(19)中华人民共和国国家知识产权局

(12)发明专利申请

(10)申请公布号 CN 106775845 A
(43)申请公布日 2017.05.31

(21)申请号 201611084766.8

(22)申请日 2016.11.30

(71)申请人 用友优普信息技术有限公司
地址 100094 北京市海淀区北清路68号用友软件园

(72)发明人 杨历

(74)专利代理机构 北京中恒高博知识产权代理有限公司 11249
代理人 宋敏

(51)Int.Cl.
G06F 9/445(2006.01)

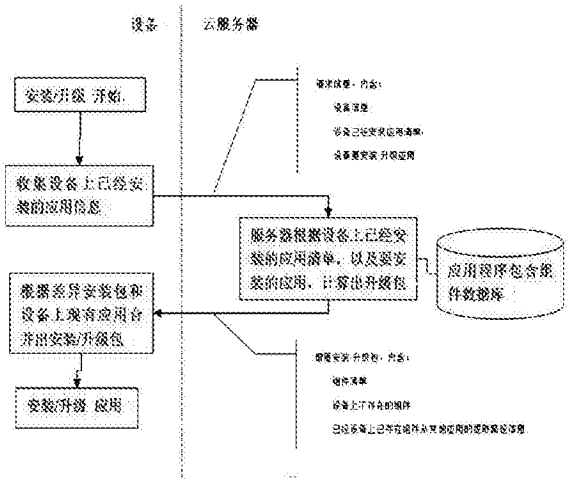
权利要求书1页 说明书4页 附图3页

(54)发明名称

一种软件安装升级的方法

(57)摘要

本发明公开了一种软件安装升级的方法，由于主要包括：收集设备上已经安装的应用信息，服务器根据设备上已经安装的应用清单，以及要安装的应用计算出升级包，根据差异安装包和设备上现有应用合并出安装/升级包，最终安装升级应用；从而可以克服现有技术中流量节省作用有限，无法节省软件首次安装时的流量占用，从而大大减少软件首次安装以及软件升级时的网络流量。



CN 106775845 A

1.一种软件安装升级的方法,其特征在于,包括以下步骤:

步骤1:设备需要安装或者升级应用时,安装或升级程序不直接向服务器请求应用安装包,首先收集设备上安装的应用清单,并计算每个应用的SHA1值;

步骤2:收集完成后,安装或升级程序向服务器发送安装或升级请求;

步骤3:服务器收到安装或升级请求后,根据设备上已经安装的应用清单,以及要安装或升级的应用,计算出增量安装包或升级包;

步骤4:根据差异安装包和设备上现有应用合并出安装包或升级包,得到全量安装包;

步骤5:安装最终得到的全量安装包。

2.根据权利要求1所述的软件安装升级的方法,其特征在于,步骤2中,请求内容包括操作系统版本、设备上已经安装的所有应用的清单,每个应用的SHA1值和需要安装或升级的应用信息。

3.根据权利要求2所述的软件安装升级的方法,其特征在于,步骤3中具体为,服务器收到安装/升级请求后,利用服务器内的应用包含的组件版本数据库信息,计算增量安装包,具体包括,

当某一安装组件在设备上的所有已安装应用内均不存在或某一安装组件在设备上已安装的应用内存在,但它的SHA1值与所有已安装应用内同名组件的SHA1值均不同,则不修改MAINFEST.MF文件内对应的组件描述信息,也不从最终安装包中剔除组件信息;

当某一安装组件在设备上已安装的应用内存在,且SHA1值也相同的组件,从安装包中剔除该组件,并修改MAINFEST.MF文件内容,将对应组件的路径信息前添加设备上存在该组件的应用名称。

4.根据权利要求3所述的软件安装升级的方法,其特征在于,步骤4具体为,设备收到增量安装/升级安装包后,分析MAINFEST.MF文件中的内容,发现组件路径上带有应用信息时,从设备上已经安装的应用中提取组件内容,并将这段内容合并进入增量安装/升级包,当所有的安装组件都在设备上已安装的应用内存在,且可以找到SHA1值也相同,则都提取并合并进入安装/升级包,最终得到对应软件的全量安装包。

一种软件安装升级的方法

技术领域

[0001] 本发明涉及无线通讯技术领域，具体地，涉及一种软件安装升级的方法。

背景技术

[0002] 安卓智能设备安装的各种软件，是通过应用商店、浏览器或PC上的助手下载、安装的。最早先的实现，无论是软件的新装还是升级，软件安装包必须完整下载。此方式对手机流量消耗较大，对服务器网络带宽资源占用也较大。

[0003] 目前普遍存在增量升级手段。比如CN 102707977 A、CN 103095838 B。这些手段在软件升级时，仅需下载新版和旧版之间的差异部分。安卓设备收到差异数据包后，根据这些差异数据以及设备上的旧安装包即可合并出新版安装包。大大减少了软件升级时的流量费用。

[0004] 现有的软件增量升级方式，由于只利用了软件前一版本的信息，没有利用设备上安装的其他软件信息。流量节省作用有限，无法节省软件首次安装时的流量占用。

[0005] CN 102707977A，CN103095838B，CN105740016A，CN104991791A和CN 105657191与本发明相比，本发明利用其他软件内组件信息，进一步缩小软件增量升级包大小。

发明内容

[0006] 本发明的目的在于，针对上述问题，提出一种软件安装升级的方法，以实现大大减少软件首次安装以及软件升级时的网络流量的优点。

[0007] 为实现上述目的，本发明采用的技术方案是：一种软件安装升级的方法，主要包括以下步骤：

步骤1：当：设备需要安装或者升级应用时，安装或升级程序不直接向服务器请求应用安装包，首先收集设备上安装的应用清单，并计算每个应用的SHA1值；

步骤2：收集完成后，安装或升级程序向服务器发送安装或升级请求；

步骤3：服务器收到安装或升级请求后，根据设备上已经安装的应用清单，以及要安装或升级的应用，计算出增量安装包或升级包；

步骤4：根据差异安装包和设备上现有应用合并出安装包或升级包，得到全量安装包；

步骤5：安装最终得到的全量安装包。

[0008] 进一步地，步骤2中，请求内容包括操作系统版本、设备上已经安装的所有应用的清单，每个应用的SHA1值和需要安装或升级的应用信息。

[0009] 进一步地，步骤3中具体为，服务器收到安装/升级请求后，利用服务器内的应用包含的组件版本数据库信息，计算增量安装包，具体包括，

当某一安装组件在设备上的所有已安装应用内均不存在或某一安装组件在设备上已安装的应用内存在，但它的SHA1值与所有已安装应用内同名组件的SHA1值均不同，则不修改MAINFEST.MF文件内对应的组件描述信息，也不从最终安装包中剔除组件信息；

当某一安装组件在设备上已安装的应用内存在，且SHA1值也相同的组件，从安装包中

剔除该组件，并修改MAINFEST.MF文件内容，将对应组件的路径信息前添加设备上存在该组件的应用名称。

[0010] 进一步地，步骤4具体为，设备收到增量安装/升级安装包后，分析MAINFEST.MF文件中的内容，发现组件路径上带有应用信息时，从设备上已经安装的应用中提取组件内容，并将这段内容合并进入增量安装/升级包，当所有的安装组件都在设备上已安装的应用内存在，且可以找到SHA1值也相同，则都提取并合并进入安装/升级包，最终得到对应软件的全量安装包。

[0011] 本发明各实施例的一种软件安装升级的方法，由于主要包括：收集设备上已经安装的应用信息，服务器根据设备上已经安装的应用清单，以及要安装的应用计算出升级包，根据差异安装包和设备上现有应用合并出安装/升级包，最终安装升级应用；从而可以克服现有技术中流量节省作用有限，无法节省软件首次安装时的流量占用，从而大大减少软件首次安装以及软件升级时的网络流量。

[0012] 本发明的其它特征和优点将在随后的说明书中阐述，并且，部分地从说明书中变得显而易见，或者通过实施本发明而了解。

[0013] 下面通过附图和实施例，对本发明的技术方案做进一步的详细描述。

附图说明

[0014] 附图用来提供对本发明的进一步理解，并且构成说明书的一部分，与本发明的实施例一起用于解释本发明，并不构成对本发明的限制。在附图中：

图1为本发明实施例所述的软件安装升级的方法流程图；

图2为安卓设备每个应用的组件组成结构图；

图3为设备上安装新应用时，设备上原本不存在的组件被下载，其他组件从其他应用提取的示意图；

图4为设备上的应用升级示意图。

具体实施方式

[0015] 以下结合附图对本发明的优选实施例进行说明，应当理解，此处所描述的优选实施例仅用于说明和解释本发明，并不用于限定本发明。

[0016] 安卓设备通常不会仅安装一个应用。如图2安卓设备往往安装了多个应用，每个应用由多个组件构成，通常，它会安装多个应用。而每个应用通常不会仅有一个组件，绝大多数情况每个应用由多个组件构成。安卓设备、应用、组件分别都会具有自身的版本信息。相同设备版本下的相同应用版本、相同组件版本的内容相同。

[0017] 不同应用有不同的版本表达方法，比如有些应用使用“ver 1.2.3”，另一些应用使用“周年庆典版”等词汇。为了便于处理，升级程序并不使用软件自己标记的版本信息，而使用软件的SHA1值作为它的版本信息。SHA1算法即安全哈希算法(Secure Hash Algorithm)。对于任意长度的输入消息，SHA1会产生一个160位的消息摘要。原始数据的任何微小变化都会让SHA1的输出值明显变化。需要注意的是，虽然本发明中采用SHA1算法作为软件版本的标记手段，换用与SHA1类似的SHA2、MD5等算法并不影响本升级方法的实质。

[0018] 软件安装包内组件及组件版本信息的提取方法是：解压该安装包，读取安装包内

META-INF目录下的MANIFEST.MF文件中描述的每个组件的SHA1摘要信息。

[0019] 结合图,1,安装升级步骤:

步骤1:当安卓设备需要安装一个应用或者升级一个应用时。安装升级程序并不直接向服务器索要应用安装包。而首先收集设备上安装的应用清单,并计算每个应用的SHA1值。

[0020] 步骤2:收集完成后,升级程序向服务器发送升级请求,请求内容包含:

操作系统版本

设备上已经安装的所有应用的清单,每个应用的SHA1值。

[0021] 需要安装或升级的应用。

[0022] 步骤3:服务器收到安装/升级请求后。利用服务器内的应用包含的组件版本数据库信息。计算增量安装包。需要被安装/升级的应用软件的组件,必在下列三种情况之内

A、某一安装组件在设备上的所有已安装应用内均不存在;

B、某一安装组件在设备上已安装的应用内存在,但它的SHA1值与所有已安装应用内同名组件的SHA1值均不同。

[0023] C、某一安装组件在设备上已安装的应用内存在,且可以找到SHA1值也相同的组件。

[0024] 如果组件的情况为A或者B,则不修改MAINFEST.MF文件内对应的组件描述信息,也不从最终安装包中剔除组件信息。对于情况为C的组件,从安装包中剔除该组件,并修改MAINFEST.MF文件内容,将对应组件的路径信息前添加设备上存在该组件的应用名称。

[0025] 通过步骤3可得到增量安装/升级安装包。

[0026] 步骤4:安卓设备收到增量安装/升级安装包后,分析MAINFEST.MF文件中的内容。发现组件路径上带有应用信息时,则从设备上已经安装的应用中提取组件内容。并将这段内容合并进入增量安装/升级包。当所有的情况为C的组件都提取并合并进入安装/升级包之后。则得到对应软件的全量安装包。

[0027] 步骤5:安装最终得到的全量安装包。即可完成软件的安装/升级。

[0028] 图3中,设备上安装新应用时,只有设备上原本不存在的组件被下载。其他组件从其他应用提取;

图4中,设备上的应用升级,也仅是设备上不存在的组件被下载,其余组件从本机其他应用获得。通常应用的前一个版本具有最多的可复用组件。

[0029] 至少可以达到以下有益效果:

1:企业内部的安卓应用发布,由于同一个企业往往采用相同的技术结构和应用程序框架。所以不同应用之间的组件重复率很高。本发明可大大减少企业多个应用发布时的网络带宽占用。

[0030] 2:安卓应用商店,由于安卓应用的生态系统日趋成熟,不同安卓应用逐渐开始同质,绝大部分应用均采用相同的优秀组件。比如PhoneGap,Unity3D,WebKit,jQuery,Xamaria这几个组件几乎覆盖所有安卓应用。跨应用提取重复组件的手段可大大减少网络流量。

[0031] 最后应说明的是:以上所述仅为本发明的优选实施例而已,并不用于限制本发明,尽管参照前述实施例对本发明进行了详细的说明,对于本领域的技术人员来说,其依然可以对前述各实施例所记载的技术方案进行修改,或者对其中部分技术特征进行等同替换。

凡在本发明的精神和原则之内，所作的任何修改、等同替换、改进等，均应包含在本发明的保护范围之内。

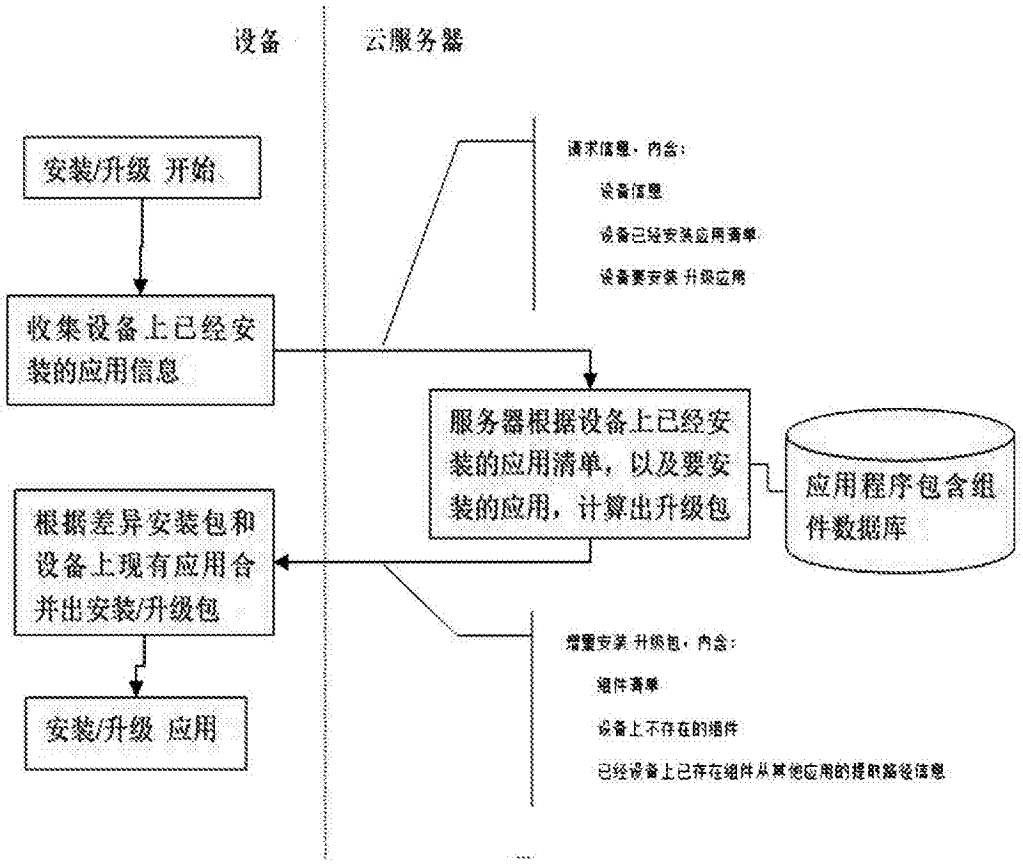

设备
云服务器
安装/升级 开始
收集设备上已经安装的应用信息
请求信息，内含：
设备信息
设备已经安装应用清单
设备要安装 升级应用
服务器根据设备上已经安装的应用清单，以及要安装的应用，计算出升级包
应用程序包含组件数据库
根据差异安装包和设备上现有应用合并出安装/升级包
安装/升级 应用
增量安装 升级包，内含：
组件清单
设备上不存在的组件
已经设备上已存在组件从其他应用的提取路径信息

图1

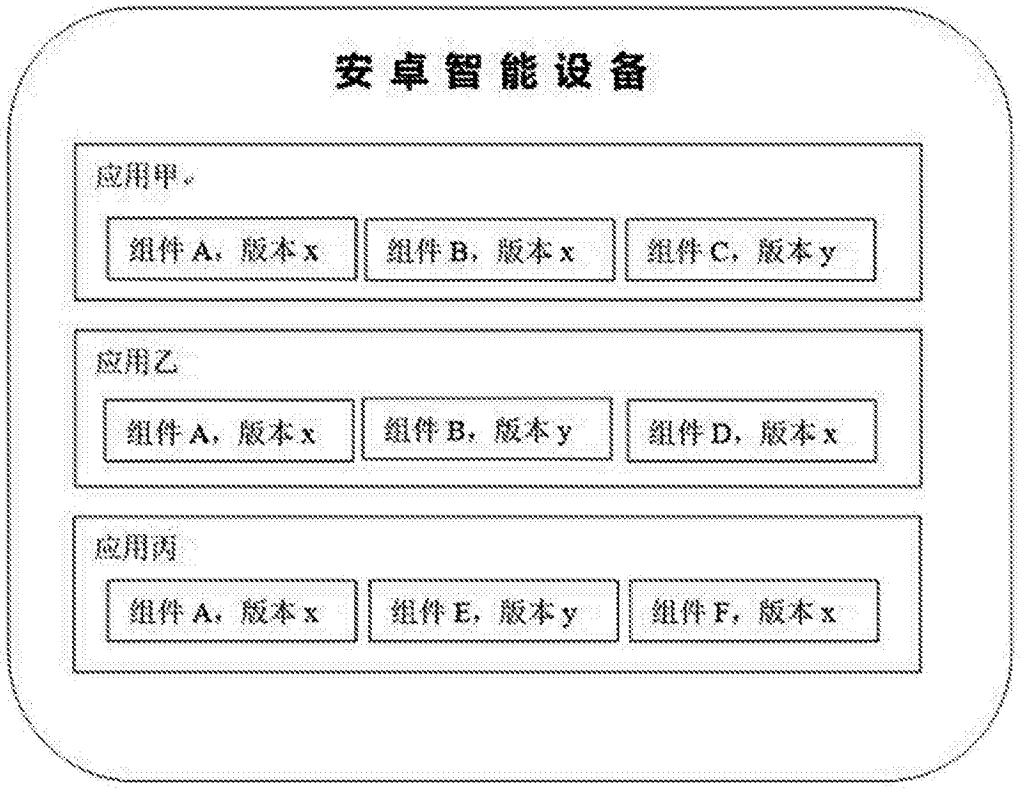

安卓智能设备
应用甲
组件A，版本x
组件B，版本x
组件C，版本y
应用乙
组件A，版本x
组件B，版本y
组件D，版本x
应用丙
组件A，版本x
组件E，版本y
组件F，版本x

图2

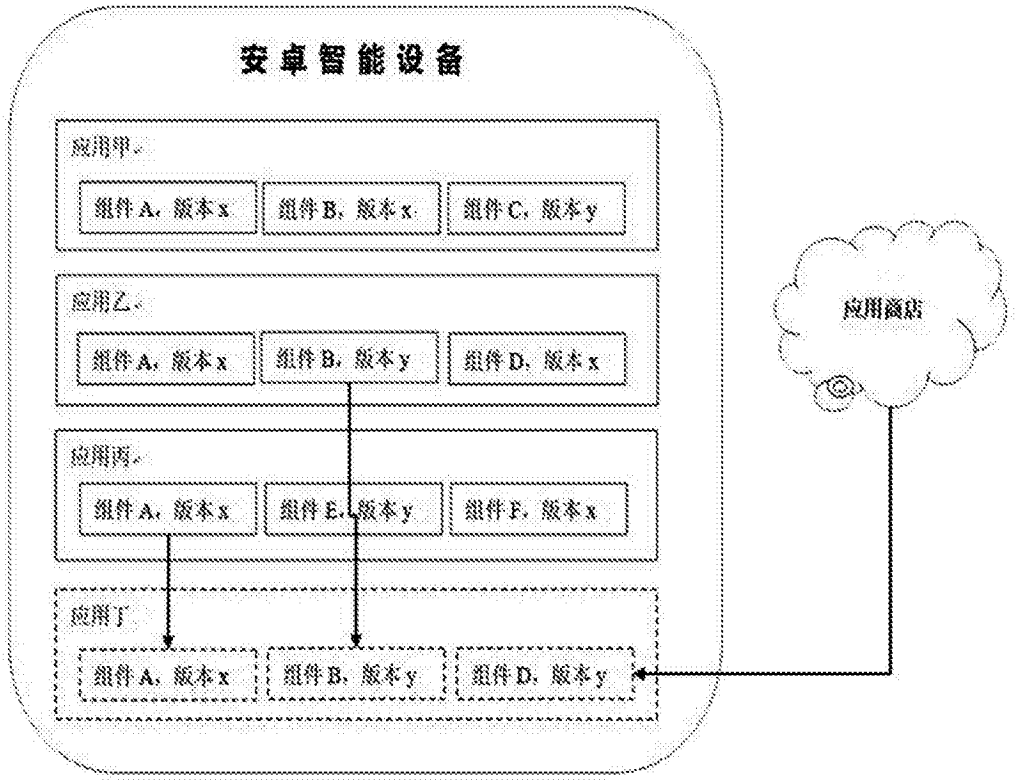

安卓智能设备
应用甲
组件A，版本x
组件B，版本x
组件C，版本y
应用乙
组件A，版本x
组件B，版本y
组件D，版本x
应用丙
组件A，版本x
组件E，版本y
组件F，版本x
应用丁
组件A，版本x
组件B，版本y
组件D，版本y
应用商店

图3

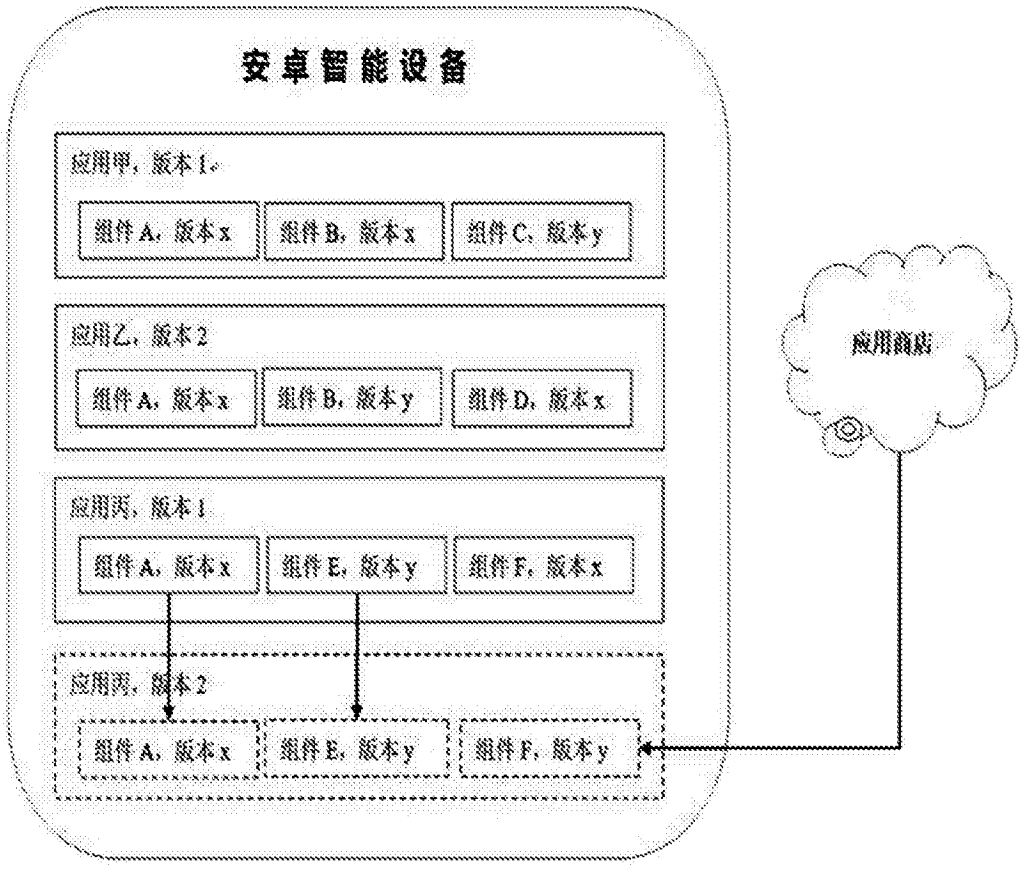

安卓智能设备
应用甲，版本1
组件A，版本x
组件B，版本x
组件C，版本y
应用乙，版本2
组件A，版本x
组件B，版本y
组件D，版本x
应用丙，版本1
组件A，版本x
组件E，版本y
组件F，版本x
应用丙，版本2
组件A，版本x
组件E，版本y
组件F，版本y
应用商店

图4