

(12) 发明专利申请

(10) 申请公布号 CN 102483782 A

(43) 申请公布日 2012. 05. 30

(21) 申请号 201080038225. 6

(51) Int. Cl.

(22) 申请日 2010. 07. 20

G06F 21/00 (2006. 01)

(30) 优先权数据

12/509, 401 2009. 07. 24 US

(85) PCT申请进入国家阶段日

2012. 02. 28

(86) PCT申请的申请数据

PCT/US2010/042606 2010. 07. 20

(87) PCT申请的公布数据

W02011/011414 EN 2011. 01. 27

(71) 申请人 苹果公司

地址 美国加利福尼亚

(72) 发明人 J·佳隆 P·德赫比蒙特

J-P·修达德

(74) 专利代理机构 中国国际贸易促进委员会专
利商标事务所 11038

代理人 鲍进

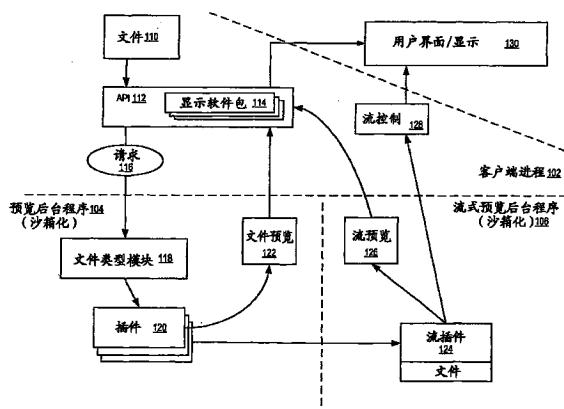
权利要求书 2 页 说明书 6 页 附图 4 页

(54) 发明名称

动态媒体内容预览

(57) 摘要

本公开涉及动态媒体内容预览。沙箱化进程响应于对于动态媒体内容的预览的请求而被启动。由在所述沙箱化进程内执行的插件以预览特定格式产生帧流。所述帧流被提供作为所述动态媒体内容的预览，其中所述预览在显示屏上是可见的。



1. 一种方法,包括 :

响应于对于动态媒体内容的预览的请求,启动沙箱化进程;

以预览特定格式产生帧流,所述帧流是由在所述沙箱化进程内执行的插件产生的;以及

提供所述帧流作为所述动态媒体内容的预览,所述预览在显示屏上是可见的。

2. 按照权利要求 1 所述的方法,其中所述启动沙箱化进程还包括:

确定所述动态媒体内容的内容类型;以及

取回流插件以至少部分地基于所述动态媒体内容的所述内容类型而将所述动态媒体内容转换成所述预览特定格式。

3. 按照权利要求 2 所述的方法,还包括:

提供所述流插件和所述动态媒体内容给所述沙箱化进程。

4. 按照权利要求 1 所述的方法,其中所述产生还包括:

为所述帧流中的每个帧单独地产生音频分量和视频分量。

5. 按照权利要求 1 所述的方法,其中所述提供所述帧流还包括:

在每一帧被产生的同时逐帧地提供所述帧流以供显示;以及

在动态预览的显示期间,提供一种或多种控制以帮助用户控制所述动态媒体内容。

6. 按照权利要求 1 所述的方法,其中所述插件与应用程序相关联,并且其中在不发起所述应用程序的情况下所述预览是可见的。

7. 一种方法,包括:

第一沙箱化进程从客户端进程接收对于动态媒体内容的预览的请求;

所述第一沙箱化进程响应于接收到所述请求而启动第二沙箱化进程;

以预览特定格式产生帧流,所述帧流是由在所述第二沙箱化进程内执行的插件产生的;以及

提供所述帧流给所述客户端进程以显示在显示屏上。

8. 按照权利要求 7 所述的方法,其中所述启动第二沙箱化进程还包括:

确定所述动态媒体内容的内容类型;

确定所述内容类型是非原有内容类型;以及

取回流插件以至少部分地基于所述动态媒体内容的所述非原有内容类型而将所述动态媒体内容转换成所述预览特定格式。

9. 按照权利要求 8 所述的方法,还包括:

提供所述流插件和所述动态媒体内容给所述第二沙箱化进程。

10. 按照权利要求 7 所述的方法,其中所述产生还包括:

为所述帧流中的每个帧单独地产生音频分量和视频分量。

11. 按照权利要求 7 所述的方法,其中所述提供所述帧流还包括:

在每一帧被产生的同时逐帧地提供所述帧流给所述客户端进程以供显示;以及

在动态预览的显示期间,提供一种或多种控制给所述客户端进程以帮助用户控制所述动态媒体内容。

12. 按照权利要求 7 所述的方法,其中所述插件与应用程序相关联,并且其中在不发起所述应用程序的情况下所述预览是可见的。

13. 一种其上存储有指令的计算机可读存储介质,所述指令在被执行时使得计算机 :
响应于对于动态媒体内容的预览的请求,启动沙箱化进程 ;
以预览特定格式产生帧流,所述帧流是由在所述沙箱化进程中执行的插件产生的 ;以及
提供所述帧流作为所述动态媒体内容的预览,所述预览在显示屏上是可见的。
14. 按照权利要求 13 所述的计算机可读存储介质,其中使得启动沙箱化进程的指令还包括使得所述计算机确定所述动态媒体内容的内容类型以及取回流插件以至少部分地基于所述动态媒体内容的所述内容类型而将所述动态媒体内容转换成所述预览特定格式的指令。
15. 按照权利要求 14 所述的计算机可读存储介质,还包括使得所述计算机提供所述流插件和所述动态媒体内容给所述沙箱化进程的指令。
16. 按照权利要求 13 所述的计算机可读存储介质,其中使得产生的指令还包括使得所述计算机为所述帧流中的每个帧单独地产生音频分量和视频分量的指令。
17. 按照权利要求 13 所述的计算机可读存储介质,其中使得提供所述帧流的指令还包括使得所述计算机在每一帧被产生的同时逐帧地提供所述帧流以供显示以及在动态预览的显示期间,提供一种或多种控制以帮助用户控制所述动态媒体内容的指令。
18. 一种设备,包括 :
用于响应于对于动态媒体内容的预览的请求而启动沙箱化进程以及以预览特定格式产生帧流的处理装置,所述帧流是由在所述沙箱化进程中执行的插件产生的 ;以及
用于将所产生的帧流显示为所述动态媒体内容的预览的显示器。
19. 按照权利要求 18 所述的设备,所述处理装置还用于确定所述动态媒体内容的内容类型,以及取回流插件以至少部分地基于所述动态媒体内容的所述内容类型而将所述动态媒体内容转换成所述预览特定格式。
20. 按照权利要求 18 所述的设备,所述处理装置还用于为所述帧流中的每个帧单独地产生音频分量和视频分量。
21. 按照权利要求 18 所述的设备,所述处理装置还用于在每一帧被产生的同时逐帧地提供所述帧流以供显示,以及在动态预览的显示期间,提供一个或多个用户界面控件以帮助用户控制所述动态媒体内容。
22. 按照权利要求 18 所述的设备,其中所述插件与应用程序相关联,并且其中在不发起所述应用程序的情况下所述预览是可见的。

动态媒体内容预览

技术领域

[0001] 本发明的实施例涉及流式媒体 (streaming media), 尤其是涉及提供流式媒体和数字媒体内容的安全预览。

背景技术

[0002] 随着计算机用户变得越来越见多识广, 对于人们使用的计算机程序的高效性的需求也在不断提高。例如, 很多计算机系统具有某一类型的文件管理系统, 该系统管理用户可能需要存取的数百甚或数千个文件。在许多常规的文件管理系统中, 难以使用与该文件的文件类型相关联的必需应用程序在无需实际打开该文件的情况下确定特定文件的内容。时常, 在打开文件之前用户可以获得的关于该文件的有用信息仅仅是文件名。很多用户希望能够在将资源实际用于打开完全应用程序以及随后打开该文件之前预览该文件的内容。某些文件管理系统, 诸如由 Cupertino, CA 的 Apple Inc. 提供的 Mac OS X 中的 Finder, 提供了用于在打开文件之前预览文件的功能。其它操作系统和文件管理系统也可以提供某些静态预览功能。

[0003] 例如, 使用 Apple 的 Finder, 用户可以预览文档, 诸如 PDF、文本文件、字处理文档和电子数据表等。Finder 也允许用户预览具有 Mac OS 识别出的原有文件类型的某些视频内容。

[0004] Mac OS 中的 Quick Look 后台程序 (daemon) 包括多种插件, 每个插件接受来自客户端进程的特定数据类型并将其转换成具有一组标准格式——诸如 PDF 和 HTML——中的一种格式的预览。随后, 由客户端进程使用 Quick Look 框架来显示标准格式的预览。这种结构的优点在于它避免由第三方插件崩溃而导致的客户端应用程序崩溃。但是, 这种模型对于诸如流式媒体的动态媒体并不能很好地工作, 原因在于将与非原有文件类型相关联的动态媒体转换成适当的标准格式花费太多的处理时间而使其不可用或不可靠。

发明内容

[0005] 当客户端进程接收到对于动态媒体内容的预览的请求时, 该请求被传递给沙箱化进程。沙箱化进程与客户端进程隔离, 这意味着如果沙箱化进程崩溃, 客户端进程保持功能性和可操作性。在沙箱化进程中, 确定动态媒体内容的类型。内容类型可以基于例如统一类型标识符 (UTI) 来确定。一旦内容类型被确定, 沙箱化进程就取回流化插件 (streaming plug-in) 以将动态媒体内容转换成客户端进程可以用来显示动态媒体的预览的媒体类型。

[0006] 插件被用来根据与该插件相关联的协议转换动态媒体内容。在一些实施例中, 插件产生音频分量和视频分量二者 (至少为具有音频和视频二者的动态媒体内容产生这二者)。随着动态内容的每一帧被产生, 其被传递给客户端进程以显示为流式预览。在一些实施例中, 插件给客户端进程提供允许用户控制动态媒体内容的预览的有限方面 (例如, “停止”、“播放”和“暂停”等) 的一种或多种控制。

附图说明

[0007] 以下说明包括对于附图的讨论，所述附图图解说明了本发明实施例的实施方式的例子。附图只是出于举例说明而非限制的目的提供的。本文中对于一个或多个“实施例”的提及要被理解为描述包括在本发明的至少一种实施方式中的特定特征、结构或特性。因此，诸如出现在本文中的“在一个实施例中”或“在替换实施例中”之类的短语描述了本发明的各种实施例和实施方式，而无须全部指的是同一实施例。但是，它们也无须互相排斥。

[0008] 图 1 是图示按照各种实施例的系统的框图。

[0009] 图 2 是图示按照各种实施例的系统的框图。

[0010] 图 3 是按照各种实施例的系统中的操作的流程图。

[0011] 图 4 是图示按照各种实施例的系统的框图。

具体实施方式

[0012] 本文中提供的各种方法、设备和系统使得能够预览动态媒体内容。当用户选择文件（例如，从文件管理系统中选择）并请求该文件的预览时，本文所描述的实施例有助于将该动态媒体转换为适合于预览该动态媒体内容的格式。

[0013] 图 1 是按照各种实施例的框图。如图所示，文件管理系统（例如，Finder、Windows Explorer 等）或其它程序（例如，客户端应用程序、网络浏览器等）包括客户端进程 102。客户端进程 102 是与特定程序的用户级交互发生之处。当用户请求文件 110 的预览时，应用编程接口（API）112 发送请求 116 给预览后台程序 104。如果预览后台程序 104 尚未运行，则其被启动。在各种实施例中，预览后台程序 104 是沙箱化进程。沙箱化进程是与其它运行的进程或程序分离开的进程。沙箱化进程，或简称为沙箱，通常为访客程序提供受到紧密控制的资源集合以便运行在诸如盘和存储器上的暂存空间（scratch space）中。在各种实施例中，在沙箱中通常不允许或严格限制网络访问、检查主机系统的能力或从输入设备中读取。

[0014] 通过发送请求 116 给预览后台程序 104，导致预览后台程序 104 发生任何错误 / 问题（例如，崩溃、挂起、不可接受的等待时间等）的对文件 104 的任何操纵或转换将不影响客户端进程 102。换言之，如果预览后台程序 104 崩溃，客户端进程 102 将继续不受影响地运行。对于用户，这意味着在产生文件的预览中发生的任何错误将不会阻止用户使用底层程序，诸如文件管理系统。如果客户端进程 102 与诸如文件管理系统（例如，Apple Finder、Windows Explorer 等）之类的基本用户程序相关联，则这可能是特别重要的。在一些实施例中，如果沙箱化进程崩溃，则客户端进程可能显示所请求的预览当前不可用的指示。

[0015] 在各种实施例中，文件类型模块 118 初始接收来自客户端进程 112 的请求 116。文件类型模块 118 确定与文件 110 相关联的文件类型。如前所述，文件类型可以是使用统一类型标识符或 UTI 确定的，UTI 是由 Apple Inc. 定义的、唯一地标识一类项目的类型的字符串。因此，UTI 被用来标识文件和文件夹、剪切板数据、软件包（bundle）、混叠（alias）、符号链接（symlink）和流式数据的类型。如果文件类型模块 118 识别出文件 110 的文件类型为原有文件类型，则文件类型模块 118 取回与该原有文件类型相关联的已知插件 120。对于包含静态媒体的文件，诸如字处理文档、PDF、HTML 文件等，所取回的插件可以处理文件请求，并将其转换成文件预览 122，其被发送回客户端进程 102 以显示为预览。在某些实施例

中,插件 120 直接能够转换动态媒体内容(音频/视频文件、交互式 3D 动画文件等),这将在下面详述。被发送给客户端进程 102 的文件预览 122 由 API 112 处理,API 112 包括各种显示软件包 114。每个显示软件包包括显示一种或多种特定文件类型的预览所需的资源。因此,如果文件预览 122 是 PDF 的预览,则来自该组显示软件包 114 的 PDF 显示软件包获得 PDF 预览并使其准备好在显示器 130 上显示。

[0016] 本文描述的其它实施例能够提供动态媒体内容(例如,流式媒体、视频、3-D 动画等)的预览。例如,如果文件 110 包括动态媒体内容,则请求 116 可能被发送给预览后台程序 104,并且文件类型模块 118 基于例如与文件 110 相关联的 UTI 来确定文件类型。其它标识符可能用在不同的实施例中。如果动态媒体具有原有文件类型并被文件类型模块 118 识别出,则对应的插件 120 被取回以转换文件 110 和提供预览。但是,如果文件 110 包括非原有文件类型的动态媒体内容,则文件类型模块 118 取回指定插件 120 以处理文件 110 的非原有转换。

[0017] 在某些实施例中,指定插件 120 负责取回额外插件(流插件 124)以处理动态媒体内容。指定插件也可以取回需要被转换的文件的复本。在一些实施例中,该文件是作为请求 116 的一部分被接收的同一文件。但是,在其它实施例中,该文件可以是要在转换成流式预览中使用的同一文件的不同复本。

[0018] 此外,为流插件 124 打开或启动另一沙箱化进程 106 以将文件 110 转换成流式预览 126。以此方式,流式预览后台程序 106 与预览后台程序 104 和客户端进程 102 隔离开。因此,如果在文件到流式预览的转换期间流插件 124 崩溃,则进程 106、预览后台程序 104 和客户端进程 102 仍不受影响。以此方式,用户不会丢失客户端应用程序(例如,文件管理系统)中的任何功能,而且用户仍可以在没有任何中断的情况下预览其它文件,尤其是不需要流式预览的那些文件。

[0019] 在各种实施例中,流预览 126 是从原始文件 110 转换成的一系列帧。与流预览 126 相关联的帧可以包括有限的用户功能。换言之,如果完全的应用程序(例如,媒体播放器)被打开以显示文件 110 中的动态媒体内容,则用户将具有应用程序的用以控制和/或操纵该文件的用户接口选项的全部范围。例如,如果用户想要查看视频文件并打开视频文件查看应用程序(例如,由 Apple Inc. 提供的 Quicktime、由 Microsoft Corporation 提供的 Windows Media Player 等),用户将有能力停止、暂停、播放、调整音量,调整窗口的大小,和/或从打开的应用程序的上下文的内部对文件执行各种其它操作(例如,保存、删除、复制等)。相对地,流式预览 126 给用户提供有限的控制,这是由于其被特别设计为预览。例如,流式预览 126 可以仅仅提供对于停止和播放预览中的内容的控制。

[0020] 流式预览 126 由 API 112——更具体而言,由指定的显示软件包 114 之一——接收和处理。在流式预览的情况下,一个或多个显示软件包 114 包括显示流式预览所需的资源。因此,来自该组显示软件包 114 的流式预览显示软件包获得流式预览 126 并使其准备好显示在显示器 130 上。

[0021] 除了产生流式预览 126 之外,流插件 124 也可以产生和提供一组一种或多种流控制 128 给客户端进程 102。流控制 128 包括一个或多个命令,其允许用户控制流式预览 126 的各个方面。因此,如前所述,流控制 128 可能包括用于停止、暂停、和/或播放流式预览中的内容的命令。在某些实施例中可以包括更多命令或更少命令。在一些实施例中,流控制

128 可以不给用户提供在与文件相关联的原有应用程序中可得到的功能的全部范围。

[0022] 图 2 是图示与产生动态媒体内容的流式预览相关联的各种实施例的框图。与图 1 类似,图 2 示出了客户端进程 202 和流式预览后台程序 206。流插件 220 被用来将文件 208 转换为流式预览。流插件 220 运行在沙箱化流式预览后台程序 206 内。流插件 220 逐帧地产生流式预览,而在某些实施例中,单独地产生视频分量和音频分量。因此,文件 208 被转换成音频分量 212 和视频分量 210。音频分量 212 被添加到流预览(或简称为“流”)214,并且视频分量 210 在每一帧被产生的同时被逐帧地添加到该流中。这样,例如,只要帧 1 被转换和产生为预览特定格式,其就被添加到该流中,并被发送给客户端进程 202,而无需等待帧 2 被转换和产生。类似地,一旦帧 2 被转换和产生,其就可以被立即添加到流 214 中。这个处理对于被转换和产生的每一帧继续进行。

[0023] 如前所述,在各种实施例中流插件 220 给客户端进程提供流控制。如图 2 所示,流控制 216 运行在客户端进程 202 中并且允许用户控制流 214 的某些方面。如本文所述,流控制 216 提供对于流 214 的有限控制,并且不提供在文件 208 在原有应用程序中被打开的情况下可以得到的控制的全部范围。

[0024] 图 3 是图示提供动态媒体内容的预览的各种实施例的流程图。对于动态媒体的预览的请求被接收 310。作为响应,至少一个沙箱化进程被启动 320。单个沙箱化进程可能已经在运行了,在此情况下,第二沙箱化进程可以被启动以将流式媒体预览与正在产生的其它预览分离开。动态媒体的内容类型被确定 330。内容类型可以基于统一类型标识符(UTI)或其它形式的文件类型标识而被确定。

[0025] 基于内容类型,流插件被取回 340 以将动态媒体转换成预览特定格式。本文中使用的“流”插件指的是被特别指定来将动态媒体内容(例如,视频、音频、交互式 3D 动画等)转换成流式媒体的任何插件。如果动态媒体的内容类型是原有文件类型,或换言之,识别出的文件类型,那么有可能快速方便地将该文件转换成预览。但是,如果确定文件类型是非原有文件类型,那么流插件需要能够将原始文件从一种格式转换为与动态预览兼容的预览特定格式。

[0026] 所取回的插件被提供 350 给沙箱化进程,并且预览被产生 360。在文件是包含音频数据和视频数据二者的视频文件的情况下,插件可以为预览产生分离的音频分量和视频分量。如果原始文件仅包含音频,那么可能仅产生音频分量。如果原始文件仅包含视频而没有音频,那么可能仅产生视频分量。流式媒体插件也可以支持包含交互式 3D 动画(诸如 COLLADA 文件)的各种文件。一旦预览分量已被产生,则流预览被提供 370 以显示给用户,并且对于动态预览的控制也被提供 380。这种控制允许用户控制与动态预览相关联的有限功能(例如,“停止”、“暂停”、“播放”)。对于 3-D 动画文件,动态预览可以包括类似控制,诸如停止、暂停或播放动画文件的能力。但是,在某些实施例中,与在原有应用程序被打开以播放动态媒体内容时可得到的控制相比,所提供的用户控制是有限的。

[0027] 应当注意:在替换实施例中图 3 中所描述的步骤可以被重新排列并以与所示顺序不同的顺序被执行。而且,应当注意:比图 3 所示的步骤更多的处理步骤或更少的处理步骤可以被用来实现由本文所述的各种实施例设想的相同动态媒体预览功能。

[0028] 图 4 图示了具有计算机系统 400 的示例性形式的机器的图形表示,在所述机器中可以执行使得该机器执行本文所述的方法中的任意一个或多个的一组指令。在替换实施例

中,机器可以连接(例如,联网)到局域网(LAN)、内联网、外联网或互联网中的其它机器。机器可以以客户端-服务器网络环境下的服务器或客户机的能力工作,或工作为对等(或分布式)网络环境下的对等机器。机器可以是个人计算机(PC)、平板PC、机顶盒(STB)、个人数字助理(PDA)、蜂窝电话、或能够执行指定要由该机器采取的动作的一组指令(顺序地或以其它方式)的任何机器。此外,虽然只示出了一个机器,但是术语“机器”应当被看作为包括能够独立地或相结合地执行一组(或多组)指令以执行本文所述的方法中的任意一个或多个的机器的任意集合。

[0029] 根据各种实施例,图4还表示可以用来实现图1-3所示的所述细节的系统的一种形式。尤其是,应当注意:显示器410可以用来显示动态媒体内容,就像图1的显示器130那样。可以将本文所述的有助于动态媒体内容预览的实施例包括为指令422(在图4的系统中),所述指令422被存储例如在驱动单元418或主存储器404中,并且可以由处理器402执行。

[0030] 示例性计算机系统400包括处理器402、主存储器404(例如,只读存储器(ROM)、闪存、动态随机存取存储器(DRAM)、诸如同步DRAM(SDRAM)或高频动态随机DRAM(Rambus DRAM,即RDRAM)等)、静态存储器406(例如,闪存、静态随机存取存储器(SRAM)等)和辅存储器418(例如,数据存储设备),其经由总线408彼此进行通信。

[0031] 处理器402代表一个或多个通用处理设备,诸如微处理器、中央处理单元等。更具体地,处理器402可以是复杂指令集计算(CISC)微处理器、精简指令集计算(RISC)微处理器、超长指令字(VLIW)微处理器、实现其它指令集的处理器、或实现指令集的组合的处理器。处理器402也可以是一个或多个专用处理设备,诸如专用集成电路(ASIC)、现场可编程门阵列(FPGA)、数字信号处理器(DSP)、网络处理器等。处理器402被配置成运行用于执行本文中所述的操作和步骤的处理逻辑422。

[0032] 计算机系统400还可以包括网络接口设备416。计算机系统400还可以包括显示单元410(例如,液晶显示器(LCD)、发光二极管(LED)显示器、阴极射线管(CRT))和输入设备412(例如,键盘和/或鼠标等)。

[0033] 辅存储器418可以包括机器可读存储介质(或更具体地,计算机可读存储介质)424,其上存储有体现本文所述的方法或功能中的任意一个或多个的一组或多组指令集(例如,软件422)。在由计算机系统400执行期间,软件422还可以完全或至少部分地驻留在主存储器404内和/或在处理设备402内,主存储器404和处理设备402也构成机器可读存储介质。软件422还可以通过网络420经由网络接口设备416被发送或接收。在各种实施例中,用户请求的网络内容可以使用网络接口设备416通过网络420被取回(例如,从广域网)。

[0034] 虽然在示例性实施例中机器可读存储介质424被示出为单个介质,但是术语“机器可读存储介质”或“计算机可读存储介质”应被看作为包括存储一个或多个指令集的单个介质或多个介质(例如,集中式或分布式数据库、和/或相关的高速缓存和服务器)。术语“机器可读存储介质”或“计算机可读存储介质”应被看作为包括任意介质,其能够存储或编码一组供机器/计算机执行的指令,并使得机器/计算机执行本发明的方法中的任意一个或多个。术语“机器可读存储介质”或“计算机可读存储介质”应被相应地看作为包括但不限于固态存储器和光学和磁性介质。

[0035] 本文所述的各种部件可以是用于执行本文所述的功能的装置。本文所述的每一部件包括软件、硬件或其组合。本文所述的操作和功能可以被实现为软件模块、硬件模块、专用硬件（例如，应用特定硬件、专用集成电路（ASIC）、数字信号处理器（DSP）等）、嵌入式控制器、硬连线电路等。

[0036] 除了本文所述的内容以外，可以对所公开的本发明的实施例和实施方式进行各种修改而不会背离本发明的范围。因此，本文中的图解说明和示例应当被看作是说明性的，而不是限制性的。

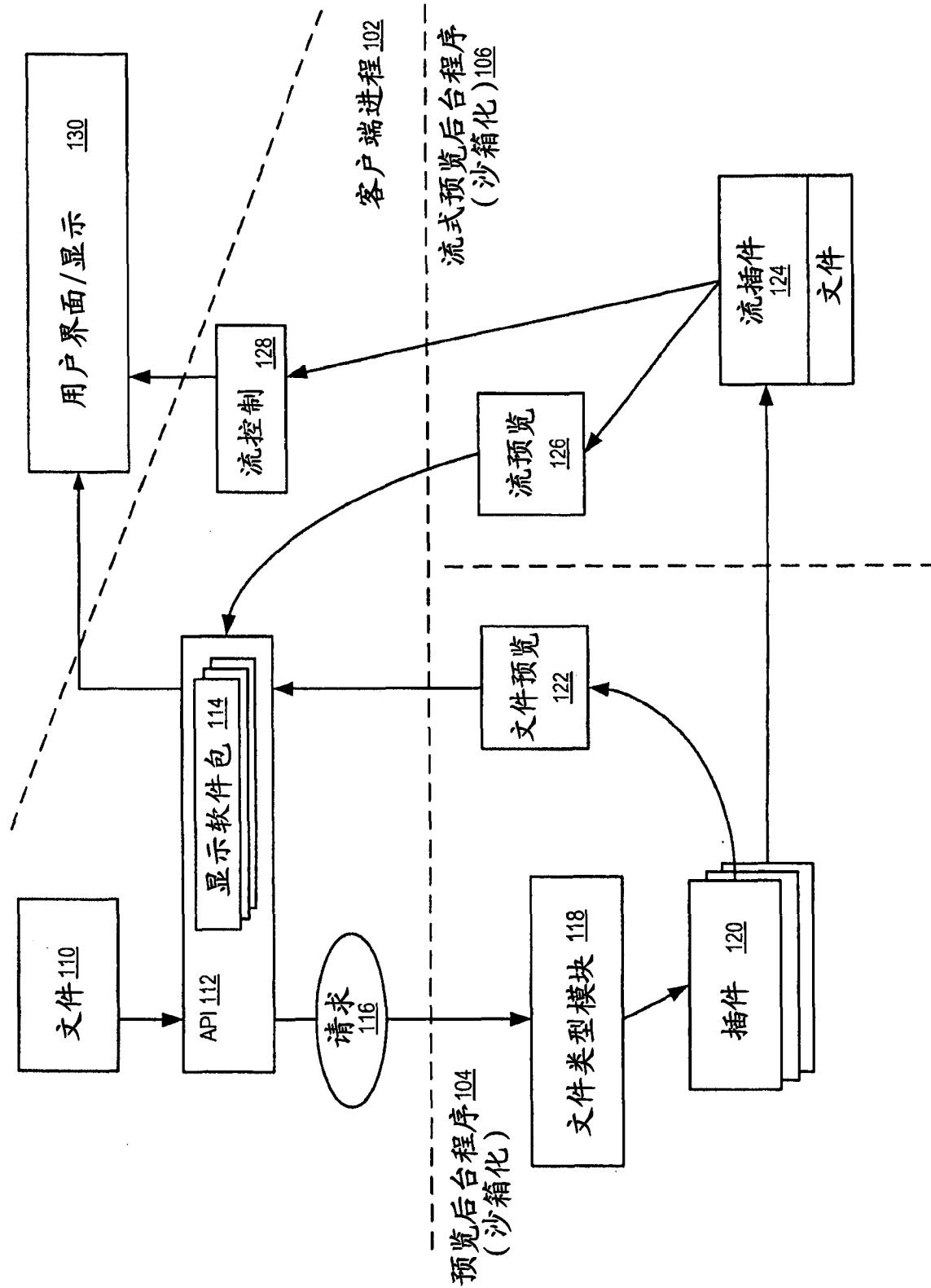


图 1

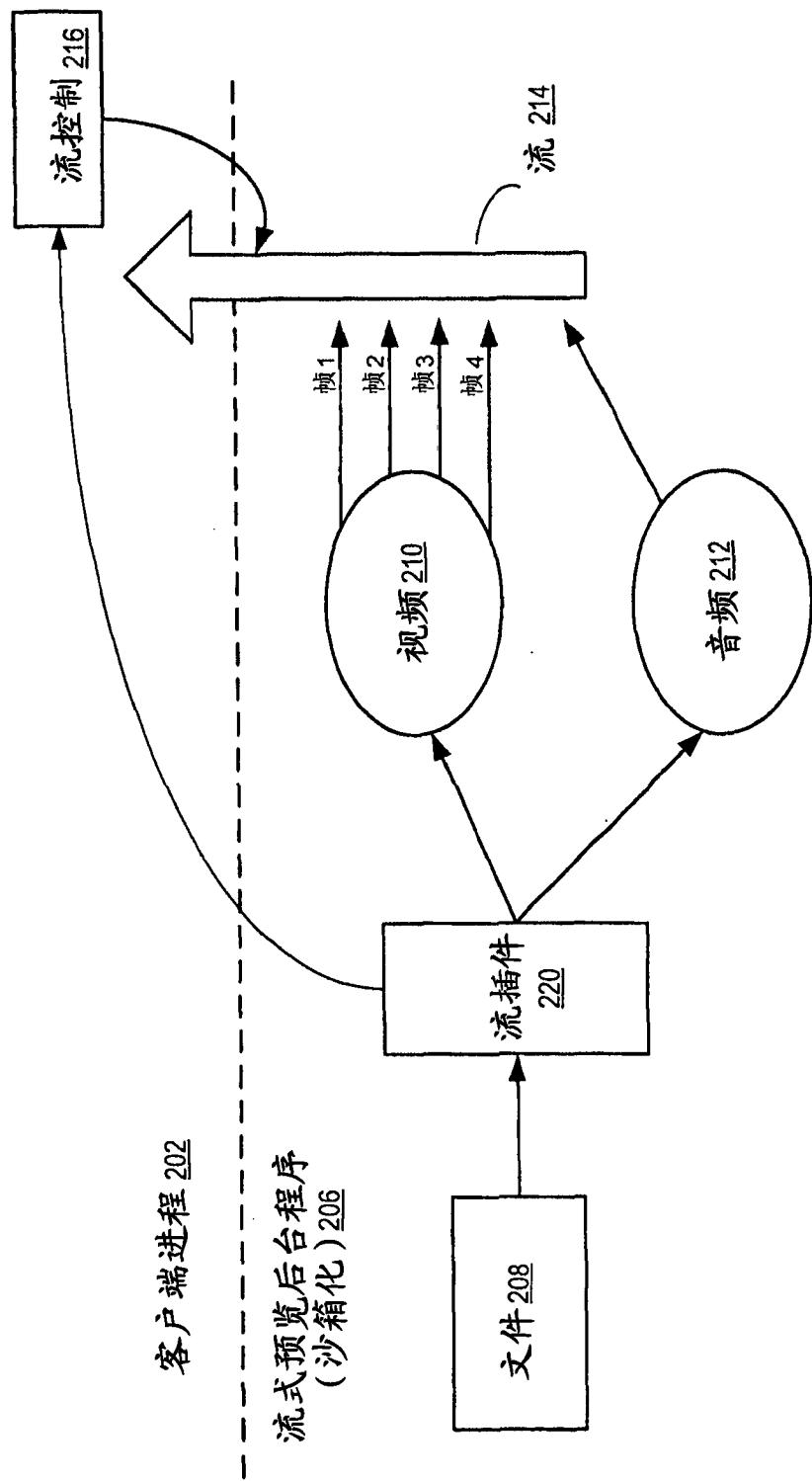


图 2

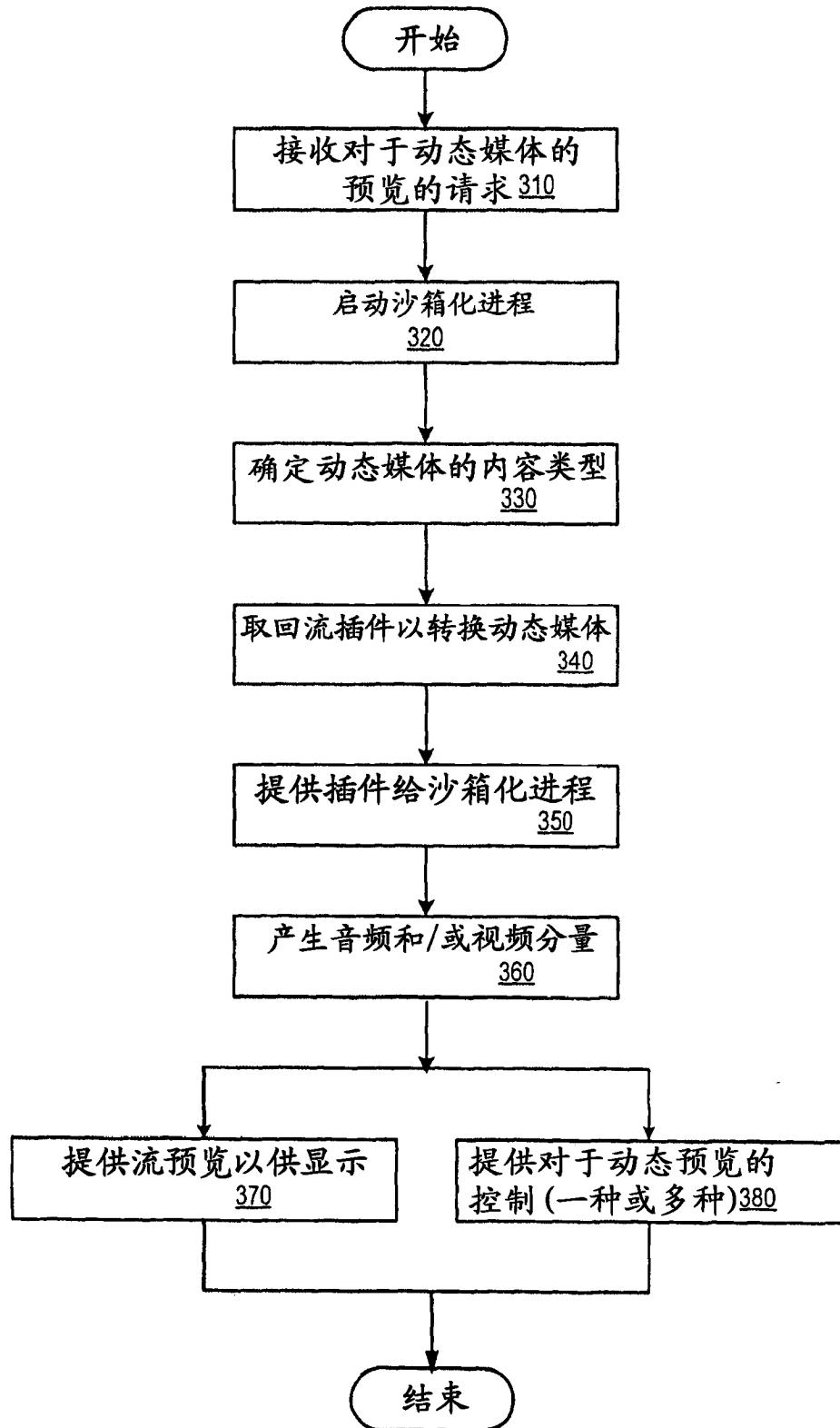


图 3

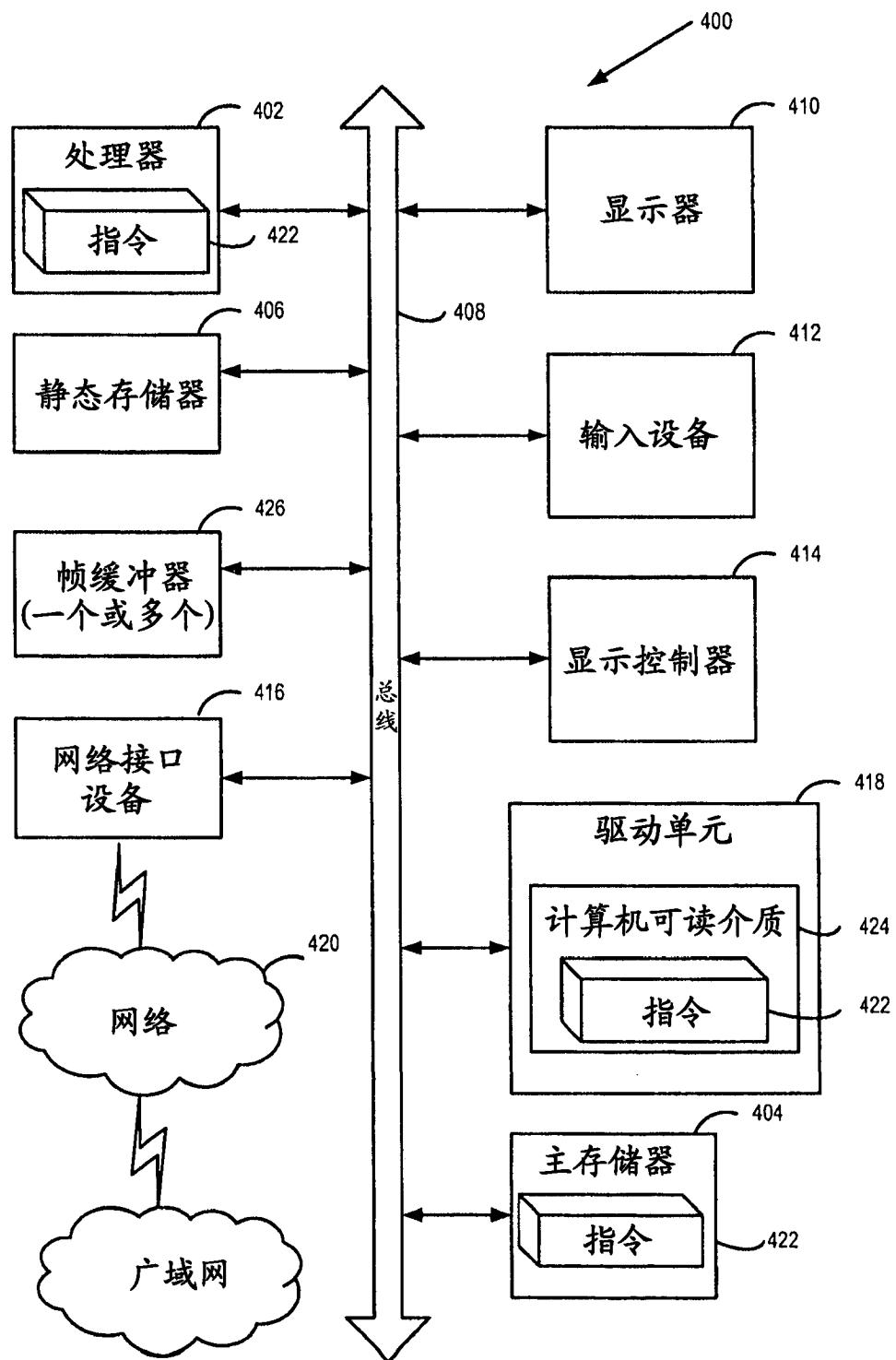


图 4