

[19] 中华人民共和国国家知识产权局



[12] 发明专利申请公布说明书

[21] 申请号 200810002232.5

[51] Int. Cl.

G11B 27/10 (2006.01)

G11B 27/32 (2006.01)

G11B 27/34 (2006.01)

G06F 9/445 (2006.01)

G06F 17/30 (2006.01)

[43] 公开日 2008 年 6 月 11 日

[11] 公开号 CN 101197175A

[22] 申请日 2004.9.30

[21] 申请号 200810002232.5

分案原申请号 200480029571.2

[30] 优先权

[32] 2003.10.10 [33] EP [31] 03292522.4

[71] 申请人 皇家飞利浦电子股份有限公司

地址 荷兰艾恩德霍芬

[72] 发明人 P·S·纽顿 D·P·凯利

[74] 专利代理机构 中国专利代理(香港)有限公司

代理人 龚海军 刘红

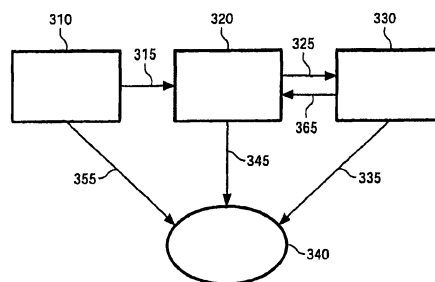
权利要求书 3 页 说明书 8 页 附图 2 页

[54] 发明名称

用于蓝光光盘的 JAVA 生存周期模型

[57] 摘要

光学存储器载体利用实现为 Java 对象的应用程序来加载, 该 Java 对象启动交互式服务, 例如游戏、交互式 A/V 内容重放、web 浏览。每个 Java 对象包括至少三种方法, 用于设置应用程序为不活动状态: 暂停状态或活动状态。也利用包括主方法的加载器应用程序来加载该载体。加载器应用程序控制载体上存在的所有 Java 对象的对应生存周期并且控制 Java 对象的相应方法。加载器应用程序每次仅仅允许运行一个 Java 对象。加载器应用程序的生存周期与该盘相链接, 当盘插入设备时, 将加载器应用程序加载到存储器中; 并且当移走盘时, 取消加载器应用程序。该设备运行 Java 虚拟机, 加载器应用程序被配置在 Java 虚拟机之上运行。因为加载器应用程序的生存周期与该盘相链接, 所以 Java 虚拟机在盘插入设备的时间内运行, 并且当 Java 对象暂停或取消时不取消 Java 虚拟机。



1. 一种重放来自存储器载体的音频-视频内容的方法, 该存储器载体包括在其上存储的:

音频-视频内容 (260);

用于重放音频-视频内容 (260) 的相关联的序列的电影对象 (250);

Java 对象应用程序 (220-240), 当在运行于重放系统中的 Java 虚拟机上执行 Java 对象应用程序时, 能显示和处理音频-视频内容 (260);

索引表, 包括允许用户选择和播放相关联的电影对象 (250) 的标题, 索引表的一部分能显示给用户以用于选择; 对于一个链接的标题所述索引表进一步表示链接到所述链接的标题的 Java 对象应用程序;

所述方法包括如下步骤:

-使用索引表用于显示菜单, 菜单包括用于允许用户选择和播放电影对象的标题列表;

-接收来自用户的选择以播放与链接的标题相关联的电影对象;

-使用索引表来确定链接到所述链接的标题的 Java 对象应用程序 (220-240);

-启动链接的 Java 对象应用程序 (220-240), 链接的 Java 对象应用程序 (220-240) 能显示与选择的电影对象相关联的音频-视频内容 (260) 的序列。

2. 根据权利要求 1 的方法, 其中所述链接的 Java 对象应用程序 (220-240) 包括至少一种暂停方法, 当所述暂停方法被调用时导致所述链接的 Java 对象应用程序 (220-240) 处于暂停状态;

该方法进一步包括如下步骤:

-调用所述链接的 Java 对象应用程序 (250) 的暂停方法;

-继续重放在调用所述暂停方法之前由所述链接的 Java 对象应用程序 (220-240) 启动的相关联的音频-视频内容 (260) 的序列, 而不考虑在调用所述暂停方法之后出现的与相关联的 A/V 内容 (260) 的序列有关的事件。

3. 一种存储器载体, 包括在其上记录的:

音频-视频内容 (260);

用于重放音频-视频内容(260)的相关联的序列的电影对象(250);

Java 对象应用程序(220-240), 当在运行于重放系统中的 Java 虚拟机上执行 Java 对象应用程序时, 能显示和处理音频-视频内容(260);

索引表, 包括允许用户选择和播放相关联的电影对象(250)标题, 索引表的一部分能显示给用户以用于选择;

其中对于一个链接的标题所述索引表进一步表示链接到该链接的标题的 Java 对象应用程序。

4. 根据权利要求 3 的存储器载体, 其特征在於所述链接的 Java 对象应用程序(220-240)包括至少一种暂停方法, 当所述暂停方法被调用时导致所述链接的 Java 对象应用程序(220-240)处于暂停状态;

所述链接的 Java 对象应用程序(220-240)适合于继续重放在进入暂停状态之前由所述链接的 Java 对象应用程序(220-240)启动的相关联的音频-视频内容(260),

所述链接的 Java 对象应用程序(220-240)进一步适合于不考虑在 Java 对象应用程序(220-240)处于暂停状态时出现的与相关联的 A/V 内容(260)有关的事件。

5. 用于重放来自根据权利要求 3 或 4 的存储器载体的音频-视频内容的系统, 所述系统适合于执行权利要求 1 或 2 的方法。

6. 一种重放来自存储器载体的音频-视频内容的方法, 该存储器载体包括在其上存储的:

音频-视频内容(260);

Java 对象应用程序(220-240), 所述 Java 对象应用程序(220-240)包括至少一种暂停方法, 当所述暂停方法被调用时导致所述 Java 对象应用程序(220-240)处于暂停状态; 当在运行于重放系统中的 Java 虚拟机上执行所述 Java 对象应用程序(220-240)时, 能使 Java 对象应用程序(220-240)启动相关联的音频-视频内容(260)的序列的重放,

该方法包括:

-执行所述 Java 对象应用程序(220-240), 执行所述 Java 对象应用程序(220-240)包括重放相关联的音频-视频内容(260);

-调用所述 Java 对象应用程序的暂停方法;

-继续重放在调用所述暂停方法之前由所述 Java 对象应用程序 (220-240) 启动的相关联的音频-视频内容 (260), 而不考虑继调用所述暂停方法之后与相关联的 A/V 内容 (260) 有关的事件。

7. 一种存储器载体, 该存储器载体包括在其上记录的:

-音频-视频内容 (260);

-Java 对象应用程序 (220-240), 当在运行于重放系统中的 Java 虚拟机上执行 Java 对象应用程序时, 能显示和处理音频-视频内容 (260); 所述 Java 对象应用程序 (220-240) 包括至少一种暂停方法, 当所述暂停方法被调用时导致所述 Java 对象应用程序 (220-240) 处于暂停状态; 当在运行于重放系统中的 Java 虚拟机上执行所述 Java 对象应用程序 (220-240) 时, 能使所述 Java 对象应用程序 (220-240) 启动相关联的音频-视频内容 (260) 的序列的重放;

所述 Java 对象应用程序 (220-240) 在暂停状态适合于继续重放在调用暂停方法之前启动的相关联的音频-视频内容 (260), 而不考虑与相关联的 A/V 内容 (260) 有关的事件。

8. 用于重放来自根据权利要求 7 的存储器载体的音频-视频内容的系统, 所述系统适合于执行权利要求 6 的方法。

用于蓝光光盘的 JAVA 生存周期模型

本案是申请号为 200480029571.2、申请日为 2004 年 9 月 30 日、发明名称为“用于蓝光光盘的 JAVA 生存周期模型”的分案申请。

技术领域

本发明涉及光学存储器的领域。本发明尤其涉及蓝光光盘、播放器以及它们的逻辑规范。

背景技术

BD-ROM(蓝光光盘 ROM)是消费电子设备、软件公司和娱乐产业的协会从事的即将到来的光学存储器规范,其中它们期望集成最大的存储器、重放以及软件技术以开发下一代载体和播放器,其将新用户体验机会和高存储密度结合。BD-ROM 是下一代只读光盘标准,它利用蓝激光以便每一数据层可能获得至少 25GB 的容量。光学标准可以定义载体的物理和逻辑格式。本发明涉及光学载体的逻辑格式,即光盘上的应用层,在此将不讨论物理格式。用于 BD-ROM 的主要预测应用是高清晰度电视,然而,诸如 web 浏览、聊天应用或游戏之类的 web 连接和交互性也在该标准的范围内。

BD-ROM 标准的当前草案规定了电影模式,其中通过与电影对象的交互体验数字音频/视频(A/V)数据。电影模式的概念包含与数字通用光盘类似的功能,该电影模式主要涉及 A/V 内容的播放。电影对象可以启动包括存储在盘上的一系列 A/V 序列的播放列表并且也允许内容创作者对有关 A/V 重放进行一些前期和后期处理。电影模式还包括标题,该标题作为用户查看的项目并且用户可以选择该标题来重放。索引表将标题与电影对象链接。

除了这个 A/V 重放能力之外,交互式服务将可用于新一代光学介质并且需要一种编程环境来启动存在于盘上的交互性应用程序。产业协会把具有现有的虚拟机、应用编程接口以及库的 Java™,SUN 技术看作好的候选技术以允许光学存储器载体上交交互性应用程序的可行性。然而,在个人计算机环境适应消费电子设备中出现若干问题,用户期望和体验是不同。

第一个问题是软件应用程序可能包含在研发或测试期间没有暴露出来的错误,并且这些没有检测到的代码错误应该不会阻止读取 BD-

ROM 或者阻止运行 BD-ROM 播放器。的确，应用程序出乎意料地停止的事实也应该不会导致播放器停止。第二个问题（这个问题是 Java 所特有的）通常是在 Java 中，一个应用程序在 Java 虚拟机的一个实例上运行并且停止该应用程序通常意味着当必须释放到本地平台的所有钩（hook）时停止 Java 虚拟机。停止和开始虚拟机意味着下载与环境有关的所有类文件并且这个过程占用 30 秒。然而，约为若干秒的延迟在消费电子应用中是不可接受的。

发明内容

因此，发明人寻找了一种缓解这些缺点的解决方案并且已知的多媒体家庭平台（MHP）节目环境提供它们一个基础以开发其中应用程序显示出与 MHP 定义的生存周期模型相似的生存周期模型的一个环境。

MHP 是用于交互式电视的基于 Java 标准。在 MHP 中，应用程序可以作为一个独立的应用程序而运行或者它可以执行所谓的 Xlet 接口。Xlet 通常是执行简单功能（例如，电子节目指南（EPG）、交互式游戏、增强的内容、管理广播介质流水线、聊天应用、菜单列表等等）的小程序。Xlet 是用于 Java 应用程序的另一名字并且像 applet，Xlet 受运行它们的软件的控制。在 applet 的情况下，下层软件是浏览器或 applet 阅读器。在 Xlet 的情况下，下层软件是数字电视接收机中的机顶盒或者支持 Java 平台的任何其他消费电子设备。Xlet 不包括主方法并且执行定义了 Xlet 的生存周期方法特征（signatures）的 Xlet 接口。Xlet 接口定义了生存周期方法以用信号通知下面状态变化：创建、启动、开始、暂停和破坏。所有 Java 电视实现都具有应用程序管理器，该应用程序管理器调用 Xlet 的生存周期方法通过它们不同的应用程序状态移动一个或多个 Xlet。

例如，观众可能正播放游戏连同游戏竞赛节目并且决定查看节目列表。如果节目列表和游戏都是 Xlet，则用信号通知接收机中的应用程序管理器：当观众选择节目列表时，Xlet 存在。此时，应用程序管理器暂停节目 Xlet，并且接收机将节目列表 Xlet 下载到接收机中。应用管理器下载并且启动节目列表 Xlet。

基于 Xlet 概念，发明人实现了用于 BD-ROM 环境（播放器和载体）的相似类型的应用程序。因此，本发明涉及包括具有预定生存周期模型的至少一个 Java 对象应用程序的光学存储器载体。生存周期模型定

义了至少三种方法，当调用这些方法时导致应用处于下面相应状态之一：暂停状态、活动状态、不活动状态。载体还包括加载器应用程序，该加载器应用程序被配置以在 Java 虚拟机上运行。这个加载器应用程序具有主方法以及生存周期，该生存周期与载体到扫描设备的插入相链接（link），并且加载器应用程序进一步被配置以控制对象应用程序的生存周期。加载器也是这样的以致于它把对象应用程序控制到暂停状态或不活动状态，无论何时调用存储在载体上的另一对象应用程序。

本发明将最初在 MHP 中使用的一些概念应用于 BD-ROM 盘，该 BD-ROM 盘现在加载至少一个 Java 对象应用程序和加载器应用程序。Java 对象应用程序和加载器应用程序在一些方面类似于对应的 Java 对象应用程序和应用程序管理器。在本发明中，仅仅加载器应用程序包括主方法并且加载器应用程序控制对象应用程序的生存周期。加载器应用程序的生存周期与盘到扫描设备的插入相链接，这意味着当插入和扫描盘时启动载入加载器应用程序并且随后当从设备弹出盘时中断载入加载器应用程序。因此，加载器应用程序在盘的持续时间内在 Java 虚拟机上运行并且当盘在设备中时 Java 虚拟机结果没有被取消或重新启动。因此，缓解了与上述的 Java 虚拟机的引导相关联的其中一个缺点。另外，本发明的一个特性是加载器应用程序每次仅仅允许运行一个对象应用程序并且如果用户或系统调用另一对象应用程序同时第一对象应用程序正运行，则导致第一对象应用程序进入暂停或不活动状态。本发明在各种方面不同于 MHP。首先，在 MHP 中，多于一个的 Xlet 同时运行，而本发明每次仅仅允许运行一个 Java 对象应用程序，并且加载器应用程序被存储在具有 Java 对象应用程序的盘上并且不在其中存储 Java 虚拟机的播放器中。本发明的一个优点是开发者共同体提供可靠且接收良好的计算环境以开发用于光介质的应用程序。

附图说明

现在参考附图通过示例将更加详细地描述了本发明，其中：

图 1 示出了 Java 环境的基本部分；

图 2 示出了本发明的盘/播放器系统的逻辑部分；以及

图 3 示出了本发明的 Java 对象应用程序的方法和状态。

具体实施方式

附图中具有相似或相应特性的元件通过相似参考数字来标识。

图1说明了典型Java环境的基本部分。典型Java环境包括在设备的硬件110上运行的软件部分。该设备可以是个人计算机或消费电子设备,例如网络读写板、个人数字助理、游戏控制台、机顶盒等。Java技术被设计使相同应用程序在不同种类网络和各种硬件结构上执行。Java虚拟机(VM)120是Java平台的第一部分,其启用中性的体系结构技术。Java VM120是可移动Java技术平台的两个主要元件之一,其中Java编程语言编译器产生代码。Java VM120被修改或关闭(ported)以适合特定硬件110。仅仅为软件的Java平台除了Java VM120之外还包括Java应用程序编程接口(API)130。Java API130是现成软件部分的大集合,它提供许多有用的能力,例如图形用户接口(GUI)、连通性、I/O等等。Java API130被分组到相关类和接口的库中;这些库在该产业中也称为包(package)。然后,在Java平台120,130之上,执行以Java代码编写的开发者应用程序组140。

图2示出了本发明的BD-ROM系统的框图。该系统包括盘200和播放器100。播放器100包括硬件110、具有Java库的应用程序编程接口125和Java VM120。BD-Rom盘或蓝光盘200包括加载器应用程序210、Java对象220、230、240,A/V数据260和电影对象250。

必须注意的是,尽管在BD-Rom的环境中将描述本发明,但是本发明包括满足本发明特性的任何现有或将来类型的光学或非光学存储器载体,不管该载体是只读、可写还是可重写的。

如在介绍部分所解释的,BD-ROM盘200可以启用交互式模式和电影模式。在图2中,通过块260和250说明电影模式。电影对象250与存储在盘200的数据存储部分260中的A/V序列的一个或多个播放列表相关联并且电影对象250允许观看和处理A/V内容260。例如,电影对象250指向例如形成电影的A/V剪辑的播放列表。用户可以通过索引表中的标题来选择和播放A/V剪辑或电影。索引表可以位于根菜单中并且索引表的一部分可显示给用户,以用于选择。

交互式模式包括Xlet或Java对象220、230、240和加载器应用程序210。Java对象220、230、240除了提供诸如聊天、菜单列表、屏幕动画、游戏、因特网浏览等之类的交互式应用程序之外,还能够

控制 A/V 重放。Java 对象 220-240 和加载器应用程序 210 可以一起存储在盘 200 上作为一个单独的 Jar 文件。Jar 文件是 Java 存档文件，它可以由 Java VM120 自动地压缩、加载和解压缩。这样一个 Jar 文件可以包括明显文件 (manifest file)，它描述包括在存档中的文件并且表示包含加载器应用程序 210 的文件。加载器应用程序 210 包括主方法并且可以是的存储存储在盘 200 上的唯一应用程序。加载器应用程序 210 的生存周期固有地与盘 200 链接，当盘 200 插入播放器 100 时加载器应用程序的生存周期被启动加载；当盘 200 从播放器 100 移出时该加载器应用程序的生存周期停止。因此，加载器应用程序在盘 200 的持续时间内运行。相反地，Java 对象 220-240 不包括主方法并且仅仅在盘 200 的持续时间的一部分内运行。

Java 对象 220-240 能够启动 A/V 序列的播放列表、捕捉和处理用户操作和命令，例如用户按下遥控键并且最后将控制传送给电影对象 250。在这个实施例，当将控制从 Java 对象 220-240 传送到电影对象 250 时，加载器应用程序 210 将取消或暂停 Java 对象 220-240，正如在下文根据图 3 所说明的生存周期模型将要解释的。然而，尽管加载器应用程序 210 能导致 Java 对象 220-240 处于不活动状态 340，但是加载器应用程序 210 仍在 Java VM120 上运行，该 Java VM120 不被终止。

加载器应用程序 210 保存在盘 200 上存在的一系列所有应用程序或 Java 对象 220-240 并且控制所有对象 220-240 的相应生存周期。加载器应用程序 210 通过调用关于应用程序或对象 220-240 状态的其方法 215、225、235、245 或 255 来发出命令给对象 220、230 或 240。应用程序 Java 对象 220-240 可以处于以下四个状态之一中：启动状态 310、暂停状态 320、活动状态 330 和不活动或取消状态 240，并且对象 220-240 的生存周期包括可以调用的四种方法 315-355。加载器应用程序 210 首先加载存储器中的 Java 对象 220-240 为加载状态 310，它基本上是自动地或通过初始化进行，并且通过加载器应用程序 210 调用初始化方法 315，迫使进入暂停状态 320，在该暂停状态 320 应用程序或对象 220-240 正等候命令。暂停状态 320 中的 Java 对象 220-240 使用来自设备 100 和 Java VM120 的最小资源，并且在设备 100 的屏幕或任何输出上不出现任何内容。从暂停状态 320 中，加载器

应用程序 210 调用启动方法 325 并且将 Java 对象 220-240 激活到活动状态 330, 在该活动状态 330 中, 与 Java 对象 220-240 相关联的应用程序正运行。从活动状态 330, 加载器应用程序 210 通过调用暂停方法 265 将 Java 对象 220-240 暂停回到暂停状态 320 或者通过调用破坏方法 335 将取消活动 Java 对象 220-240 进入到状态 340。加载器应用程序 210 破坏来自状态 310、320 或 330 的任何状态的 Java 对象 220-240, 或者加载器应用程序 210 等待直到 Java 对象 220-240 自动地终止为止。

在一个实施例中, 按照如下实现暂停状态 320。在暂停状态 320 中, Java 对象 220-240 必须停止提供任何服务并可能释放资源, 并且变为静止。释放所有资源包括释放所有用户事件和控制, 例如遥控键、键盘、声音输入、触摸屏输入等等。Java 对象 220-240 不保持对这些控制和资源的专门访问, 并且不必响应由它们产生的事件。可替换地, Java 对象 220-240 保持对一个或多个 A/V 控制资源的暂时访问。例如, 在 Java 对象 220-24 迫使进入暂停状态 320 之前, 作为播放列表一部分而启动的任何 A/V 内容可以继续运行。然而, Java 对象 220-240 不启动任何新的播放列表或启动新内容的显示, 例如播放列表上的下一项, 或响应与 A/V 内容相关的事件。

如上所述的, Java 对象通过调用其启动方法 325 而从其暂停状态 320 返回。当 Java 对象 220-240 返回活动状态 330 时, Java 对象 220-240 收回在暂停状态 320 中释放的所有资源。这意味着注册用户事件, 屏幕显示在启动方法 325 中实现或者从启动方法中开始。在启动方法之后, 注册对 A/V 控制的访问被执行并且被保持直到取消 Java 对象 220-240 或 Xlet。在计算环境中, 注册用户事件是必需的以便接收用户命令和输入。如果不包括注册过程, 则 Java 对象将不转送用户输入并且因此将对它们不响应。

如上所述的, 可以将控制从 Java 对象 220-240 传送到电影对象 250。当控制从 Java 对象 220-240 传送到电影对象 250 时, 加载器应用程序 210 取消 Java 对象 220-240, 正如下文根据图 3 所解释的。然而, 尽管加载器应用程序 210 可以导致 Java 对象 220-240 处于不活动状态 340, 但是加载器应用程序 210 仍在 Java VM120 上运行, 该 Java VM120 不被终止。加载器应用程序 210 在背景中运行直到控制被

传送回加载器应用程序 210, 在这个情况中加载器应用程序 210 运行与根菜单相关联的 Java 对象 220-240, 该根菜单示出了交互式标题的列表。可替换地, 电影对象 250 示出了交互式标题的菜单而不是 Java 对象 220-240, 并且当选择一个交互式标题时, 控制被传送到加载器应用程序 210 并且启动与所选择标题链接的 Java 对象 220-240。

当用户按压 Java 标题期间的菜单时, 也需要暂停状态 320。当这发生时, 启动菜单并且启动另一标题。如果用户按下恢复命令, 则播放器 100 恢复当第一菜单调用发生时最初播放的标题。如果用户再次选择菜单按钮, 则暂停当前播放的标题并且取消先前的标题。播放器 100 具有一个暂停的 Java 对象 220-240, 当用户要求恢复时, 该对象 220-240 继续。如果当用户试图暂停第二 Java 对象时暂停 Java 对象 220-240, 则第一暂停的 Java 对象被取消以便可以暂停第二 Java 对象。

必须理解的是, 本发明包括每次运行一个应用程序 Java 对象 220-240。因此, 在一个典型实施例中, 加载器应用程序 210 在启动时启动与交互式菜单的显示相关联的 Java 对象 220-240。用户可以从多个交互式标题中进行选择, 其中每个显示的标题与另一 Java 对象 220-240 链接。当用户选择一个标题时, 加载器应用程序 210 取消或暂停菜单 Java 对象 220-240, 并且启动与该标题链接的 Java 对象 220-240。例如, 新 Java 对象 220-240 与 A/V 内容、游戏的播放或者与包括多个可选择项目或标题的另一菜单列表的显示相关联, 该另一菜单列表与另一组 Java 对象的相关。

在另一典型实施例中, 需要一种机制, 该机制用信号通知: 哪个 Java 对象 220-240 被允许利用哪些标题来运行。需要这种机制以允许容易的实现并且能够精确的启动且关闭 Java 对象。索引表被定义, 以表示当选择一个标题时哪个 (或哪些) 电影对象应该运行。索引表存储索引, 这些索引存储了到电影对象 250 或 Java 对象 220-240 的链接。

在另一实施例中, 电影对象 250 可以被实现为执行等同功能的 Java 对象, 并且在这种情况下, 盘 100 规定了完整的 Java 环境。

对本领域的技术人员应该显而易见的是, 许多其他可替换的实施例也属于所附加权利要求的范围。在这一方面, 下面进行了结束语评

论。存在通过硬件项或软件项或二者实现功能的许多方法。例如，在此所示出的附图是图解性的，每个图仅仅表示一个实施例。因此，尽管附图示出了作为不同块的不同功能，但是这一点也不排除单个硬件项或单个软件项执行若干功能。也不排除通过多个硬件项、多个软件项或二者的集合来执行一个功能。

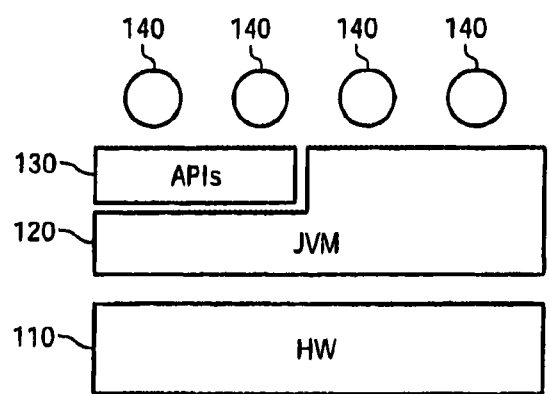


图 1

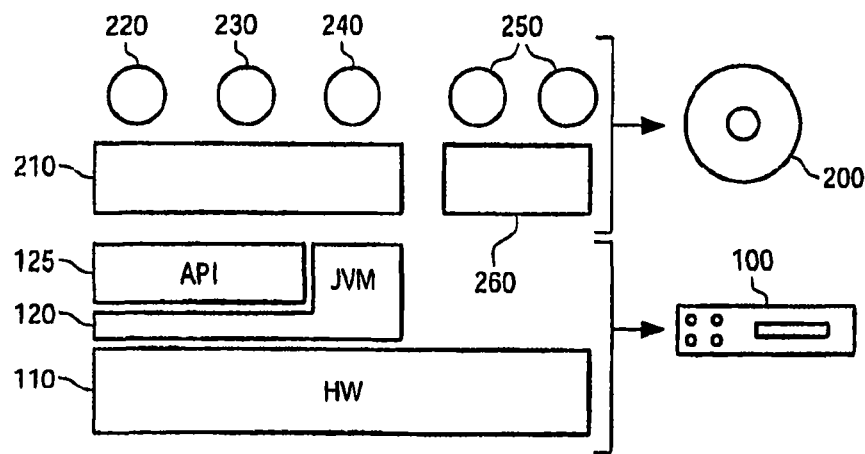


图 2

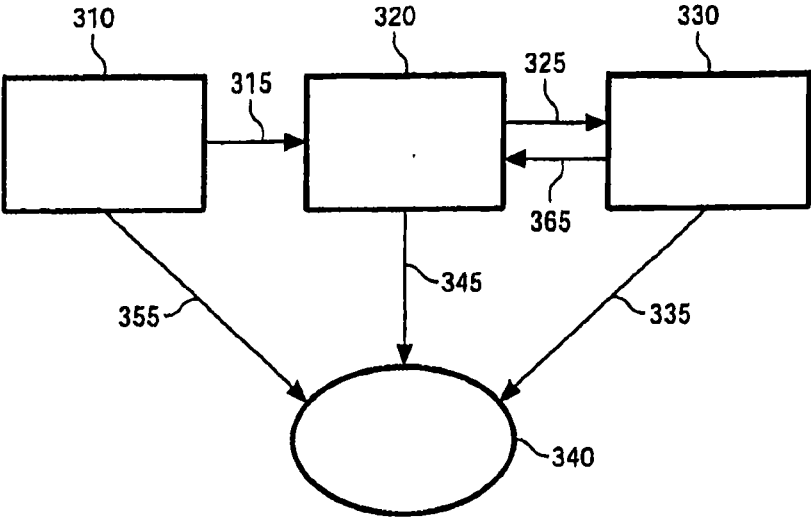


图 3