓存在每个服务对等体中的数据包的一个相对比例 p 与所述对等体的服务带宽成正比，与所述数据包的比特率成反比，并服从 1.0 的最大相对比例。

25. 一种用于在一个松散耦合的对等网络中，提供从一个或多个非合作对等体到一个或多个客户机的协同接收方驱动媒体流传送系统，包括：

使用服务器来为媒体文件编码并构造一伴随文件，所述伴随文件定义所编码的媒体文件中每个数据包的数据包时间标记和数据包长度，并且然后将所编码的媒体文件和所述伴随文件的数据包中的一个或多个分发给一个或多个服务对等体；

在每个服务对等体上，构造一可用性向量，所述可用性向量定义该服务对等体上所保持的已编码媒体文件的特定数据包；

所述客户机计算机检索一个或多个所述服务对等体的列表，所述列表包括足够的服务对等体，以便所列出的服务对等体的总计保持表示整个已编码媒体文件的数据包；

所述客户机计算机联系每个被列出的服务对等体，并下载每个被列出的服务对等体的可用性向量，并且，从所述服务对等体之一下载所述伴随文件；

在所述客户计算机上，从所述伴随文件中计算所编码的媒体文件的每个数据包的数据 ID，并结合所述可用性向量来使用所计算的数据 ID，用于向一个或多个特定服务对等体请求所编码的媒体文件的特定数据包的传输；以及

响应于所述数据包传输请求，对所述客户机计算机所接收的每个数据包进行解码和呈现。

26. 如权利要求 25 所述的系统，其特征在于，将所编码的媒体文件和所述伴随文件的数据包中的一个或多个分配给一个或多个服务对等体包括：将所编码的媒体文件和所述伴随文件的每个数据包分成固定长度数据单元，并将所述固定长度数据单元存储在所述一个或多个服务对等体上。

27. 如权利要求 26 所述的系统，其特征在于，在将所述数据单元存储在所述服务对等体上之前，每个数据单元被映射到唯一 ID 空间中，所述 ID 空间对应于所述客户机计算机所计算的数据 ID。

28. 如权利要求 27 所述的系统，其特征在于，向一个或多个特定服务对等体请求所编码的媒体文件的特定数据包的传输包括：使用所计算的数据 ID 向一个或多个所述对等体请求构成每个数据包的特定数据单元，并使用所述客户机计算机来再结合所请求的数据单元，以便在对每个数据包进行解码和呈现之前重建每个媒体数据包。

29. 如权利要求 25 所述的系统，其特征在于，被高速缓存在每个服务对等体中的数据包的一个相对比例 p 与所述对等体的服务带宽成正比例，与所述媒体的比特率成反比，并服从 1.0 的最大相对比例。

30. 如权利要求 25 所述的系统，其特征在于，所述媒体是嵌入编码媒体，并且，所述数据包对应于所编码的媒体的不同的比特率层，并且其中，被高速缓存在每个服务对等体中的数据包的一个相对比例 p 与所述对等体的服务带宽成正比例，与所述数据包的比特率成反比，并服从 1.0 的最大相对比例。

用于可缩放媒体的分布式流传送的系统和方法

技术领域

本发明涉及用于松散耦合的 P2P 网络的接收器驱动对等（P2P）媒体流，尤其涉及一种用于在不需要提供对等协作的条件下、在客户机的实时协调和控制之下将媒体从多个对等体流传送到客户机的系统和方法。

背景技术

最近的市场研究已指出，2004 年，美国一半以上的因特网用户已访问某种形式的流媒体。访问流音乐是很流行的活动，而流视频的普及程度正在迅速增高。

不幸的是，与典型的网页不同，流媒体文件的尺寸通常非常大。例如，按 2 兆比特/秒（Mbps）来编码的 3 分钟的影片宣传片会产生 45 兆字节（MB）的媒体文件，这取决于所使用的编解码器。流媒体必须解决的另一个问题是数据包传递的严格定时。因此，流媒体文件的大尺寸和数据包传递定时要求使典型的流媒体服务器设立和运行起来相对较昂贵。例如，一个当前的估算将关于流媒体的时价定为每 1GB 的服务通信量是 10 美元。通过使用 45 MB 文件尺寸的这个例子，这会导致每一被分发的影片宣传片有 0.45 美元的带宽成本。显而易见，随着媒体流量的增加，这类成本会迅速上升。

对于媒体流的相对较高的成本的一个解决方案是使用“对等”（P2P）网络来将该媒体流提供给个别的客户机。一般而言，P2P 网络的基本理念是允许每个对等节点协助媒体服务器分发流媒体。用于流媒体的 P2P 网络的成功已得到用于实现 P2P 网络的大量常规途径。

例如，被称作“终端系统多点传送”和“PeerCast”的常规 P2P 方案使用用于媒体流的应用程序级多点传送（ALM）。具体地，利用 ESM 和 PeerCast，这些对等节点被自我组织到现有 IP 网络上的覆盖树中。然后，沿该覆盖树来分发流数据。随后，在这些对等节点之间共享提供带宽的成本，从而减轻运行媒体服务器的带宽负担（因此减少美元成本）。但是，利用 ESM 和 PeerCast，该分发树的这些叶节点只接收该流媒体，而不促成内容分发。

两个其他的常规方案“CoopNet”和“SplitStream”通过使用跨越来源和对等节点的多个分发树，解决了诸如 ESM 和 PeerCast 等方案的内容分发局限。然后，CoopNet 和 SplitStream 中的每个树可以发送流媒体的单独片断。结果，所有对等节点都可以涉及内容分发。

常规 P2P 媒体流解决方案的另一例子包括被称作“OStream”的流传送方案。OStream 使用“高速缓存和中继站”方法，以便对等节点可以向客户机提供来自其高速缓存的以前分发的媒体。另一个常规系统“GnuStream”提供构建在公知的“Gnutella”系统之上的接收器驱动 P2P 媒体流系统。而被称作“CollectCast”的另一个常规方案积极地寻找最有可能实现最佳流传送质量的服务对等体，同时动态地自适应网络波动和对等体故障。

另一种类型的常规方案提供一种类型分布式文件共享，其中，文件的各个片断跨越许多对等体而被广泛的分发。然后，只要客户机请求下载该文件，就从多个对等体那里（而不是直接从服务器那里）服务该请求。例如，被称作“Swarmcast”的一个这样的方案通过将文件分成小得多的各个片断，来散布施加在网站上的提供流行的可下载内容的负载。一旦用户已安装该了 Swarmcast 客户级程序，其计算机就通过分发（即供应）它们已下载的各个数据片断来自动与其他用户的计算机进行合作，从而减少中央服务器上的总服务负载。被称作“BitTorrent”的类似方案按照很类似的原则来运作。特别是，当在低负载之下时，使用 BitTorrent 方案来供应大型文件的 web 站点将表现得很象典型的 http 服务器，因为它执行该服务本身的大部分。但是，当该服务器负载达到某个相对较高的水平时，BitTorrent 将转变为一个状态，其中，大部分的上传负担由用于供应其他下载客户机的下载客户机本身来承受。

遗憾的是，尽管诸如 Swarmcast 和 BitTorrent 等方案对于分发各个文件片断以显著地根据 P2P 网络规模增加服务器容量而言很有用，但这些系统不适用于有效率地流传送媒体。特别是，诸如 Swarmcast 和 BitTorrent 等方案不关心组成构成正被下载的一个或多个文件的数据包的传递的顺序或定时。这些文件仅仅逐段地从各对等体那里被广播到客户机，然后仅仅按正确顺序在本地重新汇编，以便在客户机计算机上重建原始文件。但是，在流媒体的情况下，必须仔细地考虑和控制数据包的定时和顺序，以便提供该媒体的有效的流传送。

所以，需要一种系统和方法，用于从松散耦合的对等体集合到客户机的媒体流的接收器驱动控制。这种系统不应该要求各个对等体之间的通信或协作。另外，

这种系统和方法应该通过要求客户机执行任何必要的计算操作的大部分，来将施加于对等体上的计算要求减到最少。

发明内容

如这里所描述的“PeerStreamer（对等流传送器）”提供用于松散耦合的 P2P 网络的接收器驱动对等（P2P）媒体流。网络中的对等体只执行简单的操作、可以高速缓存流媒体的全部或一部分、不与其他对等体协作、可以是不可靠的、并可以在任何给定的流传送会话期间脱机或联机。网络中的客户机（或接收器）进行实时操作，以便协调对等体、将媒体从多个对等体流传送、执行负载平衡、处理对等体的联机/脱机状态、以及执行对流媒体的解码和呈现。

注意，这里所描述的 PeerStreamer 系统适用于具有多个客户机和对等体的大型 P2P 网络，但出于解释清楚的目的，下文将一般涉及个别客户机。本领域的技术人员将会理解，所描述的由 PeerStreamer 提供的系统和方法适用于多个客户机。此外，由于这里所描述的对等体用来将媒体供应给接收器或客户机，因此，P2P 网络中的对等体群集在这里通常被称作“对等体”或“服务对等体”。也应该注意，如这里所描述的，这些“服务对等体”不应该与特定的流媒体文件最初所起源的“媒体服务器”混淆。

一般而言，PeerStreamer 提供接收器驱动媒体流传送。PeerStreamer 操作始于每个接收客户机检索保持所请求的流媒体的全部或一部分的附近的对等体的列表。注意，在这个上下文中，媒体服务器也可以担当这些服务对等体之一。这个列表包括 IP 地址、以及一组保持该服务媒体的完整或部分副本的一个或多个相邻服务对等体的监听端口。用于检索这个列表的方法包括：1）直接从媒体服务器中检索该列表；2）从已知的服务对等体中检索该列表；以及 3）使用用于表示这些服务对等体的分布式散列表（DHT）方法。

一旦客户机检索了可用服务对等体列表，该客户机就连接到每个服务对等体，并获得其“可用性向量”。一般而言，每个服务对等体的可用性向量是该服务对等体所保持的媒体的确切部分的简洁说明。然后，这些可用性向量由客户机用来确切地确定各个服务对等体保持该编码媒体的什么块。

例如，在特定的服务对等体保持全部服务媒体的情况下，该对等体的可用性向量可以是指出该服务对等体保持完整的媒体副本的单个标志。同样，如果服务对等体只保持服务媒体的一部分，那么该服务对等体的可用性向量将用信号通知客户

机服务对等体保持该媒体的什么部分，例如，由该服务对等体保持的每个数据包的块数以及块索引。

另外，在使用附加编码（例如，以下所描述的各种擦除编码（erasure coding）技术）的情况下，可用性向量将包括分配给服务对等体的媒体擦除编码关键字、以及由该服务对等体保持的擦除块的数目。此外，如果服务对等体使用擦除编码，并且，该媒体也被嵌入编码，那么，可用性向量将包括所分配的媒体擦除编码关键字、以及该嵌入编码所使用的不同的比特率水平处的每个数据包的擦除块的数目。

一般而言，已编码的媒体文件通常包括“媒体头部”，随后是表示所编码的媒体的多个媒体数据包（即“媒体主体”）。给定该可用性向量，下一个步骤是让客户机检索媒体头部和“媒体结构”的长度，该媒体结构可从将要从对等体群集流传送的已编码媒体文件中导出。一组数据包的媒体结构仅仅是数据包头部加上数据包比特流长度。在检索了这些长度之后，客户机计算该媒体头部和媒体结构的“数据单元 ID”，并按协作方式从该对等体群集中的一个或多个对等体中检索它们。

一旦媒体头部到达，客户机就分析该媒体头部，然后配置或初始化解码和呈现或重放正被流传送的特定类型的媒体（即 MPEG 1/2/4、WMA、WMV 等）所需要的任何音频/视频解码器和呈现设备。一旦完成了这个初始设置阶段，客户机随后就开始如下所述地协调来自媒体主题从对等体群集的正在进行的流传送。

具体地，给定特定流媒体的前述媒体结构，客户机计算流媒体（即媒体主体）的数据包的数据单元 ID，然后逐个地检索那些数据包。在一个相关的实施例中，PeerStreamer 使用嵌入编码媒体，然后，流传送比特率根据可用服务带宽和客户机队列状态而改变。在此情况下，媒体主体的媒体数据包的正在进行的检索对应于将基于可用带宽来提供最小的速率失真的那些数据包。

在任何一种情况下，客户机周期性地更新服务对等体列表，并连接到潜在的新服务对等体。在一个测试实施例中，客户机通过发出对于每个潜在的服务对等体的周期常规 TCP 连接函数调用，来核对潜在的新服务对等体。在客户机建立与新服务对等体的连接之后，它首先检索该前述可用性向量。然后，在接收器/客户机的方向上，该新对等体可以加入该群集中的其他活动对等体。然后，客户机协调这些对等体、根据其服务带宽和内容可用性来平衡这些对等体的服务负载、并将断开的或超时的对等体的无法履行的请求重定向到其他活动对等体中的一个或多个。然后，流传送操作按这个方式继续进行，直到全部流媒体被接收，或者流传送操作被

用户停止。

在一个实施例中，PeerStreamer 使用高速率擦除弹性编码，以允许多个服务对等体保持部分媒体而无冲突，以便客户机仅仅检索固定数目的擦除编码块，而不管在哪里检索和检索什么特定块。在此情况下，将所接收的擦除编码块放入客户机的分级队列，其中，媒体数据包随后被汇编。然后，向下游发送完全被汇编的媒体数据包，以便使用已为解码和呈现或重放正被流传送的特定类型的媒体而配置或初始化的任何音频/视频解码器和呈现设备来对它们进行解码和重放。在此情况下，通过控制分级队列的长度、请求队列的长度和压缩的音频/视频缓冲区的长度，客户机维持某段所需时期（在一个测试实施例中是大约 4 秒）的流缓冲区。然后，使用该组合的缓冲区来防止网络数据包丢失和抖动。

鉴于以上概述，显而易见，这里所描述的 PeerStreamer 提供一种用于在 P2P 网络中提供接收器驱动媒体流传送的独特的系统和方法。除了刚刚描述的这些好处以外，通过下文中的详细描述并结合附图，PeerStreamer 的其他优点也将会变得一目了然。

附图说明

通过以下说明、所附权利要求书和附图，本发明的这些具体特征、方面和优点将得到更好的理解。附图中：

图 1 是描绘构成实现如这里所描述的“PeerStreamer”的示例性系统的通用计算设备的通用系统框图。

图 2 示出了用于如这里所描述的接收器驱动媒体流传送的示例性对等（P2P）网络。

图 3 提供示出用于实现如这里所描述的 PeerStreamer 的程序模块的示例性体系结构流程图。

图 4 示出了如这里所描述的流媒体文件的文件格式。

图 5 示出了如这里所描述的 PeerStreamer 的测试实施例中所使用的“数据单元”。

图 6 示出了如这里所描述的已被分成 8 个数据单元的嵌入编码媒体数据包的局部高速缓存。

图 7 提供了客户机 PeerStreamer 媒体流传送会话的示例 DirectShow™ 过滤器图表。

图 8 提供了表示如这里所描述的 PeerStreamer 请求和分级队列以及流媒体解码、呈现和重放的体系结构系统突，其系统缓冲区由虚线来示出。

图 9 提供了用于到达的数据单元的 PeerStreamer 客户分级队列，以及用于每个服务对等体的 PeerStreamer 客户机请求队列的框图图解。

图 10 提供了示出如这里所描述的 PeerStreamer 的一个实施例的通用操作的操作流程图。

具体实施方式

在本发明较佳实施例的以下描述中，参考附图，这些附图构成其一部分，并且，在这些附图中，通过举例说明，示出可以在其中实践本发明的特定实施例。可理解，在不脱离本发明的范围的前提下，可以利用其他实施例，并且可以进行结构上的更改。

1.0 示范操作环境：

图 1 示出了可以在其上实现本发明的合适的计算系统环境 100 的例子。计算系统环境 100 只是合适的计算环境的一个例子，它并不意在对本发明的使用范围或功能提出任何限制。也不应该将计算环境 100 解释为对示例性操作环境 100 中所示的任何一个组件或组件组合具有任何依赖性或要求。

本发明可用于众多其他的通用或专用计算系统环境或配置。可能适用于本发明的众所周知的计算系统、环境和/或配置的例子包括（但不局限于）：个人计算机、服务器计算机、手持的、膝上型或可移动计算机或通信设备（例如，手机和 PDA）、多处理器系统、基于微处理器的系统、机顶盒、可编程消费电子设备、网络 PC、小型计算机、大型计算机、包括以上任何系统或设备的分布式计算环境等等。

可以在由计算机结合硬件模块（包括话筒阵列 198 的各个组件）而执行的计算机可执行指令（例如，程序模块）的一般上下文中描述本发明。通常，程序模块包括执行特定任务或实现特定抽象数据类型的例程、程序、对象、组件、数据结构等。也可以在分布式计算环境中实践本发明，其中，由通过通信网络连接的远程处理设备来执行任务。在分布式计算环境中，程序模块可以位于包括记忆存储设备的本地计算机存储介质和远程计算机存储介质中。参照图 1，用于实现本发明的示例性系统包括计算机 110 形式的通用计算设备。

计算机 110 的组件可以包括（但不局限于）处理单元 120、系统存储器 130 和系统总线 121，系统总线 121 将包括系统存储器的各种系统组件耦合到处理单元 120。系统总线 121 可以是几种类型的总线结构中的任一种，包括存储总线或存储控制器、外围总线、以及使用各种总线体系结构中的任一种的局部总线。举例来讲（不作限制），这类体系结构包括工业标准体系结构（ISA）总线、微通道体系结构（MCA）总线、增强型 ISA（EISA）总线、视频电子技术标准协会（VESA）局部总线和外围部件互连（PCI）总线（也被称作 Mezzanine 总线）。

计算机 110 通常包括各种计算机可读介质。计算机可读介质可以是可由计算机 110 访问的任何可用介质，它包括易失和非易失介质、可移动和不可移动的介质。举例来讲（不作限制），计算机可读介质可以包括计算机存储介质和通信介质。计算机存储介质包括以用于诸如计算机可读指令、数据结构、程序模块或其他数据等信息的存储的任何方法或技术来实现的易失性和非易失性、可移动和不可移动介质。

计算机存储介质包括（但不局限于）：RAM、ROM、PROM、EPROM、EEPROM、闪存或其他存储器技术；CD-ROM、数字多功能盘（DVD）、或其他光盘存储器；盒式磁带、磁带、磁盘存储器、或其他磁性存储设备；或可以被用来存储所需信息并可以由计算机 110 访问的其他任何介质。通信介质通常具体表现为诸如载波或其他传输机制等已调制数据信号中的计算机可读指令、数据结构、程序模块或其他数据，它包括任何信息传递介质。术语“已调制数据信号”意味着一种信号，其一个或多个特征按为该信号中的信息编码的方式来加以设置或更改。举例来讲（不作限制），通信介质包括有线介质（例如，有线网络或直线连接）和无线介质（例如，声音、RF、红外线和其他无线介质）。以上任何内容的组合也应该被包括在计算机可读介质的范围以内。

系统存储器 130 包括采取易失性和/或非易失性存储器形式的计算机存储介质，例如，只读存储器（ROM）131 和随机存取存储器（RAM）132。基本输入/输出系统 133（BIOS）通常被存储在 ROM 131 中，它包含有助于在计算机 110 内的各个元件之间传送信息（例如，在启动期间）的基本例程。RAM 132 通常包含可立即由处理单元 120 访问和/或目前正由处理单元 120 操作的数据和/或程序模块。举例来讲（不作限制），图 1 示出了操作系统 134、应用程序 135、其他程序模块 136 和程序数据 137。

计算机 110 也可以包括其他可移动/不可移动、易失性/非易失性计算机存储介

质。只举例来讲，图 1 示出了从不可移动的非易失性磁性介质读取或对其写入的硬盘驱动器 141、从可移动的非易失性磁盘 152 读取或对其写入的磁盘驱动器 151，以及从可移动的非易失性光盘 156（例如，CD ROM 或其他光学介质）读取或对其写入的光盘驱动器 155。可以用于该示例性操作环境中的其他可移动/不可移动的易失性/非易失性计算机存储介质包括（但不局限于）盒式磁带、闪存卡、数字多功能盘、数字录像带、固态 RAM、固态 ROM 等。硬盘驱动器 141 通常通过不可移动的存储器接口（例如，接口 140）而被连接到系统总线 121，磁盘驱动器 151 和光盘驱动器 155 通常由可移动的存储器接口(例如，接口 150)连接到系统总线 121。

以上所讨论的和图 1 中所示出的这些驱动器及其相关联的计算机存储介质为计算机 110 提供计算机可读指令、数据结构、程序模块和其他数据的存储。在图 1 中，例如，硬盘驱动器 141 被示为存储操作系统 144、应用程序 145、其他程序模块 146 和程序数据 147。注意，这些组件可以等同于或不同于操作系统 134、应用程序 135、其他程序模块 136 和程序数据 137。这里为操作系统 144、应用程序 145、其他程序模块 146 和程序数据 147 提供不同的标号，以说明它们至少是不同的副本。用户可以通过输入设备（例如，键盘 162 和通常指鼠标、跟踪球或触摸垫的定点设备 161），来将命令和信息输入计算机 110。

其他输入设备（未示出）可以包括操纵杆、游戏垫、圆盘式卫星天线、扫描仪、无线电接收器、以及电视或广播视频接收器、或类似的输入设备。这些和其他输入设备经常通过被耦合到系统总线 121 的有线或无线用户输入接口 160 连接到处理单元 120，但也可以由其他常规接口和总线结构（例如，并行端口、游戏端口、通用串行总线（USB）、IEEE 1394 接口、Bluetooth™（蓝牙）无线接口、IEEE 802.11 无线接口等）来连接。另外，计算机 110 也可以包括语音或音频输入设备（例如，话筒或话筒阵列 198）、以及扩音器 197 或经由音频接口 199 而连接的其他声音输出设备，也包括常规的有线或无线接口（例如，并行、串行、USB、IEEE 1394、Bluetooth™ 等）。

监视器 191 或其他类型的显示设备也经由接口（例如，视频接口 190）而被连接到系统总线 121。除监视器以外，计算机也可以包括其他外围输出设备（例如，打印机 196），这些外围输出设备可以通过输出外围接口 195 来连接。

计算机 110 可以使用与一台或多台远程计算机（例如，远程计算机 180）的逻辑连接而在联网环境中进行操作。远程计算机 180 可以是个人计算机、服务器、路由器、网络 PC、对等设备、或其他普通网络节点，它通常包括以上相对于计算机

110 而描述的许多或所有元件，尽管图 1 中只示出了记忆存储设备 181。图 1 中所描绘的逻辑连接包括局域网（LAN）171 和广域网（WAN）173，但也可以包括其他网络。这类联网环境在办公室、企业范围的计算机网络、内联网和因特网中很普遍。

当被用于 LAN 联网环境中时，计算机 110 通过网络接口或适配器 170 而被连接到 LAN 171。当被用于 WAN 联网环境中时，计算机 110 通常包括调制解调器 172 或用于通过 WAN 173（例如，因特网）建立通信的其他装置。调制解调器 172 可以是内置的，也可以是外置的，可以经由用户输入接口 160 或其他适当的机制而被连接到系统总线 121。在联网环境中，相对于计算机 110 或其各个部分而描绘的程序模块可以被存储在远程记忆存储设备中。举例来讲（不作限制），图 1 将远程应用程序 185 示为驻留在存储设备 181 上。将会理解，所示的网络连接起示例性的作用，可以使用在计算机之间建立通信链路的其他手段。

现在已讨论了示例性操作环境，本描述的剩余部分将致力于讨论实施“PeerStreamer”的程序模块和过程，该“PeerStreamer”提供对用于分布式媒体流传送的接收器驱动对等（P2P）网络中的一个或多个对等体的群集的动态实时客户机控制。

2.0 引言：

如这里所描述的“PeerStreamer”提供用于松散耦合的 P2P 网络的接收器驱动对等（P2P）媒体流传送。该网络中的对等体只执行简单的操作、可以高速缓存流媒体的全部或一部分、不与其他对等体协作、可以是不可靠的、并可以在任何给定的流传送会话期间脱机或联机。该网络中的客户机进行实时操作，以便协调对等体、使媒体从多个对等体流传送、执行负载平衡、处理对等体的联机/脱机状态、以及执行对流媒体的解码和呈现。

注意，这里所描述的 PeerStreamer 系统适用于具有多个客户机和对等体的大型 P2P 网络，但出于解释清楚的目的，下文将通常涉及个别的客户机。本领域的技术人员将会理解，由 PeerStreamer 提供的所描述的系统和方法适用于多个客户机。此外，由于这里所描述的对等体用来将媒体供应给接收器或客户机，因此，P2P 网络中的对等体群集在这里通常被称作“对等体”或“服务对等体”。也应该注意，如这里所描述的，这些“服务对等体”不应该与特定的流媒体文件最初所起源的“媒体服务器”混淆。

一般而言，PeerStreamer 在 P2P 网络（例如，图 2 所示的网络）中进行操作。对于特定的流传送会话，“服务器”200 被定义为 P2P 网络中最初发起流媒体的节点；“客户机”（或接收器）210 被定义为当前请求流媒体的节点；并且，“服务对等体”220 被定义为向客户机供应流媒体的完整或部分副本的节点。

一般而言，服务器 200、客户机 210 和服务对等体 220 都是与网络（例如，因特网）连接的最终用户节点。由于服务器 200 总是能够供应流媒体，因此，该服务器节点也担当服务对等体 220。服务器节点 200 也可以执行无法由服务对等体 220 执行的媒体管理功能，例如，维持可用服务对等体的列表、执行数字权利管理（DRM）功能等等。此外，对于常规的 P2P 方案，当部署越来越多的流对等体节点 220 时，这里所描述的 PeerStreamer 会得益于效率的提高。特别是，随着流对等体节点 220 的数目的增加，媒体服务器 200 上的负载将会减少，从而运行起来费用不会太大，同时，每个客户机节点 210 将能够在特定的媒体流传送会话期间接收好得多的媒体质量。

此外，应该显而易见，对于许多其他的 P2P 类型的网络，特定节点的角色可能会改变。例如，特定节点可以在一个特定的流传送会话中担当客户机 210，同时，在另一个会话中担当服务对等体 220。另外，特定节点可以同时担当客户节点 210 和服务器 200 或服务对等体 220，以便同时流传送一个或多个媒体文件或媒体文件的各个部分，同时，从一个或多个其他服务对等体接收其他流媒体。

在流传送会话期间，客户机 200 首先定位保持部分或全部所需媒体的多个附近的对等体 220，然后将媒体从这多个对等体（可以包括服务器 200）流传送。因此，每个服务对等体 220 通过供应客户机 210 的下载请求的一个部分，来协助服务器 200 减轻总上传负担。结果，尤其在有许多客户机的情况下，客户机 210 经常可以接收好得多的流媒体质量，因为当有许多流对等体 220 来协助服务器 200 时，有大得多的服务带宽可用。

对于任何 P2P 网络，每一个别的对等体 220 不直接得益于服务一个或多个客户机 210。但是，在一个实施例中，常规的 P2P“公平机制”用来确保与在担当服务对等体的过程中还没有平等地合作的另一个对等体相比较，合作对等体 220 在供应随后的流请求方面接收更高的优先级。因此，当利用 PeerStreamer 来实现这种公平机制时，合作对等体 220 通常可以期待在下一次它变成客户机 210 时会有更好的媒体质量。

因此，认识到每个服务对等体 220 在任何特定的流传送会话期间有效地赞同

客户机 210 和服务器 200 的事实，良好的设计原理是确保服务对等体是轻量级的，且 P2P 网络是松散耦合的。换言之，服务对等体 220 应该只需要执行具有低 CPU 负载的最简单的操作。另外，在一个实施例中，服务对等体 220 也可以选择只高速缓存媒体的一部分，以便将本质上由每个服务对等体赠予的存储空间减到最小。此外，为了降低各个对等体 220 之间的通信的任何带宽成本，不应该要求每个服务对等体与其他对等体协作。最后，运行于任何特定的服务对等体 220 上的其他程序可以在于任何特定的时刻要求 CPU 和网络资源方面具有更高的优先级，或者，特定的对等体可以在任何时候仅仅被打开或关闭。结果，特定的服务对等体 200 可能是不可靠的，可用的服务带宽中有波动。实际上，在流传送会话期间的任何时候，特定的服务对等体可以仅仅脱机或联机。

相反，在客户机 210 上增加负担，以便将资源投入于流传送会话是公平合理的。特别是，客户机 210 需要从多个对等体 220 接收该流媒体，所以，它已被连接到这些对等体。另外，客户机 210 有动力来有效地协调或管理对等体 200，以便改善其自己的流传送体验。因此，这里所描述的 PeerStreamer 系统和方法利用对松散耦合的 P2P 网络中的服务对等体的接收器驱动控制，其中，客户机负责在各个流对等体之中发送和协调数据包请求。

2.1 系统纵览：

如上所述，这里所描述的 PeerStreamer 提供一种用于松散耦合的 P2P 网络的接收器驱动对等（P2P）媒体流传送系统和方法。该网络中的对等体只执行简单的操作、可以高速缓存流媒体的全部或一部分、不与其他对等体协作、可能是不可靠的、以及可以在任何给定的流传送会话期间脱机或联机。该网络中的客户机（或接收器）进行实时操作以便协调对等体、将媒体从多个对等体流传送、执行负载平衡、处理对等体的联机/脱机状态、以及执行对流媒体的解码和呈现。

一般而言，PeerStreamer 提供接收器驱动媒体流传送。PeerStreamer 操作始于每个接收客户机检索保持所请求的流媒体的全部或一部分的附近的服务对等体的列表。注意，在这个上下文中，媒体服务器也可以担当服务对等体之一。这个列表包括一组保持服务媒体的完整或部分副本的一个或多个相邻服务对等体的 IP 地址以及监听端口。用于检索这个列表的方法包括：1）直接从媒体服务器中检索该列表；2）从已知的服务对等体中检索该列表；以及 3）使用用于识别服务对等体的分布式散列表（DHT）方法。

一旦客户机已检索可用服务对等体列表，该客户机就连接到每个服务对等体，并获得其“可用性向量”。一般而言，每个服务对等体的可用性向量是每个服务对等体所保持的媒体的确切部分的简洁描述。然后，这个可用性向量被客户机用来确切地确定服务对等体保持已编码媒体的什么块。

例如，在特定服务对等体保持全部服务媒体的情况下，该对等体的可用性向量可以是指出该服务对等体保持完整的媒体副本的单个标志。同样，如果服务对等体只保持服务媒体的一部分，那么，该服务对等体的可用性向量将用信号通知客户机该服务对等体保持该媒体的什么部分，例如，由该服务对等体保持的每个数据包的块数以及块索引。

另外，在使用附加编码（例如，以下所描述的各种擦除编码技术）的情况下，可用性向量将包括分配给服务对等体的媒体擦除编码关键字、以及由服务对等体保持的擦除块的数目。此外，如果服务对等体使用擦除编码，并且该媒体也被嵌入编码，那么，可用性向量将包括所分配的媒体擦除编码关键字、以及该嵌入编码所使用的不同比特率水平处的每个数据包的擦除块的数目。

给定可用性向量，下一个步骤是让客户机检索关于将要从对等体集群流传送的媒体的“媒体头部”和“媒体结构”的长度。在检索了这些长度之后，客户机计算该媒体头部和媒体结构的“数据单元 ID”，并且，作为已分析每个服务对等体的可用性向量的结果，根据对于什么对等体具有什么数据包 ID 的了解，来从对等体集群内的一个或多个对等体中检索它们。

一旦媒体头部到达，客户机就分析该媒体头部，然后配置或初始化解码和呈现或重放正被流传送的特定类型媒体（即 MPEG 1/2/4、WMA、WMV 等）所需要的任何音频/视频解码器和呈现设备。一旦完成了这个初始设置阶段，客户机随后就开始协调来自如下所述的对等体群集的媒体主体的正在进行的流传送。特别是，给定特定流媒体的前述媒体结构，客户机计算该流媒体的数据包的数据单元 ID，然后从这各个对等体中逐个检索那些数据包。

然后，客户机周期性地更新服务对等体列表（使用用于识别服务对等体的前述方法之一），并连接到潜在的新服务对等体。在一个测试实施例中，客户机通过发出对于每个潜在的服务对等体的周期常规 TCP 连接函数的调用，来核对潜在的新服务对等体。在客户机建立与新服务对等体的连接之后，它首先检索前述可用性向量。然后，在接收器/客户机的方向上，该新对等体可以加入群集中的其他活动对等体。然后，客户机协调这些对等体、根据其服务带宽和内容可用性来平衡这些

对等体的服务负载、并将断开的或超时的对等体的无法履行的请求重定向到其他活动对等体中的一个或多个。然后，流传送操作按这个方式继续进行，直到全部流媒体被接收，或者该流传送操作被用户停止。

2.2 系统体系结构纵览：

以上概述的过程由图 3 中的通用系统框图来示出。具体地，如这里所描述的，图 3 中的系统框图示出了用于实现 PeerStreamer 的程序模块之间的相互关系。应该注意，由图 3 中的折线或虚线来表示的任何方框以及方框之间的互连表示这里所描述的 PeerStreamer 的替换实施例；并且，如下所述，可以结合在这整个文档中描述的其他替换实施例来使用任何或所有这些替换实施例。

一般而言，通过让客户机使用对等体位置模块 305 来检索或识别保持所请求的流媒体的全部或一部分的附近的服务对等体 220 的列表 310，PeerStreamer 开始对于每个客户机 210 的操作。注意，在这个上下文中，媒体服务器 200 也可以担当服务对等体 220 之一。对等体位置模块 305 使用各种方法来检索对等体列表 310。例如，在一个实施例中，直接从服务器 200 提供对等体列表 310。在另一个实施例中，从已知的服务对等体 220 中检索对等体列表 310。最后，在又一个实施例中，对等体位置模块 305 使用常规的分布式散列表（DHT）来识别服务对等体 220。如上所述，对等体列表 305 包括保持服务媒体的完整或部分副本的一个或多个相邻服务对等体 220 的 IP 地址以及监听端口。

服务媒体本身由存在于服务器 200 上的媒体编码模块 300 通过使用多种常规编解码器（包括例如 MPEG 1/2/4、WMA、WMV 等）中的任何一个来进行编码。注意，如这里进一步详细描述的，用来为媒体编码的编解码器可以是嵌入的或非嵌入的。另外，在一个实施例中，如以下进一步详细描述的“高速率擦除弹性编码”结合任何编解码器来使用，以提供对本质上不可靠的服务对等体 220 的提高的健壮性。

最初，所编码的媒体只存在于最初在其上为该媒体编码的服务器上。然后，它全部或部分地被分发给服务对等体 220 中的一个或多个（再一次，服务器 200 也可以担当用于媒体流传送用途的服务对等体）。对服务对等体 220 的分发要么是媒体流的数据包对对等体的直接分发的结果；要么作为当媒体最初被流传送到该服务对等体时令已流传送该媒体的一个或多个对等体（当担当客户机 210 时）仅仅高速缓存该媒体的全部或一部分的结果。在任何情况下，出于解释的目的，假设有多

个已知的对等体（如对等体列表 310 所定义的），并且，每个对等体保持将要流传送的已编码媒体的全部或一部分。

一旦客户机 210 已检索可用服务对等体的列表 310，该客户机就经由从每个对等体中检索前述可用性向量的可用性向量检索模块 320 连接到每个服务对等体 220。接下来，给定每个对等体 320 的可用性向量的信息，客户机 210 随后使用媒体头部/媒体结构分析模块 325 来检索关于将要从对等体群 220 流传送的媒体头部和媒体结构的“媒体头部”和“媒体结构”的长度。在检索了这些长度之后，客户机 210 分析该媒体头部，然后使用客户机配置模块 330 来配置或初始化解码和呈现或重放正被流传送的特定类型媒体（即 MPEG 1/2/4、WMA、WMV 等）所需要的任何音频/视频解码器和呈现设备。

此外，媒体头部/媒体结构分析模块 325 也根据该媒体结构和媒体头部的分析来确定：在对将要流传送的媒体进行编码的过程中，是否已使用嵌入编码媒体或高速率擦除弹性编码中的任何一项或两项。

然后，数据单元 ID 计算模块 335 用于基于媒体头部和媒体结构中所包括的信息来计算流媒体的数据包的“数据单元 ID”。然后，数据单元请求模块 340 使用所计算的数据单元 ID 向对等体群机 220 中的各个对等体请求流媒体的特定数据包或数据块。

在 PeerStreamer 使用嵌入编码媒体的情况下，如以下进一步详细描述的，流传送比特率根据可用的服务带宽和客户机队列状态而改变。在此情况下，由数据单元请求模块 340 提出的对于媒体数据包或数据单元的检索的正在进行的请求对应于将基于可用带宽来提供最小的速率失真的那些数据包（或数据块）。另外，在使用高速率擦除弹性编码的另一情况下，多个服务对等体保持部分媒体而无冲突，以便客户机仅仅检索固定数量的擦除编码块，而不管在哪里检索和检索什么特定的块。

在任何情况下，当客户机 210 经由数据单元处理模块 345 来检索媒体的流传送块时，该客户机将会要么如下所述那样传递将要被解码的那些数据包，要么数据单元处理模块将首先从数据块重建该媒体流的数据包（见以下关于高速率擦除编码的讨论）。此外，客户机 210 将周期性地更新服务对等体列表 310（使用用于识别服务对等体的前述方法之一）。无论何时或按某个所需的频率更新了列表 310，客户机 210 将连接到潜在的新服务对等体，以检索前述可用性向量。然后，接收器/客户机 210 的方向上，该新对等体可以加入群集 220 中的其他活动对等体。

然后，客户机 210 协调对等体 320、根据其服务带宽和内容可用性来平衡这些对等体的服务负载、并将断开的或超时的对等体的无法履行的请求重定向到其他活动对等体中的一个或多个。然后，流传送操作按这个方式继续进行，直到全部流媒体经由解码/呈现/重放模块 350 而被加以接收、解码、呈现和重放。注意，经由常规的显示设备 355 和/或扬声器 360 来提供已解码媒体的重放，向这些显示设备 355 和/或扬声器 360 提供其来自解码/呈现/重放模块 350 的输入。

3.0 操作纵览：

上述程序模块用于实现 PeerStreamer。如以上概述，PeerStreamer 提供用于松散耦合的 P2P 网络的接收器驱动对等（P2P）媒体流传送。以下各个章节详细讨论 PeerStreamer 的操作、以及用于实现根据图 2 而在章节 2 中描述的程序模块的示例性方法。特别地，在详细描述以下在章节 3.1 和 3.2 中所提供的 PeerStreamer 操作之后，在图 10 中呈现操作流程图，它鉴于该详细描述来概述 PeerStreamer 的总操作。

3.1 PeerStreamer 的操作细节：

以下各段详述这里所描述的 PeerStreamer 的特定操作实施例和替换实施例。具体地，以下各段描述 PeerStreamer 所使用的“流媒体模型”；所请求的流媒体的“媒体结构”（基本上是定义计算用于检索媒体数据包或“数据单元”的数据 ID 所需要的流媒体的特征的“伴随文件”）；表示用于流传送的媒体数据包的固定尺寸部分的 PeerStreamer 数据单元；用于降低存储要求的媒体的局部高速缓存；用于提高 PeerStreamer 系统对本质上不可靠的服务对等体的健壮性的媒体的高速率擦除编码。

3.1.1 流媒体模型：

一般而言，流媒体包括当到达时被解码和呈现的数据包流（因此具有名称流传送）。若无流传送，在可以使用它之前，必须以一个大块下载全部媒体。在图 4 中，示出了 PeerStreamer 所使用的流媒体文件的通用结构。

具体地，如图 4 所示的，媒体由“媒体头部”来引导，它包含描述该媒体的全局信息，例如，该媒体中的通道数目、每个通道的属性和特征（音频采样率、视频分辨率/帧速率）、所使用的编解码器、该媒体的作者/版权持有者等。通常在流

传送会话开始之前下载媒体头部，以便客户机可以设立这些必要的工具来对随后接收的数据包进行解码和呈现。注意，流媒体可以包括几个通道，其中每个通道是可以被独立地选择和解码的单独的媒体分量，例如，英语音轨、西班牙语音轨、4:3视频、16:9视频等。

媒体头部后面是媒体数据包序列，其中每个媒体数据包包含跨越短时期的某个通道的压缩比特流。每个媒体数据包由数据包头部来引导，它包含诸如通道索引、数据包的起始时间标记、数据包的持续时间、以及标志数量等信息，例如，该数据包是否是关键帧（例如，MPEG I 帧）、该数据包是否是嵌入编码的数据包（具有可截断的比特流）等等。然后是数据包的压缩比特流。

如今大多数的常规压缩媒体编解码器（例如，MPEG1/2/4 音频/视频、WMA/WMV、RealAudio®/Realvideo®等）生成非嵌入的编码媒体数据包。因此，无法更改这类系统所生成的媒体数据包的大小。而且，只要这一比特流中的媒体数据包之一被丢失或被过度延迟，结果都是：要么该媒体不是可解码的，要么该重放变得紊乱或间断，从而降低流媒体的重放质量。为了保持与这些常规编解码器相兼容，在一个实施例中，PeerStreamer 系统和方法允许媒体数据包是非嵌入编码的（不可缩放）。但是，除了支持传统的压缩媒体格式以外，PeerStreamer 在一个实施例中也支持嵌入的编码媒体。

利用嵌入的编码媒体，每个媒体数据包按可以被独立地向后截断的方法来加以编码。一般而言，两种类型的嵌入编码得到 PeerStreamer 的支持——位平面编码和增强层编码。注意，对本领域的技术人而言，这两种类型的嵌入编码都是公知的。因此，在以下各段中，将只概括地描述这些编码。

例如，利用位平面编码，通常通过逐个位平面地（从最高位平面（MSB）到最低位平面（LSB））对一音频/视频变换系数块进行编码，来实现媒体块的可缩放编码。如果在编码之后截断比特流，那么，为所有这些系数的最高位平面中的几个保留该信息。而且，被截断的比特流对应于较低比特率的压缩比特流，它可以被认为是嵌入在较高比特率率的压缩比特流中，因而是具有名称嵌入编码。结果，可以截断该嵌入编码器所生成的媒体数据包，具有适度的速率-失真折衷。

利用增强层编码，媒体内容被压缩成基层以及一个或多个增强层，其中的每个层通常占据单独的通道。具体地，这类编码允许将要被接收的最低质量媒体流预订该基层。随着每个接连的增强层的添加，解码的媒体质量得到改善。因此，利用这类系统，取决于可用带宽，并通过预订基层和尽可能多的增强层，接收器或客户

机通常可优化所接收的信息的质量。

3.1.2 PeerStreamer 媒体结构：

为了在接收器驱动模式中进行操作，PeerStreamer 客户机需要知道将要被请求的媒体数据包的结构，以便它可以知道向每个对等体请求什么数据包和每个数据包的什么部分。该信息在一种“伴随文件”中提供，该“伴随文件”包括将要被请求的流媒体的结构的定义。一般而言，该媒体结构为 PeerStreamer 客户机提供全部媒体的概观（例如，每个数据包的起始时间标记、每个数据包的持续时间等），以便它可以智能地计划 P2P 流传送会话，并确定特定的媒体数据包及时到达以供解码和呈现。注意，在原先为媒体文件编码时，最初生成包含媒体结构信息的伴随文件；然后，该伴随文件在每个流传送会话的起始处请求时，连同该媒体头部的初始请求一起被流传送到客户机。注意，在由常规编解码器为该媒体编码之后，通过分析该媒体头部和数据包头部，也可以生成伴随文件中的信息。

具体地，一组数据包的媒体结构由数据包头部加上数据包比特流长度组成。因此，该信息可以由客户机用来确定应该请求哪些特定的数据包、应该请求那些数据包的时间、以及应该向其请求那些数据包的对等体。因此，PeerStreamer 在流“设置”阶段首先检索全部媒体的媒体结构。通过在实际上流传送媒体之前检索信息，可引起流设置中的小延迟。但是，通过在媒体流传送之前首先检索信息，在带宽方面没有用于将媒体结构信息供应给客户机的额外成本（在媒体流传送期间）。

注意，相对于流媒体的总长度而言，起始流中的前述延迟通常很小。例如，在 PeerStreamer 的一个测试实施例中，大小范围从 31 兆字节（MB）到 49 MB 的五个测试影片剪辑具有在大约 37 千字节（KB）至大约 53 KB 的范围内的媒体结构伴随文件。所以，该媒体结构大小已被观察到是总媒体主体的大约 0.10-0.15%的数量级。所以，假设服务带宽大于或等于媒体比特率，并且媒体结构是媒体主体的 0.15%，那么，下载 10 分钟剪辑的媒体结构会引起少于 0.9s 的额外的延迟。

在一个相关实施例中，为某个预定长度（即 10 秒、30 秒、1 分钟等）的顺序媒体段生成局部媒体结构。然后，在对应的媒体段将要在不远的将来被流传送之前，只检索每个局部媒体结构。这略微增加带宽要求，因为媒体结构请求和传输可以与媒体数据包请求和传输共存。但是，由于媒体结构的大小在此情况下是如此小，因此，通常可以忽略对全部带宽要求的影响。

3.1.3 PeerStreamer 数据单元：

在一个实施例中，PeerStreamer 将媒体数据包、媒体头部和媒体结构分成长度为 L 的各个固定大小数据单元。对于使用固定大小数据单元的原因是：PeerStreamer 客户机和服务对等体随后可以预先分配大小为 L 的存储器块，从而在流过程期间避免费用昂贵的存储器分配操作。另外，通过将媒体数据包（潜在地说，很大）分成小固定大小数据单元，也可允许 PeerStreamer 客户机将服务负载分配给具有较小粒度的对等体，从而在这些对等体之中实现更好的带宽负载平衡。

一般而言，通过将每个数据包分成$\lceil P/L \rceil$个数据单元，来将长度为 P 的数据包（可以是媒体数据包、媒体头部或媒体结构）分成大小为 L 的块，其中，$\lceil x \rceil$是返回大于或等于 x 的最小整数的常规天棚函数。于是，所有数据单元具有固定长度 L，潜在地除每个数据包的最后一个的数据单元以外（其长度是 P mod L）。

在使用媒体的非嵌入编码的情况下，在网络传输期间，不能丢弃构成每个媒体数据包的数据单元，而不会降低媒体重放质量。所以，这些数据包被指定为“必要数据单元”，因为它们都必须加以接收。

相反，当嵌入编码媒体数据包被分成各个数据单元时，只有基层数据单元必须被传递，并且，如果服务带宽不足够，那么，可以随意地丢弃剩余的数据单元。这些可任选的数据单元被指定为“非必要数据单元”。这些非必要数据单元的供应所要求的带宽可以被计算如下。例如，在嵌入编码的情况下，媒体数据包将持续 T 秒。假设媒体数据包被分成多个数据单元，那么，为了将层 i 处的数据单元供应给客户机，也必须将层 i 以下的所有数据单元供应给客户机。结果，供应层 i 处的数据单元所要求的服务带宽是：

$$R_i = (i+1)L/T \qquad \text{公式 1}$$

所以，公式 1 提供当考虑到嵌入编码媒体时数据单元的比特率 R。然后，通过丢弃将会导致可用服务带宽以上的比特率的非必要数据单元，PeerStreamer 客户机可调整成更改服务带宽。

在任何情况下，无论媒体被非嵌入编码还是被嵌入编码，特定媒体流的所有数据单元（包括媒体数据包、媒体头部和媒体结构的据单元）都被映射到唯一 ID 空间。例如，在一个测试实施例中，媒体数据包的数据单元从 0x00000000 到 0xfdffffff（十六进制）编入索引，媒体头部的数据单元从 0xfe000000-0xfeffffff 编入索引，媒体结构的数据单元从 0xff000000-0xffffffff 编入索引。这个测试实施例中所使用的数据单元是关于图 5 中所示的 PeerStreamer。

注意，为了获得媒体头部和媒体结构的数据单元 ID，首先需要媒体头部和媒体结构的长度。这些被称作它们的“巨大结构”。为了获得媒体数据包的数据单元 ID，需要媒体数据包比特流的长度。该信息被包括在媒体结构中。

3.1.4 媒体的局部高速缓存：

出于服务目的，每个服务对等体只需要保持与其服务带宽成比例的媒体的一部分。与因特网连接的大多数计算机的服务（或上传带宽）常常实质上小于其下载带宽（规定每个特定节点可以接收的最高流传送比特率）。因此，因特网上的每个最终用户节点往往在其上传带宽与其下载带宽之间具有不平衡。例如，给定家庭用户可以获得的典型商业 ADSL/电缆调制解调器网络上的节点，那么，下载带宽比其上传带宽高一个数量级并非是不平常的。同样，校园/企业网上的节点通常已改进了服务带宽，以便任何给定节点参与 P2P 类型活动都不会影响其他任务关键功能。

因此，由于每个服务对等体通常不能够个别地将全部媒体流供应给客户机，因此，不需要将全部媒体流高速缓存在任何一个服务对等体上。所以，用于减少服务对等体中的任一个所要求的存储资源数量的有效方法是允许每个服务对等体只保持将要被流传送的媒体的一部分。例如，如果流传送非嵌入编码媒体所需要的比特率是 R，并且，流传送会话中的对等体所提供的最大服务带宽是 B，那么，每个对等体节点只需要在其高速缓存中保持该流媒体的 p 部分，其中的值 p 由公式 2 来表示：

$$p = \max(1.0, B/R) \qquad \text{公式2}$$

例如，假设媒体比特率是服务带宽的两倍，即，$R = 2B$。然后，服务对等体只需要在其存储器中保持该流媒体的一半，因为该对等体无法独自按完全的流传送比特率来服务于客户机。实际上，给定这个例子的前述限制，该对等体所能够做得最好的是至多提供该媒体的一半。因此，该对等体只需要在其高速缓存中保持该媒体的一半。然后，将要被流传送的媒体的其余部分必须由其他服务对等体来提供。

另外，然后应该注意，公式 1 和 2 的组合随后允许确定为使用嵌入编码媒体的情况而保持的媒体量。如以上章节 3.1.3 中所讨论的，嵌入编码媒体的媒体数据包被分成具有不同的比特率的多个数据单元。所以，R 是特定层 L 的数据单元的比特率，公式 2 现在给出将要为该数据单元而保持的媒体的部分。例如，如图 6 所示，嵌入媒体数据包可以被分成多个数据单元（在这个例子中是 8 个）。如图 6 所示，

需要为每个数据单元（具有 $L/T=0.5B$）高速缓存的媒体量被示出为然后根据公式 2 来确定。

但是，在一个实施例中，在特定服务对等体的存储资源足够大的情况下，该服务对等体可以通过仅仅使用公式 2 中的较高的“潜在服务带宽”B’，来选择高速缓存该媒体的一个较大的部分。然后，被高速缓存的媒体的该额外部分允许按紊乱的、但高质量的方式来供应该媒体。例如，假设每个服务对等体选择使用其实际服务带宽两倍的潜在服务带宽 B'（即，B' = 2B）”，那么，该 P2P 网络中的所得的媒体量将足够客户机按流传送速率的一半来检索该媒体。换言之，假设所有这些可用对等体的合计服务带宽大于 $R/2$，那么，客户机应该能够首先下载该媒体的一半，然后连续不断地流传送剩余的一半传送并对其进行重放。同样，客户机也可以选择下载该媒体的 $T_s/2$ 片段（具有时间 T_s）、连续不断地流传送另一个 $T_s/2$ 片段并重放该片段、然后下载另一个片段并流传送它。这样，该流媒体可以按速率 R 来重放（纵使按紊乱的方式）。

3.1.5 媒体的高速率擦除编码：

如上所述，对等体可能本质上是不可靠的。因此，有利的是提供用于在 PeerStreamer 系统和方法中提供增加的冗余度的某个手段，以便有效地处理服务对等体的本质上不可靠的服务行为。对这个问题的处理提出了必须解决的许多关注问题。例如，确定该媒体的哪个部分 p 应该由每个对等体来保持是所关注的。另外，由于媒体最终被分成前述数据单元，因此，确定每个对等体应该保持这些数据单元的哪个部分 p 也是所关注的。

解决这些问题的一个策略是：仅仅将每个数据单元分成 k 个块。保持该媒体的 p 部分的对等体随后可以随机地保持$\lceil k \cdot p \rceil$个块，$\lceil x \rceil$是前述天棚函数。但是，对于这个方案的随机性的一个问题是：即使对等体群集中存在比 k 个块多得多的块，该群集作为一个整体也可能会缺乏特定的块 j，从而致使整个数据单元无法恢复。另外，在这种方案中，客户机仍然负责定位来自这些对等体的每个截然不同的块，这使客户机与对等体之间的协议设计复杂化。

因此，更好的策略是：使用“高速率擦除弹性代码”来确保对等体中的一个或多个将具有重建特定的数据单元所必要的数据块，同时简化客户机上识别对等体中的哪个对等体包含该必要数据的要求。一般而言，擦除弹性代码是具有参数（n, k）的块纠错码，其中，k 是原始消息的数目，n 是编码消的息的数目。高速率擦除

弹性代码满足 n 比 k 大得多的属性；这样，k 个原始消息被扩大为 n 个消息的大得多的编码消息空间。尽管擦除编码技术一般被公认为用于为数据编码，但如这里所描述的，该技术在 P2P 网络环境中流传送媒体的应用是未知的。

作为块纠错码，可以通过伽罗瓦域（Galois Field）GF(p)上的矩阵乘法来描述高速率擦除弹性代码的操作：

$$\begin{bmatrix} c_0 \\ c_1 \\ \vdots \\ \vdots \\ c_{n-1} \end{bmatrix} = \mathbf{G} \begin{bmatrix} x_0 \\ x_1 \\ \vdots \\ x_{k-1} \end{bmatrix}, \qquad \text{公式 3}$$

其中，p 是伽罗瓦域的阶，$\{x_0, x_1, \cdots, x_{k-1}\}$是原始消息，$\{c_0, c_1, \cdots, c_{n-1}\}$是编码的消息，G 是生成矩阵。注意，公式 3 不被用来立即生成所有编码的消息。而是，生成矩阵 G 定义已编码消息空间。所以，当客户机接收 k 个编码的消息$\{c'_0, c'_1, \cdots, c'_{k-1}\}$时，它们可以被公式 4 表示为：

$$\begin{bmatrix} c'_0 \\ c'_1 \\ \vdots \\ c'_{k-1} \end{bmatrix} = \mathbf{G}_k \begin{bmatrix} x_0 \\ x_1 \\ \vdots \\ x_{k-1} \end{bmatrix}, \qquad \text{公式 4}$$

其中，G_k 是对应于已编码消息的、生成矩阵 G 的 k 行所形成的子生成矩阵。另外，如果子生成矩阵 G_k 具有满秩 k，那么，矩阵 G_k 可以被求逆，从而可以为原始消息解码。

可以使用几种众所周知的擦除编码技术，包括（例如）Reed-Solomon 擦除码、tornado 码和 LPDC 码。但是，在一个实施例中，PeerStreamer 基于伽罗瓦域 GF(2^{16})上的修改过的 Reed-Solomon 码来提供一种新的高速率擦除弹性代码。在这个例子中，原始消息的数目 k 是 16。已编码消息空间的大小 n 是 $2^{16} = 65536$。Reed-Solomon 码是最大距离可分（MDS）码。因此，生成矩阵 G 的任何 16 行形成具有满秩 16 的子生成矩阵。换言之，可以从任何 16 个已编码消息中恢复原始消息。应该注意，也可以使用其他域大小 p；并且，PeerStreamer 不局限于这里所描述的特定域大小的运用。另外，对于使用非 MDS 擦除编码的实施例，取决于所使用的特定擦除编码，可能有必要检索 $k' \geq k$ 个块，以恢复原始消息。使用基于 Reed-Solomon 的擦除代码部分是因为它们是 MDS 码，并且，它们可以被有效地编码和解码，同时，只将很少的计算额外开销施加于大多数常规计算机的 CPU 上。

利用高速率（n, k）擦除弹性代码，为每个对等体节点分配 n 的已编码消息空间中的 k 个关键字，每个关键字是生成矩阵 G 的行索引。关键字分配可以由服务器来执行。另外，如果高速缓存该媒体的对等体的数目小于 n/k，那么，可以为每个对等体分配一组唯一的关键字。结果，可以保证每个对等体保持与众不同的已编码消息。这个策略提供许多好处，但它仍然要求中心协调节点（例如，服务器）。

因此，在另一个实施例中，通过允许每个对等体选择 k 个随机关键字，来消除中心协调节点的这个角色。如果对等体节点的数目大于 n/k，或者没有利用中心协调节点来分配关键字，那么，某些对等体节点可以保持相同的关键字。然而，在其中客户机被连接到 m 个对等体的大多数媒体流传送会话中，m 通常比 n/k 小得多。所以，两个服务对等体碰巧保持相同的关键字，并且因此这些对等体之一的一个关键字无用的概率很小。但是，即使有关键字冲突，当客户机首先连接到对等体时，它也可以容易地识别这类冲突。在识别这种冲突的情况下，该客户机仅仅在流传送会话的剩余部分内使这些重复的关键字中的一个无效。因此，客户机不需要实际上在流过程期间处理关键字冲突。

例如，假设 S1 和 S2 分别是服务对等体 1 和服务对等体 2 的擦除编码关键字空间，并且 S1 = {1, 7, 23, 43, 48}，S2 = {3, 7, 28, 49, 99}。显而易见，关键字空间 S1 和 S2 是不同的。但是，关键字 7 由这两个关键字空间来共享，所以，服务对等体 1 和服务对等体 2 可以保持共享同一关键字（即关键字“7”）的擦除编码块。所以，在请求特定编码块之前，根据服务对等体之一来使关键字“7”无效，以便只从这些对等体之一中检索由关键字“7”编码的那个块，从而避免由重复关键字引起的任何解码冲突。但是，应该注意，在一个服务对等体在媒体流传送操作期间脱机的情况下，如果作为使用一个或多个重复关键字的结果该脱机服务对等体以前处于冲突状态，那么，可以使另一个服务对等体的特定的已无效编码关键字重新生效。

利用（65536, 16）Reed-Solomon 码，每个数据单元被分割成 16 个块。通过使用一组预先分配的关键字，该对等体选择高速缓存$\lceil 16p \rceil$个擦除编码块，其中，p 是从公式 1 和 2 中计算的参数。被分配给该对等体的关键字、以及其最大服务带宽 B 组成该对等体的前述可用性向量，因为该客户机可以通过使用该对等体可用性向量所提供的信息来确定，该对等体保持多少和什么擦除编码块（按数据单元/块 ID）。再一次，在最初连接每个对等体的时候，该客户机解决任何关键字冲突。在流传送会话期间，客户机随后可以从任何服务对等体节点中检索任何 k 个已编码消息，并

对相关联的数据单元进行解码。

另外，没有必要将用于为特定数据单元解码的编码块的整个集合存储在任何一个服务对等体上。换言之，对于任何特定数据单元的任何特定服务对等体所保持的块的数目可能小于 k。所以，在一个实施例中，只生成实际上正被传递到特定对等体的那些编码块，而不是浪费计算能力来计算每个编码关键字的每个已编码块。换言之，如果 $j < k$ 个块被存储在特定的服务对等体上，那么，只应该为该特定数据单元生成 j 个块。

3.2 P2P 网络中的 PeerStreamer 操作的实现：

在以下各段中，鉴于前面对 PeerStreamer 的操作细节的讨论，来描述 PeerStreamer 操作的实现。具体地，以下各段描述客户机对服务对等体的定位；基于所检索的媒体结构的客户机解码和呈现的设置；PeerStreamer 网络连接；流传送比特率控制；PeerStreamer 客户机请求和对等体答复；以及，最后是 PeerStreamer 请求和分级队列。

3.2.1 定位服务对等体：

如上所述，客户机所执行的第一项任务是获得保持服务媒体的完整或部分副本的相邻服务对等体的列表的 IP 地址以及监听端口。另外，该列表也在媒体流传送会话期间被更新。如以上所解释的，用于获得该列表的一般方法包括：1）从服务器中检索该列表；2）从已知的服务对等体中检索该列表；以及 3）在预先既不知道媒体服务器，也不知道服务对等体的情况下，使用用于识别服务对等体的分布式散列表（DHT）方法。

3.2.2 解码和呈现设置：

在获得服务对等体列表之后，客户机试图连接到这些服务对等体中的每一个。如上所述，一旦被连接，客户机就检索每个对等体的可用性向量，并解决任何关键字冲突。然后该客户机从这些对等体之一中检索媒体头部和媒体结构的长度。在检索了这两个长度之后，构造媒体头部和媒体结构的数据单元的 ID。然后，按如章节 3.2.6 中进一步详细描述的 P2P 方式来检索媒体头部和媒体结构。一旦检索了媒体头部，客户机就确定应该对哪些解码器和呈现器进行初始化，以便在将媒体流传送到客户机时对它进行解码和呈现。

在使用 DirectX™ 来实现的一个测试实施例中，该设置通过首先根据媒体头部中所提供的信息来构造 DirectShow™ 过滤器图来实现。应该注意，这里所描述的 PeerStreamer 不局限于使用 DirectX™ 功能的实现，并且，只出于解释的目的来提供 DirectX™ 的运用及其相对于测试实施例的讨论，用于描述在为客户机重放而解码和呈现流媒体的过程中客户计算机的设置。

所以，假设客户设置的 DirectX™ 实现，客户机的网络组件由 DirectShow™ 网络源过滤器来表示，其输出被馈入正确的音频/视频解码器 DirectX™ 媒体对象（DMO）。然后，该 DMO 被进一步连接到适当的音频/视频呈现设备。例如，图 7 示出了客户机 PeerStreamer 媒体流传送会话的示例 DirectShow™ 过滤器图。在这个例子中，流媒体被非嵌入编码。音频比特流按 WMA 压缩，视频比特流按 MPEG-4 压缩。

经由 DirectShow™ 框架来使用和执行 PeerStreamer 客户机设置的一个优点是它可以使用在 DirectShow™ 之下开发的巨大的现有音频/视频编码器/解码器库。例如，利用 DirectShow™，PeerStreamer 客户机能够解码和呈现由各种编解码器（包括例如 MPEG 1/2/4、WMA/WMV、Indeo Video 等）或具有 DirectShow™ 解码器 DMO 组件的任何其他编解码器来编码的媒体。DirectShow™ 也提供附加的音频/视频处理模块，例如，分辨率/色彩空间转换和解除交错，以便所解码的音频/视频可以自动与客户机的音频/视频呈现设备的性能相匹配。

另外，DirectShow™ 自动处理音频/视频轨道的同步。例如，在音频流保持全部流的参考时钟的情况下，当播放流视频时，DirectShow™ 确保该视频流的系统定时时钟尽可能接近于用于解决诸如嘴唇同步等问题的音频流时钟。最后，DirectShow 应用程序本质上是多线程的。因此，在多处理器 PC（或启用了超线程的 PC）上，客户机的各个组件（例如，网络组件、音频解码器、视频解码器和音频/视频呈现引擎等）的计算负载可以被分发到多个处理器上。这大大加速了客户机的执行，并允许使用更复杂的音频/视频解码器。

最后，也应该注意，这里所描述的 PeerStreamer 不局限于使用 DirectX™ 功能的实现；并且，出于解释的目的，提供了 DirectX™ 的运用及其相对于测试实施例的讨论，只用于描述在为客户机重放而解码和呈现流媒体的过程中客户机计算机的设置。

3.2.3 PeerStreamer 网络链路和数据包丢失管理：

大多数媒体流客户机（例如，Windows®媒体播放器、或 RealPlayer®）使用在 UDP 之上所携带的公知的实时传送协议（RTP）。通常为媒体流应用程序选择 UDP/RTP，这是因为：1）UDP 协议支持 IP 多点传送，它在将媒体发送到启用了 IP 多点传送的网络上的一组节点的过程中会是有效率的；以及 2）UDP 协议不具有任何重发或数据速率管理功能。因此，流传送服务器和客户机可以实现行高级数据包传递功能（例如，前向纠错（FEC）），以确保媒体数据包的及时传递。

但是，与以上所标识的公知的媒体流传送方案对比而言，PeerStreamer 将 TCP 连接用作客户机与服务对等体之间的网络链路。选择 TCP 连接而不是常规的 UDP/RTP 协议的一个原因是：由于诸如域内路由协议、ISP 商业模型（收费模型）、沿分布树的拥塞控制等问题，在真实世界中，没有广泛地采用 IP 多点传送。

此外，与许多商业媒体播放器一样，PeerStreamer 客户机并入（在测试实施例中是 4s 的）流媒体缓冲区，以防止诸如抖动和拥塞等网络异常。实际上，给定比客户机与服务对等体之间的往返时间（RTT）大许多倍的流媒体缓冲区，那么，TCP ARQ（自动化重复请求）机制对于媒体数据包在充分的时间内的传递而言是足够好了，以便提供流媒体的流畅的重放。

一般而言，有三种公知的机制（具有大量公知的变体）用于解决媒体数据包丢失。例如，这些机制通常包括：FEC、选择性数据包重发、以及自动重复请求（ARQ）。PeerStreamer 可以使用所有这些数据包丢失机制中的任一种。但是，如以下所解释的，使用特定的机制有胜过其他机制的各种优点。

具体地，对于因特网信道（可以被认为是具有变化的特征和未知的数据包丢失率的擦除通道），固定的 FEC 方案要么浪费带宽（具有太多的保护），要么无法恢复丢失的数据包（具有太少的保护）。这样，它未能有效地利用客户机与对等体之间的带宽资源。所以，利用比 RTT 大许多倍的流传送缓冲区、以及（因而）关于重发的许多机会，基于重发的出错防止（例如，选择性重发和 ARQ）比 FEC 更可取。

考虑到 ARQ 和选择性重发，可见到，在使用 TCP 协议的因特网信道中，只有当许多数据包没有被选择来重发的时候，选择性重发将比 ARQ 强。对于非嵌入编码媒体，丢失的数据包通常会导致严重的重放降级，包括无法解码和提供特定数据包的重放。所以，丢失的数据包几乎总是被重发。相反，利用嵌入编码媒体，丢失的数据包可能不会阻止媒体重放。但是，任随机数据包的丢失仍然会使许多导出的数据包变得不可用。结果，只有最高增强层数据包可能不会被选择来重发。

与选择性重发相比较，一旦请求数据包，ARQ 总是重发它们；即使它们属于最高增强层。然而，ARQ 方案可以选择不请求以后媒体数据包的最高增强层数据包，从而利用选择性传输方案来实现相同的带宽使用率和察觉到的媒体重放质量。因此，除非网络条件变化非常迅速，否则，TCP 协议所使用的 ARQ 机制足以处理媒体流传送中的数据包丢失。

将 TCP 用作网络协议也可提供胜过常规媒体流传送方案（例如，以上所标识的方案）的几个额外的好处。例如，利用 TCP，不需要明确地处理流量控制、吞吐量估算、拥塞控制和避免、保活（keep alive）等等。所有这些问题由 TCP 协议来自动处理。TCP 协议也可以检测脱机的对等体，并适度地处理该对等体与客户机之间的连接链路的关闭。

3.2.4 具有嵌入编码的 PeerStreamer 流传送比特率控制：

非嵌入编码的媒体较佳地总是按该媒体的比特率来流传送，以避免在客户机处的媒体重放降级。但是，嵌入编码媒体的流传送比特率可以在流传送会话期间变化。

所以，在一个实施例中，每个嵌入编码媒体数据包的流传送比特率 R_{recv} 首先由公式 5、6 和 7 来计算，如下所示：

$$R_{raw} = Th \cdot (1 + T_{rft} - T_{staging}) + B_{staging} - B_{outstanding} \quad \text{公式 5}$$

$$R_{filter} = (1-\alpha)R_{filter} + \alpha R_{raw} \quad \text{公式 6}$$

$$R_{recv} = \min(R_{min}, R_{inst}) \quad \text{公式 7}$$

其中，Th 是多个服务对等体的总计的服务带宽，$T_{staging}$ 是目标分级缓冲区大小（在测试实施例中具有默认值 2.5s），T_{rft} 是所需的请求履行时间（在测试实施例中具有默认值 1.0s），$B_{staging}$ 是分级队列中所接收的数据包的长度，$B_{outstanding}$ 是将要被接收的未完成答复的长度，R_{min} 是基层比特率（只具有必要数据单元），α 是低通控制参数。

然后，使用公式 5-7 的结果来通过以下总计的服务带宽 Th、以及各个分级和请求队列状态而控制流传送比特率 R_{recv}，这些分级和请求队列状态在以下章节 3.2.6 中进一步详细描述。一旦确定该流传送比特率，客户机就只发出对具有低于流传送比特率 R_{recv} 的比特率的数据单元的请求。

在一个相关实施例中，通过也考虑数据单元的失真作用，使用更高级的策略来控制比特率 R_{recv}。但是，这要求客户机获得对数据单元的失真（或速率失真斜率）

的访问，它必须被包括在媒体结构中并被发送到客户机。但是，与媒体结构中的现有信息不同，数据单元的失真在解码的过程中是不需要的，因而被认为是额外开销。因此，它是将要被发送到客户机的额外开销量与速率控制准确度之间的折衷。

3.2.5 PeerStreamer 数据块请求和答复：

图 8 概括地示出了客户机数据块请求及其对对等体的答复的使用期限。具体地，如图 8 所示，客户机生成该请求，并通过出站 TCP 连接来将它发送到特定的服务对等体。另外，在网络传递中，TCP 可以将该请求与发给同一对等体的原先的请求捆绑在一起。如果原先的请求在传输过程中丢失，那么，TCP 也处理该请求的重发。

在数据包请求被传递到对等体之后，它被存储在该服务对等体的 TCP 接收缓冲区中。然后，该对等体处理这些请求，每次处理一个请求。对于每个请求，对等体从其磁盘或记忆存储中读取所请求的块（可能被或可能不被加以擦除编码，取决于所使用的编码），并将所请求的内容发回到客户机。倘若从服务对等体到客户机的 TCP 套接字被阻断（即，不再有带宽可用），那么，服务对等体将阻断进一步的客户机请求，直到 TCP 连接打开为止。

客户机发出请求的时间与客户机接收其答复的时间之间的间隔被定义为请求履行时间（RFT）。请求通常比其答复小得多，并且，与用来发回内容的网络传递时间相比，处理请求的过程中所涉及的操作（例如，磁盘读取）通常是不重要的。所以，请求的 RFT－T_{rft} 由公式 8 来计算，如下所示：

$$T'_{rft} = (B_{i,\,outstanding} + B_{cur}) / Th_i \qquad \text{公式 8}$$

其中，T_{hi} 是对等体 i 的服务带宽，$B_{i,outstanding}$ 是在该请求之前的未被接收的答复的长度，B_{cur} 是所请求的内容的长度。所以，RFT 是根据对等体的服务带宽、请求的大小、以及来自该对等体的未被接收的内容的大小来确定的。

一旦所请求的内容数据包到达客户机处，它就被立即移动到分级队列。在该分级队列中，来自多个对等体的数据块（可以包括擦除编码块）被组合和解码为数据单元，这些数据单元被进一步组合成媒体数据包。客户机周期性地从分级队列中除去所传递的媒体数据包，并将它们压入对应的音频/视频解码器。在媒体数据包被解码器解压之后，未压缩的音频/视频数据流被发送到音频/视频呈现单元，用于客户机重放设备（监视器、扬声器等）上的流重放。

在一个实施例中，图 8 中所示的缓冲区用来防止网络异常（例如，数据包丢

失和抖动）。（但是，当使用 DirectShow™ 实现时，未压缩的音频/视频缓冲区在 DirectShow 过滤器图的控制之下，并且是不可编程的。）在 PeerStreamer 的一个测试实施例中，分级缓冲区的大小被设为 $T_{staging}=2.5s$，所需的 RFT 被设为 $T_{rft}=1.0s$，并且，所压缩的音频/视频缓冲区被设为 0.5s。因此，在这个测试实施例中，PeerStreamer 客户机的总缓冲区因而大约是 4s。

在其中使用擦除编码的实施例中，每个数据块请求被明确地表达为某个数据单元的一组擦除编码块的请求。可以利用起始块索引和所请求的块的数量来标识该擦除编码块组。可以通过 32 位 ID 来标识该数据单元。该请求因而采取以下形式：

Data_Unit_ID [32], *Start_Index* [4], *Number_of_Blocks* [4] 公式 9

其中，括号中的数字是每个分量的比特数。

所以，如公式 9 所示，在擦除编码块的情况下，每个请求是 5 字节长。另一方面，所请求的内容在大小方面的范围是 128~2048 个字节（数据单元长度 $L=2048$，$k=16$）。结果，请求的大小只是答复的大约 0.24%~3.91%。所以，客户机用于发送请求的上传带宽的量相对于请求的内容而言非常小。

3.2.6 PeerStreamer 请求和分级队列：

如上所述，PeerStreamer 客户机维持单个分级队列，以保持所接收的数据块（可以被擦除编码）；并且，从其中，数据块被汇编成数据单元，然后被汇编成媒体数据包。客户机也保持用于服务对等体中的每一个的单独的请求队列，以保持被发送到每个对等体的无法履行的请求。图 9 示出了请求队列和分级队列的一个例子。

分级队列是 PeerStreamer 客户机的主要流缓冲区。所有接收到的内容首先被放入该分级队列。这些请求队列用于三个目的：1）用于执行吞吐量控制和负载平衡；2）用于识别由每个服务对等体发回的答复；以及 3）用于处理断开的对等体。

请求队列的第一个功能是：平衡务对等体之中的负载。在媒体被擦除编码的情况下，对数据单元的请求被分成多个擦除编码块组的请求，每个组被定向到一个对等体。通过以下操作来生成这些请求。一请求数据单元，客户机就首先检验对等体的可用性向量，并对该数据单元计算每个对等体所保持的擦除编码块的数量（a_i）。如果所有联机对等体所保持的块的总数小于 k，那么，该数据单元不可检索。如果不可检索的数据单元是非必要的（即嵌入编码媒体的非基层），那么，客户机只需跳过该数据单元。

相反，如果不可检索的数据单元是必要的（即，属于非嵌入编码媒体数据包、

或嵌入编码媒体数据包的基层），那么，客户机无法继续进行流媒体的下载和重放。所以，在一个实施例中，它将等待更多的对等体联机，以提供这些缺少的块。在一个替换实施例中，客户机将跳过整个媒体数据包，并向以后的音频/视频解码器将它标记为“缺少的”。其结果将是所呈现的媒体中的间隙或跳跃。但是，如果无法从对等体群集中重新获得一个必要的数据单元，那么，以后更多的必要数据单元将很可能也是不可检索的。因此，通常更好的做法是让客户机等待，直到数据可用为止，以便为用户提供更好的重放体验。

在确保特定数据单元是可重新检索的之后，即，

$$\sum_i a_i \geq k \qquad \text{公式 10}$$

客户机检验每个对等体的请求队列中的可用空间。需要将每个对等体的 RFT 保持在系统常数 T_{rft} 左右。在一个测试实施例中，1.0s 数量级的 T_{rft} 提供较好结果。（注意，使用太短的请求队列可能不会有效地利用从客户机到对等体的带宽。）

具体地，如果客户机所发送的请求数据包被丢失或被延迟，那么，服务对等体可能没有什么东西要发送，这会浪费其服务带宽。相反，使用过长的请求队列可以防止客户机迅速适应变化，例如，对等体之一的断开。另外，如果请求队列对于所有的对等体在 RFT 方面都是相同的长度，该请求队列的容量变得与其服务带宽成比例：$Th_i \cdot T_{rft}$。

例如，假设 T_{rft} 是 1.0s，那么，具有服务带宽 16kbps 的对等体允许其请求队列中有 2KB 的无法履行的请求待决，而具有服务带宽 1Mbps 的对等体允许 128KB 的无法履行的请求待决。因此，可以向特定对等体请求的擦除编码块的数量由其请求队列中留出的空间来定上限：

$$e_i = \min(a_i, (Th_i \cdot T_{rft} - B_{i,outstanding})/bk) \qquad \text{公式 11}$$

其中，e_i 是可以向对等体 i 请求的擦除编码块的数量，bk 是擦除编码块的大小。

公式 11 保证客户机从不发出具有大于 T_{rft} 的预期的 RFT 的请求。如果客户机无法找到足够的当前可用的擦除编码块，即，

$$\sum_i e_i < k \qquad \text{公式 12}$$

它将等待，指导服务对等体的请求队列清除为止。当 $\sum_i e_i \geq k$ 时，数据单元请求只是被形成并被发送到对等体。向某个对等体请求的块的实际数目（b_i）被计算如下：

$$\begin{cases} \sum_i b_i = k, \\ b_i = \min(e_i, c \cdot Th_i), \end{cases} \qquad \text{公式 13}$$

其中，c 是满足 $\sum_i b_i = k$ 的常数。

一般而言，以上略述的过程与其服务带宽 Th_i 成比例地将服务负载分配给每个对等体（公式 13）。它也确保客户机不会向特定服务对等体请求比该服务对等体实际上所高速缓存或存储的更多的块。最后，如公式 11 所示的，该过程也确保请求的 RFT 不超过 T_{rft}。

请求队列的第二个功能是识别由每个服务对等体发回的内容。如上所述，PeerStreamer 客户机和对等体通过 TCP 来进行通信，它保存数据传输的顺序，并保证数据包传递。而且，每个对等体依次处理传入的请求。结果，不需要明确地识别被发回的内容，因为它必须赞成每个对等体的请求队列中待决的第一个请求。

对于上述的请求队列的第三个功能，该请求队列也用来使断开的对等体的各个请求重定向。例如，只要特定的服务对等体与客户机断开，该断开事件都由 TCP 协议来获得，TCP 协议随后将该断开报告给客户机。然后，客户机将断开的对等体的队列中待决的所有无法履行的请求再分配给剩余的对等体中的一个或多个。用于再分配请求的过程十分类似于用于分配第一个地方的请求的过程。唯一的例外是：在请求再分配时，必须考虑已经向断开的对等体请求的块的数目。

最后，只要擦除编码块到达客户机处，它们都立即离开 TCP 套接字。在将到达的内容与待决的请求配对之后，从请求队列中除去已履行的请求。然后，所识别的擦除编码块被放入分级队列。结果，分级队列的大小增加。如果分级队列达到预定大小 $T_{staging}$，那么，不发送媒体数据包/数据单元的更多请求。一旦已接收某个数据单元的所有擦除编码块，该数据单元就被擦除解码，并被标记为“就绪”。如果所有其所请求的数据单元就绪，那么，媒体数据包变成就绪。音频/视频解码器周期性地从分级队列中除去“就绪”的媒体数据包。这减小分级队列的大小，并可能会触发新媒体数据包请求的生成。

以上所述的媒体流传送操作随后继续进行，直到完成媒体文件的重放，或直到诸如没有足够的对等体可用来流传送媒体等时刻，或用户终止流传送会话。

3.3 PeerStreamer 操作：

以上根据图 2 至图 9 描述的过程由图 10 中的通用操作流程图来示出。一般而言，图 10 展示了示出 PeerStreamer 的几个操作实施例的示例性操作流程图。应该注意，由图 10 中的折线或虚线来表示的任何方框和方框之间的互连表示这里所描述的 PeerStreamer 的替换实施例；并且，如下所述，可以结合在整个文档中描述的其他替换实施例来使用任何或所有这些替换实施例。

具体地，如图 10 所示，在媒体流传送操作之前，服务器 200（也可能是对等体 220 之一）对将要流传送的媒体进行编码（1000）。如上所述，PeerStreamer 能够利用许多常规编解码器（例如，MPEG 1/2/4、WMA、WMV 等）中的任何一种来进行操作。此外，在编码过程 1000 期间，服务器 200 也生成前述的媒体头部和包含媒体结构的伴随文件。

如上所述，在一个实施例中，一旦媒体被编码（1000），所编码的媒体数据包就被分成（1005）多个固定大小的数据单元。另外，对于所编码的媒体，媒体头部和媒体结构也被分成（1005）多个与用来分割已编码媒体数据包的相同的固定大小的数据单元。如以上所解释的，通过将信息分成（1005）固定长度数据单元，来允许客户机和服务对等体在媒体流传送操作之前预先分配存储块，从而在流传送过程期间避免计算上昂贵的存储器分配操作。另外，较小的数据单元的运用允许客户机对每个服务对等体所消耗的确切的带宽数量的较精细的控制，以便在流传送操作期间满足客户机数据单元请求。

除了将已编码媒体、媒体头部和媒体结构分成（1005）较小的数据单元以外，在一个实施例中，附加编码层还用来在服务对等体本质上不可靠的典型 P2P 环境中提供增加的冗余度。具体地，如上所述，在一个实施例中，使用基于关键字的高速率擦除弹性编码过程 1010，来将这些数据单元进一步分成多个数据块。

这类编码 1010 的运用确保对等体中的一个或多个将具有重建特定的数据单元所必要数据块，同时，简化客户机上识别对等体中的哪一个包含必要数据的要求。另外，如上所述，在一个实施例中，每个服务对等体 220 所使用的擦除弹性编码关键字被服务器 200 自动分配给每个对等体。但是，在另一个实施例中，每个服务对等体 220 仅仅随机地选择擦除弹性编码关键字。然后，这些关键字连同前述的可用性向量一起被包括在内；当每个对等体 220 最初由客户机来联系时，可用性向量由客户机 210 来检索。在随机关键字实施例中，在对给定数据单元存在关键字冲突的情况下，客户机随后使一个或多个对等体的关键字无效。

一旦媒体最初已被编码（1000）、被分成数据单元（1005）并可能进一步被擦除编码（1010），最后得到的数据单元或数据块随后就被分发（1015）给各个服务对等体 220。该分发（1015）会是深思熟虑的，这体现在已编码媒体的块或数据包仅仅被全部或部分地提供给多个对等体，它随后被高速缓存或存储在那里，用于当被与 P2P 网络连接的客户机调用时的进一步的流传送操作。

或者，如上所述，只要客户机 210 流传送特定的媒体文件，恢复的媒体数据

包都只是编码操作 1000 之后的媒体数据包。它们可以被分成数据单元（1005），并可被进一步擦除编码（1010）；并且，客户机可能在本地存储器或存储内维持流传送到它那里的内容的至少一部分。然后，客户机被识别为服务对等体 220（在前述对等体列表 310 中），用于将来的流传送操作。这个实施例的一个优点是：包含特定媒体文件的各个部分的对等体的数目最初很低，从而增加服务器本身上的要求，以满足服务请求，但随着时间的流逝并且更多客户机流传送该媒体，那些客户机随后将能够担当用于以后的流传送请求的对等体。因此，不需要明确地选择服务对等体 220，以保持将要被流传送的媒体的全部或一部分的初始高速缓存。结果，对于试图识别愿意接受将要被流传送的媒体的初始高速缓存的对等体，进一步减少服务器上的任何要求。

在任何情况下，一旦媒体已被分发（1015）给服务对等体 220，客户机 210 随后就准备好开始将请求流传送到那些服务对等体。另外，如上所述，出于流传送到客户机 210 的目的，服务器 200 也可以担当服务对等体 220。再有，鉴于以上讨论，应该清楚，随着时间的流逝，特定媒体文件的初始流传送可以要求更大的服务器 200 牵连，并且，更多客户机 210 流传送该媒体（并且随后可担当服务对等体），实际上担当服务对等体的服务器上的各个要求被减少或甚至被消除。

这时，客户机 210 通过首先检索可用服务对等体 220 的列表 310，来开始流传送会话。如上所述，该列表 310 直接从服务器 200 中、从对等体 220 之一中、或通过使用用于识别潜在服务对等体的常规 DHT 方法 315 来检索。一旦客户机 210 检索了对等体列表 310，该客户机随后就连接到每个服务对等体 220，并从每个对等体中检索（1025）可用性向量。另外，在一个实施例中，客户机 210 在正在进行的流传送操作期间周期性核对对于对等体列表 310 的更新（1030）。执行这类周期性核对（1030）的一个优点是：在大型 P2P 网络中，多个服务对等体任何给定的时刻正在联机和脱机是可能的。因此，确保客户机 210 具有已更新的对等体列表 310 将允许该客户机响应于当前正将媒体流传送到该客户机的对等体 220 的衰减或降级。只要列表 310 的周期性核对（1030）指出对该列表添加新的对等体 220，客户机 210 就再次连接到该新的对等体，并检索（1025）该新的对等体的可用性向量。

一旦客户机 210 检索（1025）了每个对等体 220 的可用性向量，该客户机随后就通过经由该客户机与那些对等体之间的网络连接而请求对应于来自这些对等体中的一个或多个的信息的数据单元，来检索（1035）将要从这些服务对等体中的一个或多个流传送的媒体的媒体头部和媒体结构。

如上所述，媒体头部通常包含描述该媒体的全局信息，例如，媒体中的通道数目、每个通道的属性和特征（音频采样率、视频分辨率/帧速率）、所使用的编解码器、媒体的作者/版权持有者、等等。因此，媒体流传送会话的起始处的媒体头部的检索允许客户机 220 对这些必要的工具进行设置或初始化（1040），以便在该流传送会话期间接收那些数据包之前对随后接收的数据包进行解码（1070）和呈现（1075）。

另外，在检索（1035）特定流媒体的媒体结构之后，客户机分析该媒体结构，并计算在流传送过程期间将需要被请求的流媒体的数据单元的数据单元 ID（1045）。然后，客户机 210 向服务对等体 220 中的一个或多个逐个地请求那些数据单元（1050）。

另外，如上所述，在结合编码关键字的随机对等体选择来使用擦除编码的实施例中，客户机 210 将使对等体 220 中的一个或多个上的重复关键字无效，以便管理关键字冲突（1055）。在一个相关实施例中，PeerStreamer 使用嵌入编码媒体，并且，随后根据可用的服务带宽和客户机 210 队列状态来管理（1060）每个对等体 220 的数据请求（和流传送比特率）。在此情况下，数据单元的正在进行的请求（1050)对应于将基于各个服务对等体的可用带宽来提供最小速率失真的那些数据包。在任何情况下，如上所述，都根据是使用嵌入还是非嵌入编码、对等体的连接状态、以及剩下来用于请求和接收缺少的或新近的数据单元的时间，来再次向同一或另一对等体 220 请求（1050）缺少的或新近的数据单元。

最后，一旦根据客户机 220 请求（1050）检索了构成特定媒体数据包的所有数据单元，那些数据包就被重新汇编（1065）成原始媒体数据包。然后，重新汇编的媒体数据包被解码（1070）、被呈现（1075）和被提供，用于常规的显示设备 355 或扬声器 260 中的任何一个或两者上的重放。

已出于举例说明和描述的目的来呈现对 PeerStreamer 的前述描述。它并不意在做到详尽或将本发明局限于所揭示的精确形式。按照以上的教导，许多修改和变更是可能的。另外，应该注意，可以在形成 PeerStreamer 的其它混合式实施例所需要的任何组合中使用任何或所有这些上述的替换实施例。本发明的范围意在不受到本详细描述的限制，而是受到所附权利要求书的限制。

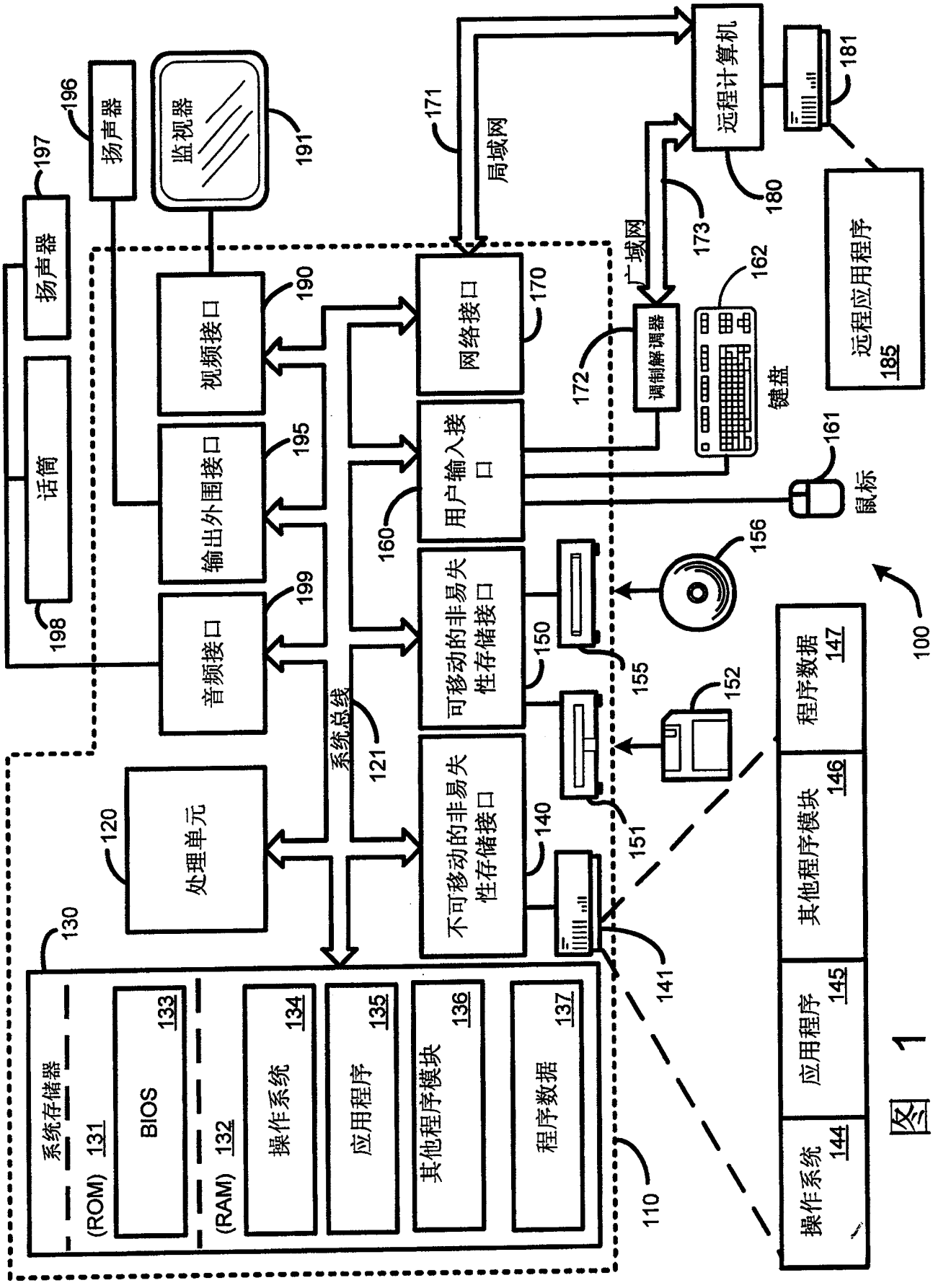

系统存储器
130
(ROM) 131
BIOS
133
(RAM) 132
操作系统
134
应用程序
135
其他程序模块
136
程序数据
137
处理单元
120
系统总线
121
音频接口
199
输出外围接口
195
视频接口
190
网络接口
170
用户输入接口
160
可移动的非易失性存储接口
150
不可移动的非易失性存储接口
140
扬声器
196
监视器
191
话筒
198
扬声器
197
局域网
171
远程计算机
180
远程应用程序
185
调制解调器
172
广域网
173
键盘
162
鼠标
161
操作系统
144
应用程序
145
其他程序模块
146
程序数据
147
110
141
151
152
155
156
100

图 1

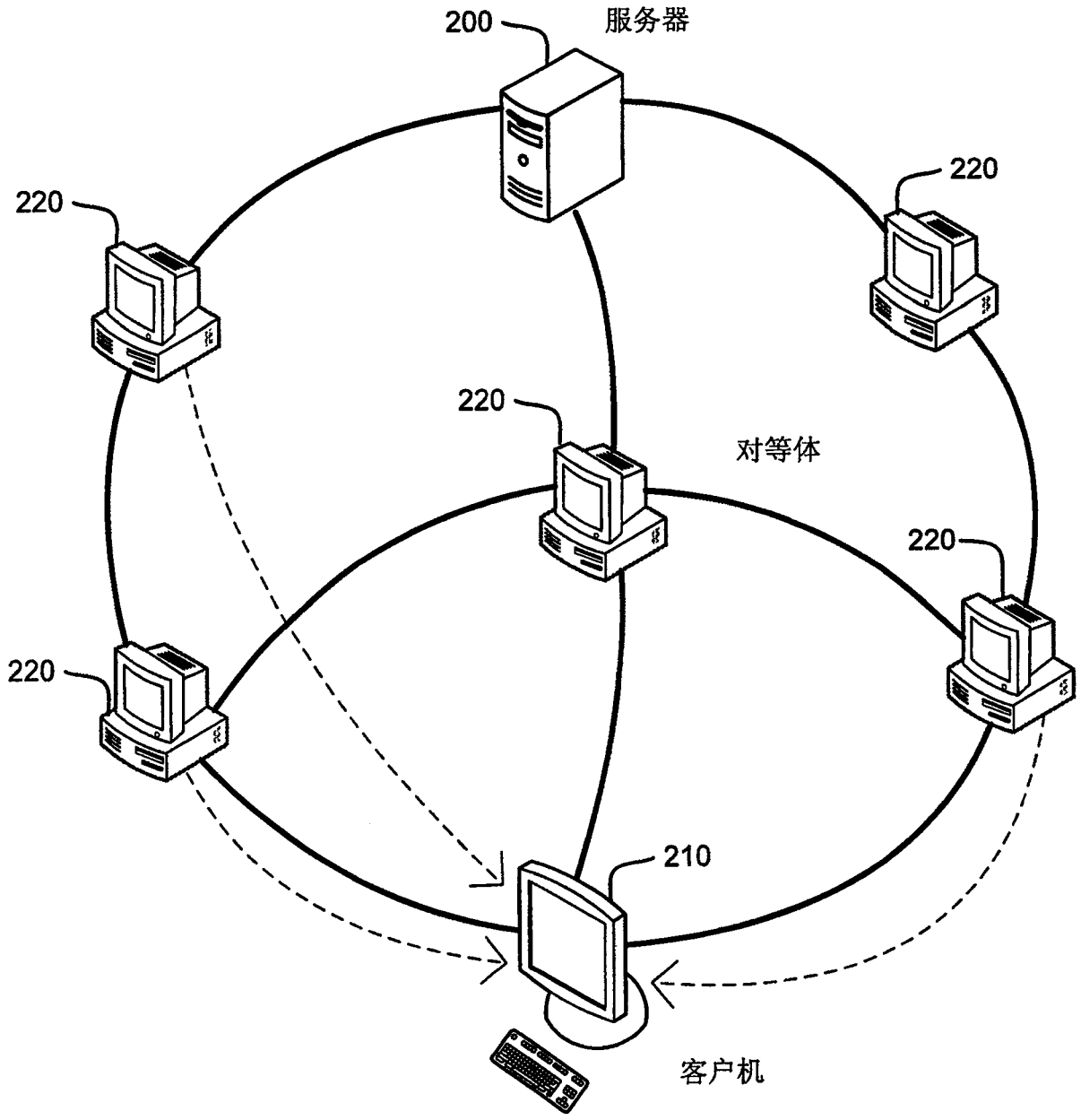


图 2

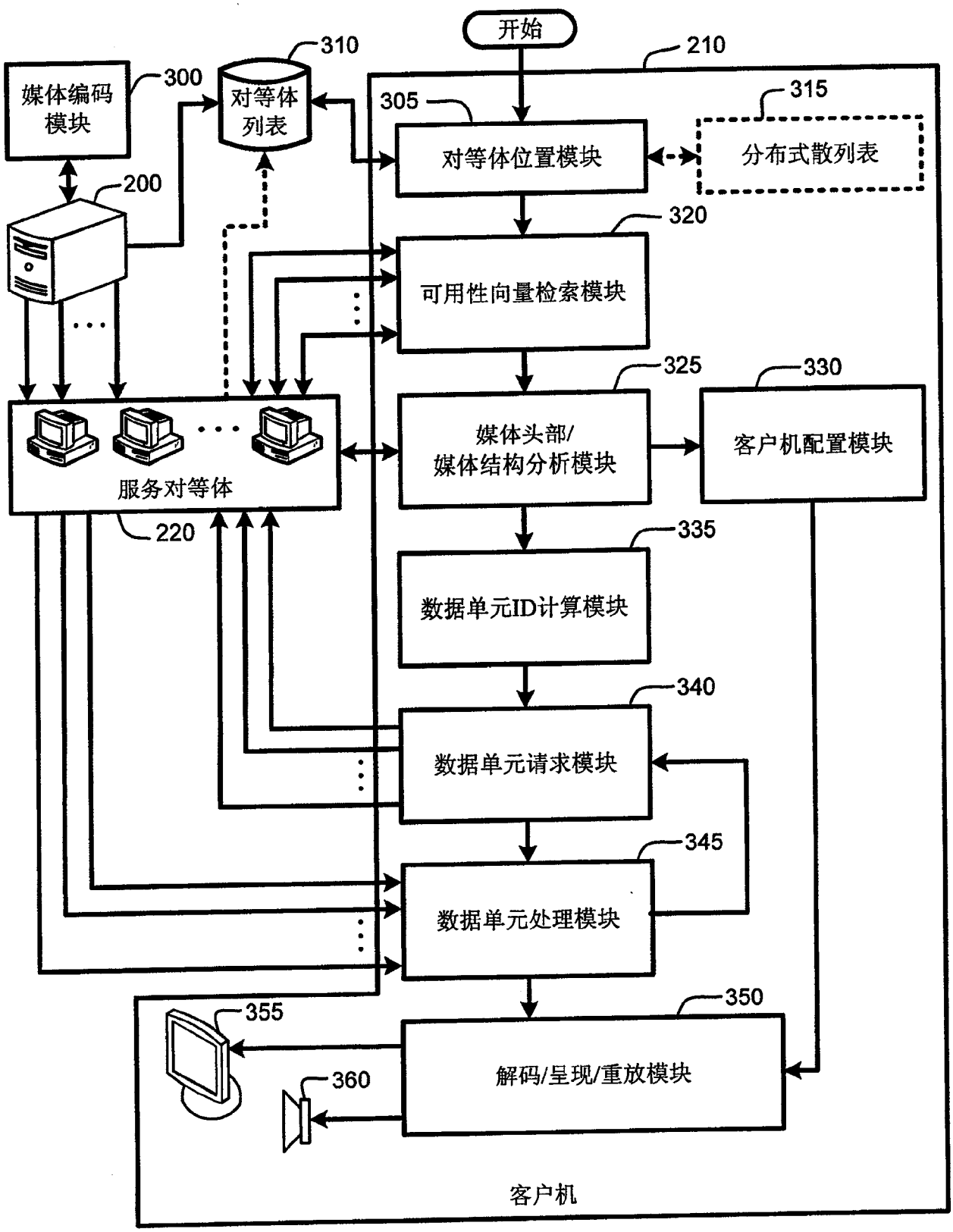

开始
210
310
媒体编码模块
300
对等体列表
305
315
对等体位置模块
分布式散列表
200
320
可用性向量检索模块
325
330
媒体头部/媒体结构分析模块
客户机配置模块
服务对等体
220
335
数据单元ID计算模块
340
数据单元请求模块
345
数据单元处理模块
350
355
解码/呈现/重放模块
360
客户机

图 3

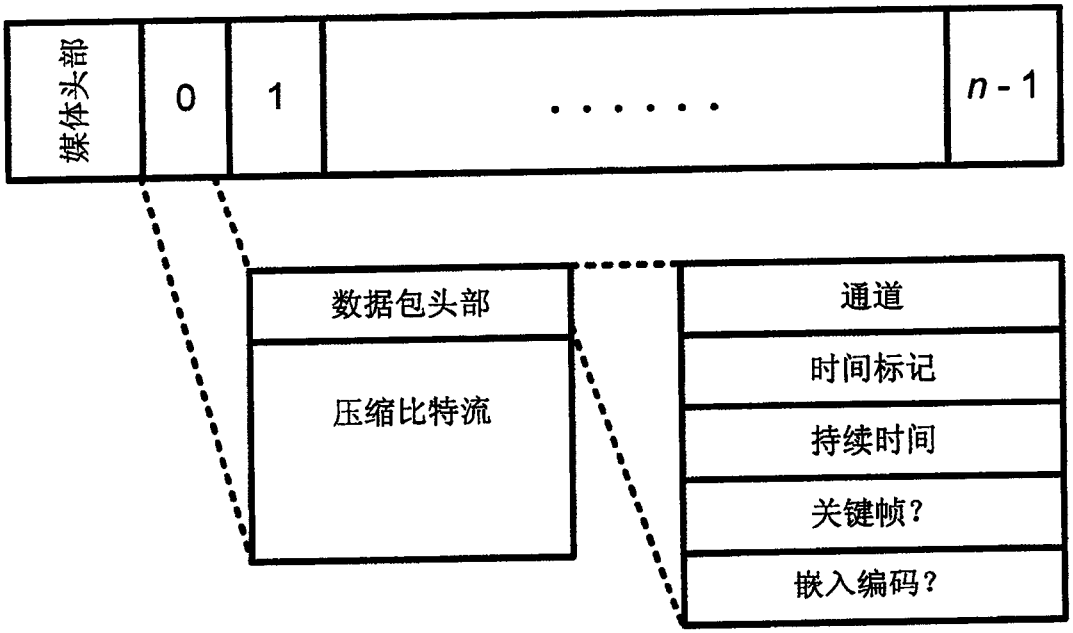

媒体头部
0
1
. . . . . .
n - 1
数据包头部
压缩比特流
通道
时间标记
持续时间
关键帧?
嵌入编码?

图 4

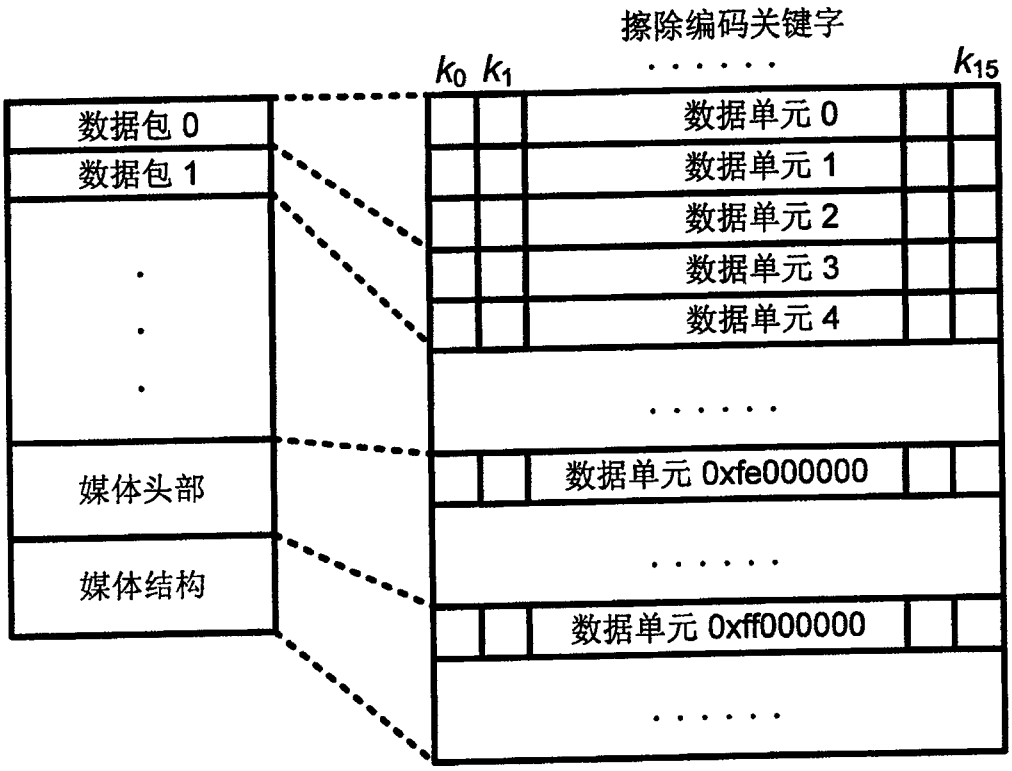

擦除编码关键字
k0 k1
. . . . . .
k15
数据包 0
数据包 1
媒体头部
媒体结构
数据单元 0
数据单元 1
数据单元 2
数据单元 3
数据单元 4
. . . . . .
数据单元 0xfe000000
. . . . . .
数据单元 0xff000000
. . . . . .

图 5

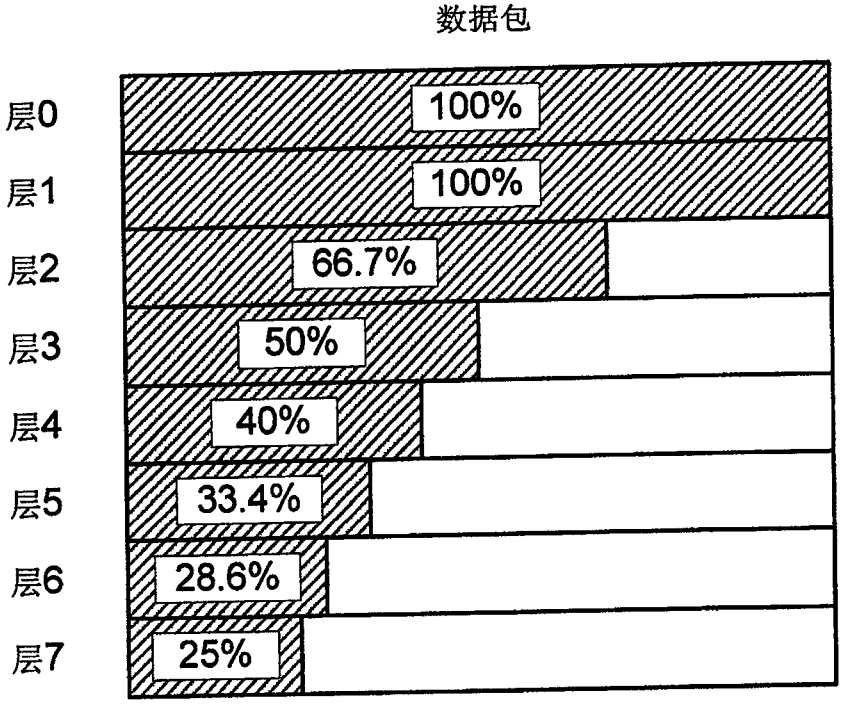

数据包
层0
层1
层2
层3
层4
层5
层6
层7
100%
100%
66.7%
50%
40%
33.4%
28.6%
25%

图 6

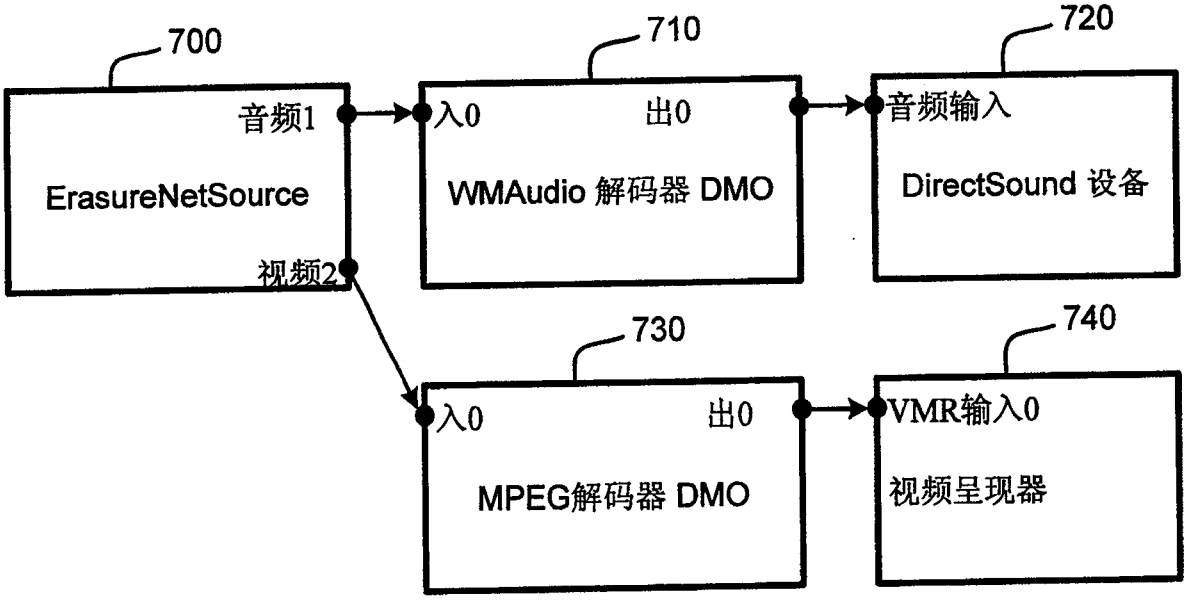

700
音频1
ErasureNetSource
视频2
710
入0
出0
WMAudio 解码器 DMO
720
音频输入
DirectSound 设备
730
入0
出0
MPEG解码器 DMO
740
VMR输入0
视频呈现器

图 7

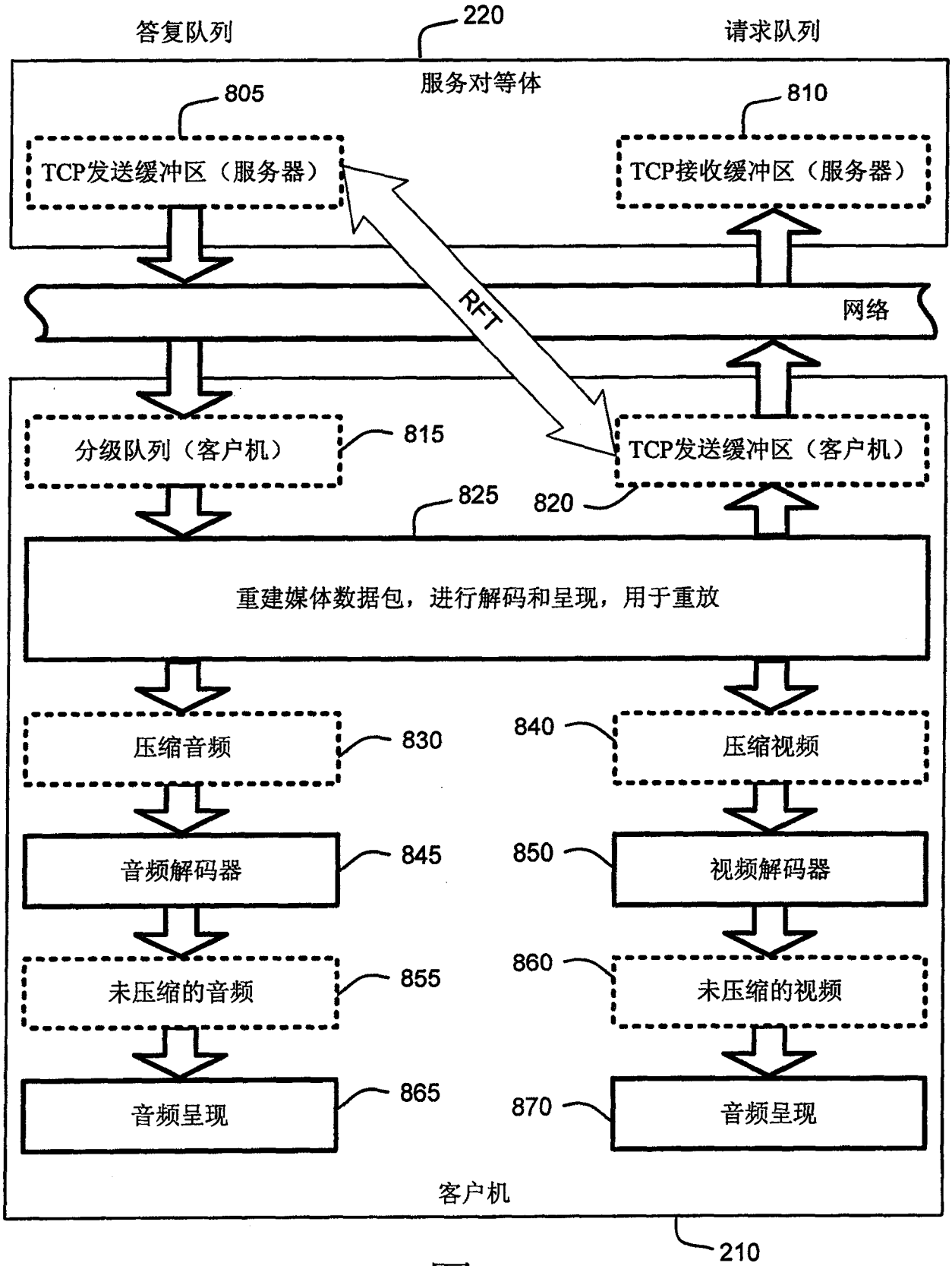

答复队列
220
请求队列
服务对等体
805
TCP发送缓冲区（服务器）
810
TCP接收缓冲区（服务器）
RFT
网络
分级队列（客户机）
815
TCP发送缓冲区（客户机）
825
820
重建媒体数据包，进行解码和呈现，用于重放
压缩音频
830
840
压缩视频
音频解码器
845
850
视频解码器
未压缩的音频
855
860
未压缩的视频
音频呈现
865
870
音频呈现
客户机
210

图 8

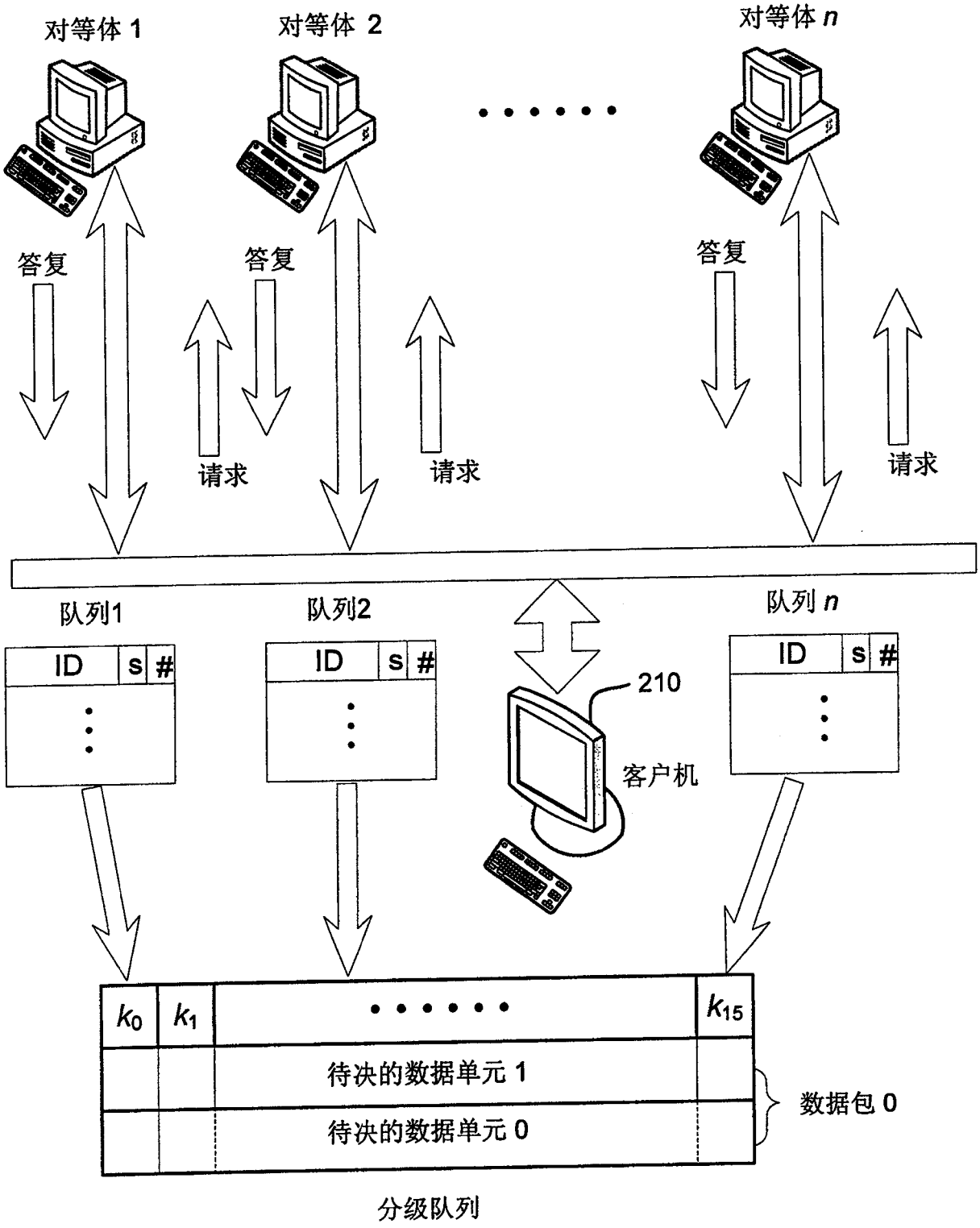

对等体 1
对等体 2
对等体 n
答复
答复
答复
请求
请求
请求
队列1
队列2
队列 n
ID
s
#
210
客户机
k_0
k_1
k_{15}
待决的数据单元 1
待决的数据单元 0
数据包 0
分级队列

图 9

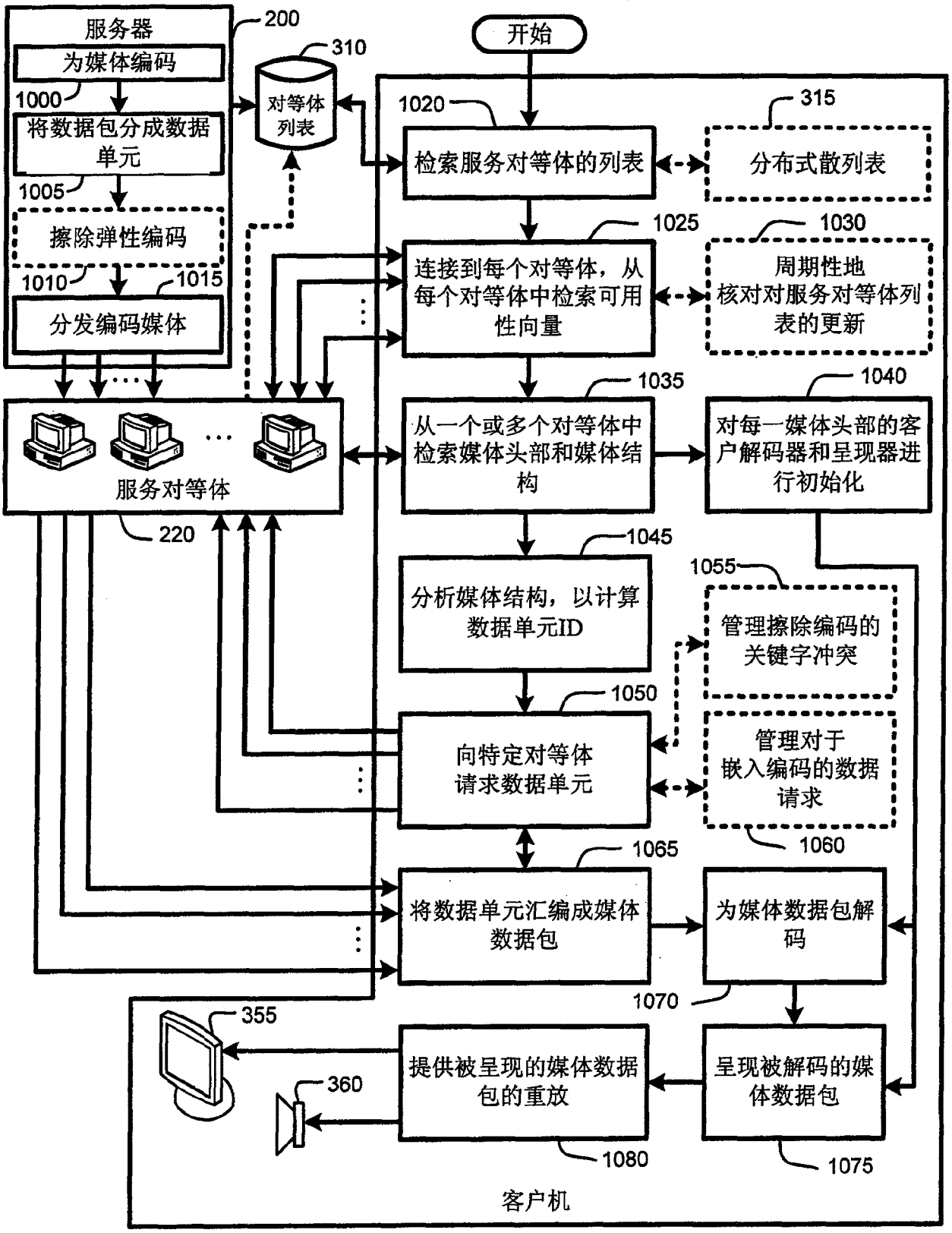

服务器
200
为媒体编码
1000
将数据包分成数据单元
1005
擦除弹性编码
1010
1015
分发编码媒体
310
对等体列表
服务对等体
220
开始
1020
检索服务对等体的列表
315
分布式散列表
1025
连接到每个对等体，从每个对等体中检索可用性向量
1030
周期性地核对对服务对等体列表的更新
1035
从一个或多个对等体中检索媒体头部和媒体结构
1040
对每一媒体头部的客户解码器和呈现器进行初始化
1045
分析媒体结构，以计算数据单元ID
1055
管理擦除编码的关键字冲突
1050
向特定对等体请求数据单元
管理对于嵌入编码的数据请求
1060
1065
将数据单元汇编成媒体数据包
为媒体数据包解码
1070
355
360
提供被呈现的媒体数据包的重放
呈现被解码的媒体数据包
1080
1075
客户机

图 10