

(12) 按照专利合作条约所公布的国际申请

(19) 世界知识产权组织
国际局

(43) 国际公布日
2019 年 9 月 6 日 (06.09.2019)



(10) 国际公布号
WO 2019/165705 A1

(51) 国际专利分类号:
G06F 9/44 (2018.01)

(21) 国际申请号: PCT/CN2018/085250

(22) 国际申请日: 2018 年 5 月 2 日 (02.05.2018)

(25) 申请语言: 中文

(26) 公布语言: 中文

(30) 优先权:
201810176085.7 2018 年 3 月 2 日 (02.03.2018) CN

(71) 申请人: 平安科技(深圳)有限公司(PING AN TECHNOLOGY (SHENZHEN) CO., LTD.) [CN/CN];
中国广东省深圳市福田区八卦岭八卦三路平安大厦六楼, Guangdong 518000 (CN)。

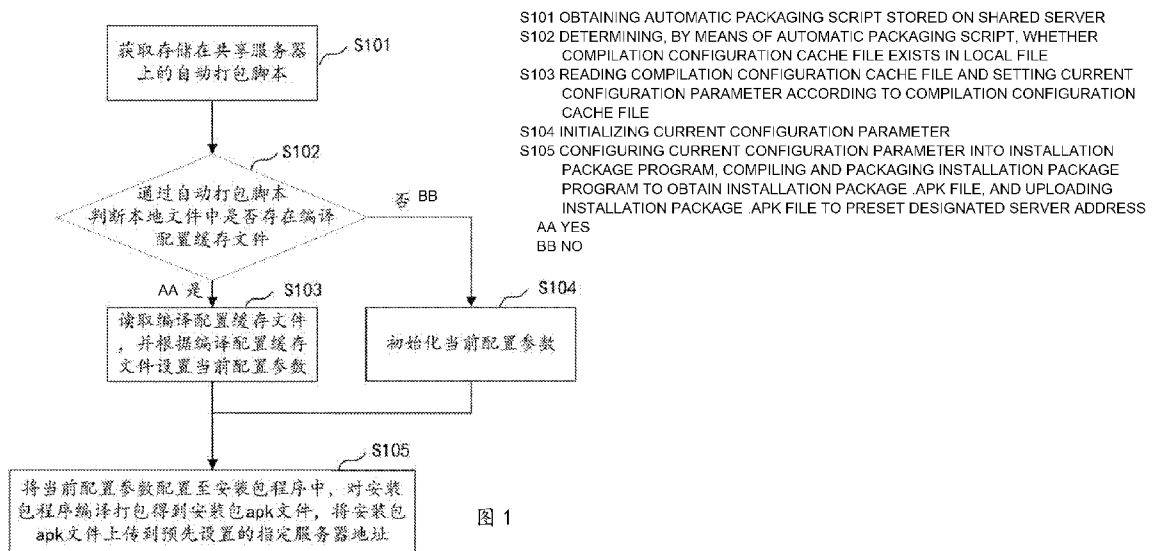
(72) 发明人: 杨文(YANG, Wen); 中国广东省深圳市福田区八卦岭八卦三路平安大厦六楼, Guangdong 518000 (CN)。

(74) 代理人: 深圳市精英专利事务所(SHENZHEN TALENT PATENT SERVICE); 中国广东省深圳市福田区深南中路 6009 号绿景广场 B 栋 20 层 B, Guangdong 518000 (CN)。

(81) 指定国(除另有指明, 要求每一种可提供的国家保护): AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BN, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DJ, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IR, IS, JO, JP, KE, KG, KH, KN, KP, KR, KW, KZ, LA, LC, LK, LR, LS, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PA, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, SA, SC, SD, SE, SG, SK, SL,

(54) Title: ANDROID APPLICATION PACKAGE UPLOAD METHOD, APPARATUS, COMPUTER DEVICE, AND STORAGE MEDIUM

(54) 发明名称: 安卓应用打包上传方法、装置、计算机设备及存储介质



(57) Abstract: Disclosed by the present application are an Android application package upload method, apparatus, computer device, and storage medium. The method comprises: obtaining an automatic packaging script stored on a shared server; determining, by means of an automatic packaging script, whether a compilation configuration cache file exists in a local file; if a compilation configuration cache file is present in the local file, then reading the compilation configuration cache file and setting a current configuration parameter according to the compilation configuration cache file; if a compilation configuration cache file is not present in the local file, then initializing the current configuration parameter; configuring the current configuration parameter into an installation package program; compiling and packaging the installation package program to obtain an installation package .apk file; uploading the installation package .apk file to a preset designated server address. The method automates packaging; it is not necessary to manually configure the parameters of a packaging tool, and there is no need to deploy a standalone server, thus packaging efficiency and success rate are improved.

SM, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG,
US, UZ, VC, VN, ZA, ZM, ZW。

- (84) 指定国(除另有指明, 要求每一种可提供的地区
保护): ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ,
NA, RW, SD, SL, ST, SZ, TZ, UG, ZM, ZW), 欧亚 (AM,
AZ, BY, KG, KZ, RU, TJ, TM), 欧洲 (AL, AT, BE, BG,
CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU,
IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT,
RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI,
CM, GA, GN, GQ, GW, KM, ML, MR, NE, SN, TD, TG)。

本国际公布:

- 包括国际检索报告(条约第21条(3))。

(57) 摘要: 本申请公开了一种安卓应用打包上传方法、装置、计算机设备及存储介质。该方法包括: 获取存储在共享服务器上的自动打包脚本; 通过自动打包脚本判断本地文件中是否存在编译配置缓存文件; 若本地文件中存在编译配置缓存文件, 读取编译配置缓存文件, 并根据编译配置缓存文件设置当前配置参数; 若本地文件中不存在编译配置缓存文件, 初始化当前配置参数; 将当前配置参数配置至安装包程序中, 对安装包程序编译打包得到安装包apk文件, 将安装包apk文件上传到预先设置的指定服务器地址。该方法实现了打包自动化, 无需手动配置打包工具的参数, 也无需部署独立服务器, 提高了打包效率和成功率。

安卓应用打包上传方法、装置、计算机设备及存储介质

本申请要求于 2018 年 3 月 2 日提交中国专利局、申请号为 201810176085.7、申请名称为“安卓应用打包上传方法、装置、计算机设备及存储介质”的中国专利申请的优先权，其全部内容通过引用结合在本申请中。

技术领域

本申请涉及安卓应用开发技术领域，尤其涉及一种安卓应用打包上传方法、装置、计算机设备及存储介质。

背景技术

目前，Android 应用（即安卓系统中的应用程序）在打包时，一般采用如下两种方案：

1) 在本地使用 Eclipse 或 Android Studio 等 IDE 打包，手动修改配置，打完包之后，手动上传到指定服务器地址（其中，Eclipse 是一个开放源代码的、基于 Java 的可扩展开发平台，也即是一个框架和一组服务，用于通过插件组件构建开发环境；Android Studio 是一个安卓开发环境，提供了集成的 Android 开发工具用于开发和调试；IDE，全称是 Integrated Development Environment，表示集成开发环境，其用于提供程序开发环境的应用程序，一般包括代码编辑器、编译器、调试器和图形用户界面工具）；

2) 使用 Jenkins（其是一个开源软件项目，是基于 Java 开发的一种持续集成工具，用于监控持续重复的工作）集成工具打包，需要独立的服务器做配置，流水线执行 job（job 即数据库中的定时任务）方式打包；

但采用 1) 中方案时，依赖依靠人工手动配置，容易出错，效率低；

采用 2) 中方案时，需要独立服务器配置，耗时耗资源，且不够灵活。

发明内容

本申请提供了一种安卓应用打包上传方法、装置、计算机设备及存储介质，旨在解决现有技术中安卓应用打包依赖依靠人工手动配置打包工具的参数或是

配置独立服务器，容易出错，效率低下的问题。

第一方面，本申请提供了一种安卓应用打包上传方法，其包括：获取存储在共享服务器上的自动打包脚本；通过自动打包脚本判断本地文件中是否存在编译配置缓存文件；若本地文件中存在编译配置缓存文件，读取编译配置缓存文件，并根据编译配置缓存文件设置当前配置参数；若本地文件中不存在编译配置缓存文件，初始化当前配置参数；将当前配置参数配置至安装包程序中，对安装包程序编译打包得到安装包 apk 文件，将安装包 apk 文件上传到预先设置的指定服务器地址。

第二方面，本申请提供了一种安卓应用打包上传装置，其包括：

脚本获取单元，用于获取存储在共享服务器上的自动打包脚本；

判断单元，用于通过自动打包脚本判断本地文件中是否存在编译配置缓存文件；

第一配置参数获取单元，用于若本地文件中存在编译配置缓存文件，读取编译配置缓存文件，并根据编译配置缓存文件设置当前配置参数；

第二配置参数获取单元，用于若本地文件中不存在编译配置缓存文件，初始化当前配置参数；

上传单元，用于将当前配置参数配置至安装包程序中，对安装包程序编译打包得到安装包 apk 文件，将安装包 apk 文件上传到预先设置的指定服务器地址。

第三方面，本申请又提供了一种计算机设备，包括存储器、处理器及存储在所述存储器上并可在所述处理器上运行的计算机程序，所述处理器执行所述计算机程序时实现本申请提供的任一项所述的安卓应用打包上传方法。

第四方面，本申请还提供了一种存储介质，其中所述存储介质存储有计算机程序，所述计算机程序包括程序指令，所述程序指令当被处理器执行时使所述处理器执行本申请提供的任一项所述的安卓应用打包上传方法。

本申请提供一种安卓应用打包上传方法、装置、计算机设备及存储介质。该方法实现了打包自动化，无需手动配置打包工具的参数，也无需部署独立服务器，提高了打包效率和成功率。

附图说明

为了更清楚地说明本申请实施例技术方案，下面将对实施例描述中所需要

使用的附图作简单地介绍，显而易见地，下面描述中的附图是本申请的一些实施例，对于本领域普通技术人员来讲，在不付出创造性劳动的前提下，还可以根据这些附图获得其他的附图。

图1为本申请实施例提供的一种安卓应用打包上传方法的示意图；

图2是本申请实施例提供的一种安卓应用打包上传方法的子流程示意图；

图3是本申请实施例提供的一种安卓应用打包上传方法的另一子流程示意图；

图4为本申请实施例提供的一种安卓应用打包上传方法的另一子流程示意图；

图5为本申请实施例提供的一种安卓应用打包上传方法的另一示意图；

图6为本申请实施例提供的一种安卓应用打包上传装置的示意性框图；

图7为本申请实施例提供的一种安卓应用打包上传装置的子单元示意性框图；

图8为本申请实施例提供的一种安卓应用打包上传装置的另一子单元示意性框图；

图9为本申请实施例提供的一种安卓应用打包上传装置的另一子单元示意性框图；

图10为本申请实施例提供的一种安卓应用打包上传装置的另一示意性框图；

图11为本申请实施例提供的一种计算机设备的示意性框图。

具体实施方式

下面将结合本申请实施例中的附图，对本申请实施例中的技术方案进行清楚、完整地描述，显然，所描述的实施例是本申请一部分实施例，而不是全部的实施例。基于本申请中的实施例，本领域普通技术人员在没有做出创造性劳动前提下所获得的所有其他实施例，都属于本申请保护的范围。

请参阅图1，图1是本申请实施例提供的一种安卓应用打包上传方法的示意图。该方法应用于台式电脑、手提电脑、平板电脑等终端中。如图1所示，该方法包括步骤S101~S105。

S101、获取存储在共享服务器上的自动打包脚本。

在本实施例中，选取shell脚本作为自动打包脚本，通过shell脚本判断本地

文件中是否存在编译配置缓存文件、将当前配置参数配置至安装包程序中、以及对安装包程序编译打包和上传服务器。其中，shell脚本与Windows/Dos下的批处理相似，也就是用各类命令预先放入到一个文件中，方便一次性执行的一个程序文件。由于是将自动打包脚本（即shell脚本）存储在共享服务器的项目工程目录下，任何需要拉取shell脚本的终端与共享服务器建立连接后，直接从项目工程目录下拉取shell脚本即可，这样同一时刻多台终端可以分别拉取shell脚本来自动打包apk文件，实现多台终端都可以在同一时刻打包，互不干扰，提高了打包效率。

S102、通过自动打包脚本判断本地文件中是否存在编译配置缓存文件。

在本实施例中，当在系统中检测到指定指令（如sh genProductBetaApk.sh命令，该指令用于激活生成apk打包）被执行时，就开始执行自动打包脚本以启动安卓应用打包。在通过执行shell脚本所启动的安装包打包时，会记录上一次打包的配置缓存文件，配置缓存文件主要包括编译的用户信息（由谁触发编译的）、编译的版本号、编译的时间、及安装包的适用环境（表示安装包是生产环境安装包还是测试环境安装包等）。如果判断存在编译配置缓存文件，则表示无需再次设置配置参数，而直接从编译配置缓存文件中直接获取即可；若不存在编译配置缓存文件，则表示需要初始化当前配置参数。本申请中通过自动打包脚本代替人工判断，自动从本地文件中拉取编译配置缓存文件，提高了配置文件读取效率。

S103、若本地文件中存在编译配置缓存文件，读取编译配置缓存文件，并根据编译配置缓存文件设置当前配置参数。

如图2所示，该步骤S103中读取编译配置缓存文件，并根据编译配置缓存文件设置当前配置参数包括以下子步骤：

S1031、获取编译配置缓存文件中所包括编译的用户信息、编译的版本号、编译的时间、及安装包的适用环境；

S1032、将编译的版本号自动加1；

S1033、将编译的时间置为当前系统时间；

S1034、将编译的用户信息置为当前编译用户信息；

S1035、将安装包的适用环境置为自动打包脚本中编译环境标识符对应的适用环境，得到更新后的编译配置缓存文件；

S1036、将当前配置参数置为更新后的编译配置缓存文件。

在一实施例中，自动打包脚本中编译环境标识符为 test，将安装包的适用环境置为测试环境安装包；自动打包脚本中编译环境标识符为 product，将安装包的适用环境置为生产环境安装包。

在本实施例中，shell 脚本每次打包完成后，在编译配置缓存文件都会记录上次打包的时间，Build 版本号(一个数字,也即编译的版本号，如 23)，编译的用户信息等在本地的缓存文件中，下次就直接读取缓存文件在上次的基础上修改编译(如上次编译的版本号是 23，程序自动加 1，将 24 作为本次编译的版本号)，无需用户手动修改，直接通过自动打包脚本来读取编译配置缓存文件即可。

S104、若本地文件中不存在编译配置缓存文件，初始化当前配置参数。

在一实施例中，如图 3 所示，所述步骤 S104 中初始化当前配置参数包括以下子步骤：

S1041、获取系统中当前编译用户信息，将编译的用户信息置为当前编译用户信息；

S1042、将编译的版本号置为 1；

S1043、获取当前系统时间，将编译的时间置为当前系统时间；

S1044、获取 shell 脚本中编译环境标识符对应的适用环境，将安装包的适用环境置为编译环境标识符对应的适用环境；

S1045、根据编译的用户信息、编译的版本号、编译的时间、及安装包的适用环境，得到初始化的当前配置参数。

在本实施例中，若本地文件中不存在编译配置缓存文件，则表示需要初始化当前配置参数。与本地文件中存在编译配置缓存文件不同之处在于，需将编译的版本号置为 1，也实现了无需用户手动修改，直接通过自动打包脚本来初始化当前配置参数。

S105、将当前配置参数配置至安装包程序中，对安装包程序编译打包得到安装包 apk 文件，将安装包 apk 文件上传到预先设置的指定服务器地址。

如图 4 所示，该步骤 S105 中将当前配置参数配置至安装包程序中，对安装包程序编译打包得到安装包 apk 文件包括以下子步骤：

S1051、获取本地存储的未进行编译的安装包程序；

S1052、将当前配置参数中所包括的编译的用户信息、编译的版本号、编译

的时间、及安装包的适用环境，对应传入安装包程序的文件夹。

在本实施例中，由于多台终端可分别拉取共享服务器上的 shell 脚本来自动打包 apk 文件，故当前终端拉取了共享服务器上的 shell 脚本并在本地获得了当前配置参数后，则需将当前配置参数编译到本地待打包且未进行编译的安装包程序中。shell 脚本识别了安装包程序中存在当前配置参数，即可进一步进行自动打包。

如图 5 所示，该步骤 S105 中对安装包程序编译打包得到安装包 apk 文件包括以下子步骤（也即步骤 S1052 之后还包括以下子步骤）：

S1053、获取安装包程序的源码，及与源码对应的依赖库源码；

S1054、定位依赖库源码目录，通过执行预设的第一指令，生成依赖库下的第一自动打包文件；

S1055、定位源码的根目录，通过执行预设的第二指令，生成源码下的第二自动打包文件；

S1056、在源码的根目录下依次进行目录清理及编译，得到未签名的 apk 文件；

S1057、获取 shell 脚本中设置的存储路径所对应的签名文件，将签名文件、第一自动打包文件、及第二自动打包文件增加至未签名的 apk 文件，得到安装包 apk 文件。

在本实施例中，预设的第一指令为 `android update lib-project -p . -t android-17`，其中 `-p .` 的意思是在当前根目录下执行，`-t android-17` 的意思是此源码使用 `android-17 api` 进行编译，上述参数可根据项目情况自行修改，所有的依赖库都要执行这一步操作。

预设的第二指令为 `android update project -p . -t android-17`，其与第一指令的不同之处在于，第一指令是更新依赖库 `lib` 以生成依赖库下的第一自动打包文件，第二指令是更新项目 `project` 中的源码以生成第二自动打包文件。

获取 shell 脚本中设置的存储路径所对应的签名文件中，所述签名文件为 `META-INF` 文件，相当于一个信息包，目录中的文件和目录获得 Java 2 平台的认可与解释，其用来配置应用程序、扩展程序、类加载器和服务 `manifest.mf` 文件，在用 `jar` 打包时自动生成。通过上述方式，实现了对安装包程序自动编译打包，而且打包过程无需部署独立服务器。

可见，该方法实现了安卓应用打包自动化，无需手动配置打包工具的参数，也无需部署独立服务器，提高了打包效率和成功率。

本申请实施例还提供一种安卓应用打包上传装置，该安卓应用打包上传装置用于执行前述任一项安卓应用打包上传方法。具体地，请参阅图 6，图 6 是本申请实施例提供的一种安卓应用打包上传装置的示意性框图。安卓应用打包上传装置 100 可以安装于台式电脑、平板电脑、手提电脑、等终端中。

如图 6 所示，安卓应用打包上传装置 100 包括脚本获取单元 101、判断单元 102、第一配置参数获取单元 103、第二配置参数获取单元 104、及上传单元 105。

脚本获取单元 101，用于获取存储在共享服务器上的自动打包脚本。

在本实施例中，选取 shell 脚本作为自动打包脚本，通过 shell 脚本判断本地文件中是否存在编译配置缓存文件、将当前配置参数配置至安装包程序中、以及对安装包程序编译打包和上传服务器。其中，shell 脚本与 Windows/Dos 下的批处理相似，也就是用各类命令预先放入到一个文件中，方便一次性执行的一个程序文件。由于是将自动打包脚本（即 shell 脚本）存储在共享服务器的项目工程目录下，任何需要拉取 shell 脚本的终端与共享服务器建立连接后，直接从项目工程目录下拉取 shell 脚本即可，这样同一时刻多台终端可以分别拉取 shell 脚本来自动打包 apk 文件，实现多台终端都可以在同一时刻打包，互不干扰，提高了打包效率。

判断单元 102，用于通过自动打包脚本判断本地文件中是否存在编译配置缓存文件。

在本实施例中，当在系统中检测到指定指令（如 sh genProductBetaApk.sh 命令，该指令用于激活生成 apk 打包）被执行时，就开始执行自动打包脚本以启动安卓应用打包。在通过执行 shell 脚本所启动的安装包打包时，会记录上一次打包的配置缓存文件，配置缓存文件主要包括编译的用户信息（由谁触发编译的）、编译的版本号、编译的时间、及安装包的适用环境（表示安装包是生产环境安装包还是测试环境安装包等）。如果判断存在编译配置缓存文件，则表示无需再次设置配置参数，而直接从编译配置缓存文件中直接获取即可；若不存在编译配置缓存文件，则表示需要初始化当前配置参数。本申请中通过自动打包脚本代替人工判断，自动从本地文件中拉取编译配置缓存文件，提高了配置文件读取效率。

第一配置参数获取单元 103, 用于若本地文件中存在编译配置缓存文件, 读取编译配置缓存文件, 并根据编译配置缓存文件设置当前配置参数。

在一实施例中, 如图 7 所示, 所述第一配置参数获取单元 103 包括以下子单元:

本地缓存文件获取单元 1301, 用于获取编译配置缓存文件中所包括编译的用户信息、编译的版本号、编译的时间、及安装包的适用环境; 自增单元 1302, 用于将编译的版本号自动加 1; 第一时间设置单元 1303, 用于将编译的时间置为当前系统时间; 第一用户信息设置单元 1304, 用于将编译的用户信息置为当前编译用户信息; 第一环境设置单元 1305, 用于将安装包的适用环境置为自动打包脚本中编译环境标识符对应的适用环境, 得到更新后的编译配置缓存文件; 参数更新单元 1306, 用于将当前配置参数置为更新后的编译配置缓存文件。

在一实施例中, 自动打包脚本中编译环境标识符为 test, 将安装包的适用环境置为测试环境安装包; 自动打包脚本中编译环境标识符为 product, 将安装包的适用环境置为生产环境安装包。

在本实施例中, shell 脚本每次打包完成后, 在编译配置缓存文件都会记录上次打包的时间, Build 版本号(一个数字, 也即编译的版本号, 如 23), 编译的用户信息等在本地的缓存文件中, 下次就直接读取缓存文件在上次的基础上修改编译(如上次编译的版本号是 23, 程序自动加 1, 将 24 作为本次编译的版本号), 无需用户手动修改, 直接通过自动打包脚本来读取编译配置缓存文件即可。

第二配置参数获取单元 104, 用于若本地文件中不存在编译配置缓存文件, 初始化当前配置参数。

在一实施例中, 如图 8 所示, 所述第二配置参数获取单元 104 包括以下子单元: 第二用户信息设置单元 1041, 用于获取系统中当前编译用户信息, 将编译的用户信息置为当前编译用户信息; 版本号初始设置单元 1042, 用于将编译的版本号置为 1; 第二时间设置单元 1043, 用于获取当前系统时间, 将编译的时间置为当前系统时间; 第二环境设置单元 1044, 用于获取 shell 脚本中编译环境标识符对应的适用环境, 将安装包的适用环境置为编译环境标识符对应的适用环境; 参数初始化单元 1045, 用于根据编译的用户信息、编译的版本号、编译的时间、及安装包的适用环境, 得到初始化的当前配置参数。

在本实施例中, 若本地文件中不存在编译配置缓存文件, 则表示需要初始

化当前配置参数。与本地文件中存在编译配置缓存文件不同之处在于，需将编译的版本号置为 1，也实现了无需用户手动修改，直接通过自动打包脚本来初始化当前配置参数。

上传单元 105，用于将当前配置参数配置至安装包程序中，对安装包程序编译打包得到安装包 apk 文件，将安装包 apk 文件上传到预先设置的指定服务器地址。

如图 9 所示，所述上传单元 105 包括以下子单元：安装包获取单元 1051，用于获取本地存储的未进行编译的安装包程序；信息传入单元 1052，用于将当前配置参数中所包括的编译的用户信息、编译的版本号、编译的时间、及安装包的适用环境，对应传入安装包程序的文件夹。

在本实施例中，由于多台终端可分别拉取共享服务器上的 shell 脚本来自动打包 apk 文件，故当前终端拉取了共享服务器上的 shell 脚本并在本地获得了当前配置参数后，则需将当前配置参数编译到本地待打包且未进行编译的安装包程序中。shell 脚本识别了安装包程序中存在当前配置参数，即可进一步进行自动打包。

如图 10 所示，所述上传单元 105 还包括以下子单元：安装包解析单元 1053，用于获取安装包程序的源码，及与源码对应的依赖库源码；第一定位单元 1054，用于定位依赖库源码目录，通过执行预设的第一指令，生成依赖库下的第一自动打包文件；第二定位单元 1055，用于定位源码的根目录，通过执行预设的第二指令，生成源码下的第二自动打包文件；未签名 apk 获取单元 1056，用于在源码的根目录下依次进行目录清理及编译，得到未签名的 apk 文件；安装包 apk 获取单元 1057，用于获取 shell 脚本中设置的存储路径所对应的签名文件，将签名文件、第一自动打包文件、及第二自动打包文件增加至未签名的 apk 文件，得到安装包 apk 文件。

在本实施例中，预设的第一指令为 `android update lib-project -p . -t android-17`，其中 `-p .` 的意思是在当前根目录下执行，`-t android-17` 的意思是此源码使用 `android-17 api` 进行编译，上述参数可根据项目情况自行修改，所有的依赖库都要执行这一步操作。预设的第二指令为 `android update project -p . -t android-17`，其与第一指令的不同之处在于，第一指令是更新依赖库 `lib` 以生成依赖库下的第一自动打包文件，第二指令是更新项目 `project` 中的源码以生成第二自动打包文

件。

获取 shell 脚本中设置的存储路径所对应的签名文件中，所述签名文件为 META-INF 文件，相当于一个信息包，目录中的文件和目录获得 Java 2 平台的认可与解释，其用来配置应用程序、扩展程序、类加载器和服务 manifest.mf 文件，在用 jar 打包时自动生成。通过上述方式，实现了对安装包程序自动编译打包，而且打包过程无需部署独立服务器。

可见，该装置实现了安卓应用打包自动化，无需手动配置打包工具的参数，也无需部署独立服务器，提高了打包效率和成功率。

上述安卓应用打包上传装置可以实现为一种计算机程序的形式，该计算机程序可以在如图 11 所示的计算机设备上运行。

请参阅图 11，图 11 是本申请实施例提供的一种计算机设备的示意性框图。该计算机设备 500 设备可以是终端。该终端可以是平板电脑、笔记本电脑、台式电脑、个人数字助理等电子设备。

参阅图 11，该计算机设备 500 包括通过系统总线 501 连接的处理器 502、存储器和网络接口 505，其中，存储器可以包括非易失性存储介质 503 和内存储器 504。

该非易失性存储介质 503 可存储操作系统 5031 和计算机程序 5032。该计算机程序 5032 包括程序指令，该程序指令被执行时，可使得处理器 502 执行一种安卓应用打包上传方法。该处理器 502 用于提供计算和控制能力，支撑整个计算机设备 500 的运行。该内存储器 504 为非易失性存储介质 503 中的计算机程序 5032 的运行提供环境，该计算机程序 5032 被处理器 502 执行时，可使得处理器 502 执行一种安卓应用打包上传方法。该网络接口 505 用于进行网络通信，如发送分配的任务等。本领域技术人员可以理解，图 11 中示出的结构，仅仅是与本申请方案相关的部分结构的框图，并不构成对本申请方案所应用于其上的计算机设备 500 的限定，具体的计算机设备 500 可以包括比图中所示更多或更少的部件，或者组合某些部件，或者具有不同的部件布置。

其中，所述处理器 502 用于运行存储在存储器中的计算机程序 5032，以实现如下功能：获取存储在共享服务器上的自动打包脚本；通过自动打包脚本判断本地文件中是否存在编译配置缓存文件；若本地文件中存在编译配置缓存文件，读取编译配置缓存文件，并根据编译配置缓存文件设置当前配置参数；若

本地文件中不存在编译配置缓存文件，初始化当前配置参数；将当前配置参数配置至安装包程序中，对安装包程序编译打包得到安装包 apk 文件，将安装包 apk 文件上传到预先设置的指定服务器地址。

在一实施例中，处理器 502 还执行如下操作：获取编译配置缓存文件中所包括编译的用户信息、编译的版本号、编译的时间、及安装包的适用环境；将编译的版本号自动加 1；将编译的时间置为当前系统时间；将编译的用户信息置为当前编译用户信息；将安装包的适用环境置为自动打包脚本中编译环境标识符对应的适用环境，得到更新后的编译配置缓存文件；将当前配置参数置为更新后的编译配置缓存文件。

在一实施例中，处理器 502 还执行如下操作：获取系统中当前编译用户信息，将编译的用户信息置为当前编译用户信息；将编译的版本号置为 1；获取当前系统时间，将编译的时间置为当前系统时间；获取 shell 脚本中编译环境标识符对应的适用环境，将安装包的适用环境置为编译环境标识符对应的适用环境；根据编译的用户信息、编译的版本号、编译的时间、及安装包的适用环境，得到初始化的当前配置参数。

在一实施例中，处理器 502 还执行如下操作：获取本地存储的未进行编译的安装包程序；将当前配置参数中所包括的编译的用户信息、编译的版本号、编译的时间、及安装包的适用环境，对应传入安装包程序的文件夹。

在一实施例中，处理器 502 还执行如下操作：获取安装包程序的源码，及与源码对应的依赖库源码；定位依赖库源码目录，通过执行预设的第一指令，生成依赖库下的第一自动打包文件；定位源码的根目录，通过执行预设的第二指令，生成源码下的第二自动打包文件；在源码的根目录下依次进行目录清理及编译，得到未签名的 apk 文件；获取 shell 脚本中设置的存储路径所对应的签名文件，将签名文件、第一自动打包文件、及第二自动打包文件增加至未签名的 apk 文件，得到安装包 apk 文件。

本领域技术人员可以理解，图 11 中示出的计算机设备的实施例并不构成对计算机设备具体构成的限定，在其他实施例中，计算机设备可以包括比图示更多或更少的部件，或者组合某些部件，或者不同的部件布置。例如，在一些实施例中，计算机设备可以仅包括存储器及处理器，在这样的实施例中，存储器及处理器的结构及功能与图 11 所示实施例一致，在此不再赘述。

应当理解，在本申请实施例中，处理器 502 可以是中央处理单元 (Central Processing Unit, CPU)，该处理器 502 还可以是其他通用处理器、数字信号处理器 (Digital Signal Processor, DSP)、专用集成电路 (Application Specific Integrated Circuit, ASIC)、现成可编程门阵列 (Field-Programmable Gate Array, FPGA) 或者其他可编程逻辑器件、分立门或者晶体管逻辑器件、分立硬件组件等。其中，通用处理器可以是微处理器或者该处理器也可以是任何常规的处理器等。

在本申请的另一实施例中提供一种存储介质。该存储介质可以为非易失性的计算机可读存储介质。该存储介质存储有计算机程序，其中计算机程序包括程序指令。该程序指令被处理器执行时实现本申请实施例的安卓应用打包上传方法。

所述存储介质可以是前述设备的内部存储单元，例如设备的硬盘或内存。所述存储介质也可以是所述设备的外部存储设备，例如所述设备上配备的插接式硬盘，智能存储卡 (Smart Media Card, SMC)，安全数字 (Secure Digital, SD) 卡，闪存卡 (Flash Card) 等。进一步地，所述存储介质还可以既包括所述设备的内部存储单元也包括外部存储设备。

所属领域的技术人员可以清楚地了解到，为了描述的方便和简洁，上述描述的设备、装置和单元的具体工作过程，可以参考前述方法实施例中的对应过程，在此不再赘述。

以上所述，仅为本申请的具体实施方式，但本申请的保护范围并不局限于此，任何熟悉本技术领域的技术人员在本申请揭露的技术范围内，可轻易想到各种等效的修改或替换，这些修改或替换都应涵盖在本申请的保护范围之内。因此，本申请的保护范围应以权利要求的保护范围为准。

权利要求书

1. 一种安卓应用打包上传方法，其特征在于，包括：

获取存储在共享服务器上的自动打包脚本；

通过自动打包脚本判断本地文件中是否存在编译配置缓存文件；

若本地文件中存在编译配置缓存文件，读取编译配置缓存文件，并根据编译配置缓存文件设置当前配置参数；

若本地文件中不存在编译配置缓存文件，初始化当前配置参数；

将当前配置参数配置至安装包程序中，对安装包程序编译打包得到安装包 apk 文件，将安装包 apk 文件上传到预先设置的指定服务器地址。

2. 根据权利要求 1 所述的安卓应用打包上传方法，其特征在于，所述读取编译配置缓存文件，并根据编译配置缓存文件设置当前配置参数，包括：

获取编译配置缓存文件中所包括编译的用户信息、编译的版本号、编译的时间、及安装包的适用环境；

将编译的版本号自动加 1；

将编译的时间置为当前系统时间；

将编译的用户信息置为当前编译用户信息；

将安装包的适用环境置为自动打包脚本中编译环境标识符对应的适用环境，得到更新后的编译配置缓存文件；

将当前配置参数置为更新后的编译配置缓存文件。

3. 根据权利要求 2 所述的安卓应用打包上传方法，其特征在于，所述初始化当前配置参数，包括：

获取系统中当前编译用户信息，将编译的用户信息置为当前编译用户信息；

将编译的版本号置为 1；

获取当前系统时间，将编译的时间置为当前系统时间；

获取 shell 脚本中编译环境标识符对应的适用环境，将安装包的适用环境置为编译环境标识符对应的适用环境；

根据编译的用户信息、编译的版本号、编译的时间、及安装包的适用环境，得到初始化的当前配置参数。

4. 根据权利要求 2 所述的安卓应用打包上传方法，其特征在于，所述将当

前配置参数配置至安装包程序中，包括：

获取本地存储的未进行编译的安装包程序；

将当前配置参数中所包括的编译的用户信息、编译的版本号、编译的时间、及安装包的适用环境，对应传入安装包程序的文件夹。

5. 根据权利要求 1 所述的安卓应用打包上传方法，其特征在于，所述对安装包程序编译打包得到安装包 apk 文件，包括：

获取安装包程序的源码，及与源码对应的依赖库源码；

定位依赖库源码目录，通过执行预设的第一指令，生成依赖库下的第一自动打包文件；

定位源码的根目录，通过执行预设的第二指令，生成源码下的第二自动打包文件；

在源码的根目录下依次进行目录清理及编译，得到未签名的 apk 文件；

获取 shell 脚本中设置的存储路径所对应的签名文件，将签名文件、第一自动打包文件、及第二自动打包文件增加至未签名的 apk 文件，得到安装包 apk 文件。

6. 一种安卓应用打包上传装置，其特征在于，包括：

脚本获取单元，用于获取存储在共享服务器上的自动打包脚本；

判断单元，用于通过自动打包脚本判断本地文件中是否存在编译配置缓存文件；

第一配置参数获取单元，用于若本地文件中存在编译配置缓存文件，读取编译配置缓存文件，并根据编译配置缓存文件设置当前配置参数；

第二配置参数获取单元，用于若本地文件中不存在编译配置缓存文件，初始化当前配置参数；

上传单元，用于将当前配置参数配置至安装包程序中，对安装包程序编译打包得到安装包 apk 文件，将安装包 apk 文件上传到预先设置的指定服务器地址。

7. 根据权利要求 6 所述的安卓应用打包上传装置，其特征在于，所述第一配置参数获取单元，包括：

本地缓存文件获取单元，用于获取编译配置缓存文件中所包括编译的用户信息、编译的版本号、编译的时间、及安装包的适用环境；

自增单元，用于将编译的版本号自动加 1；

第一时间设置单元, 用于将编译的时间置为当前系统时间;

第一用户信息设置单元, 用于将编译的用户信息置为当前编译用户信息;

第一环境设置单元, 用于将安装包的适用环境置为自动打包脚本中编译环境标识符对应的适用环境, 得到更新后的编译配置缓存文件;

参数更新单元, 用于将当前配置参数置为更新后的编译配置缓存文件。

8.根据权利要求6所述的安卓应用打包上传装置, 其特征在于, 所述上传单元, 包括:

安装包解析单元, 用于获取安装包程序的源码, 及与源码对应的依赖库源码;

第一定位单元, 用于定位依赖库源码目录, 通过执行预设的第一指令, 生成依赖库下的第一自动打包文件;

第二定位单元, 用于定位源码的根目录, 通过执行预设的第二指令, 生成源码下的第二自动打包文件;

未签名apk获取单元, 用于在源码的根目录下依次进行目录清理及编译, 得到未签名的apk文件;

安装包apk获取单元, 用于获取shell脚本中设置的存储路径所对应的签名文件, 将签名文件、第一自动打包文件、及第二自动打包文件增加至未签名的apk文件, 得到安装包apk文件。

9.根据权利要求7所述的安卓应用打包上传装置, 其特征在于, 所述第二配置参数获取单元包括以下子单元:

第二用户信息设置单元, 用于获取系统中当前编译用户信息, 将编译的用户信息置为当前编译用户信息;

版本号初始设置单元, 用于将编译的版本号置为1;

第二时间设置单元, 用于获取当前系统时间, 将编译的时间置为当前系统时间;

第二环境设置单元, 用于获取shell脚本中编译环境标识符对应的适用环境, 将安装包的适用环境置为编译环境标识符对应的适用环境;

参数初始化单元, 用于根据编译的用户信息、编译的版本号、编译的时间、及安装包的适用环境, 得到初始化的当前配置参数。

10.根据权利要求7所述的安卓应用打包上传装置, 其特征在于, 所述所述

上传单元, 包括:

安装包获取单元, 用于获取本地存储的未进行编译的安装包程序;

信息传入单元, 用于将当前配置参数中所包括的编译的用户信息、编译的版本号、编译的时间、及安装包的适用环境, 对应传入安装包程序的文件夹。

11. 一种计算机设备, 包括存储器、处理器及存储在所述存储器上并可在所述处理器上运行的计算机程序, 其特征在于, 所述处理器执行所述计算机程序时实现以下步骤:

获取存储在共享服务器上的自动打包脚本;

通过自动打包脚本判断本地文件中是否存在编译配置缓存文件;

若本地文件中存在编译配置缓存文件, 读取编译配置缓存文件, 并根据编译配置缓存文件设置当前配置参数;

若本地文件中不存在编译配置缓存文件, 初始化当前配置参数;

将当前配置参数配置至安装包程序中, 对安装包程序编译打包得到安装包 apk 文件, 将安装包 apk 文件上传到预先设置的指定服务器地址。

12. 根据权利要求 11 所述的计算机设备, 其特征在于, 所述读取编译配置缓存文件, 并根据编译配置缓存文件设置当前配置参数, 包括:

获取编译配置缓存文件中所包括编译的用户信息、编译的版本号、编译的时间、及安装包的适用环境;

将编译的版本号自动加 1;

将编译的时间置为当前系统时间;

将编译的用户信息置为当前编译用户信息;

将安装包的适用环境置为自动打包脚本中编译环境标识符对应的适用环境, 得到更新后的编译配置缓存文件;

将当前配置参数置为更新后的编译配置缓存文件。

13. 根据权利要求 12 所述的计算机设备, 其特征在于, 所述初始化当前配置参数, 包括:

获取系统中当前编译用户信息, 将编译的用户信息置为当前编译用户信息;

将编译的版本号置为 1;

获取当前系统时间, 将编译的时间置为当前系统时间;

获取 shell 脚本中编译环境标识符对应的适用环境, 将安装包的适用环境置

为编译环境标识符对应的适用环境；

根据编译的用户信息、编译的版本号、编译的时间、及安装包的适用环境，得到初始化的当前配置参数。

14.根据权利要求 12 所述的计算机设备，其特征在于，所述将当前配置参数配置至安装包程序中，包括：

获取本地存储的未进行编译的安装包程序；

将当前配置参数中所包括的编译的用户信息、编译的版本号、编译的时间、及安装包的适用环境，对应传入安装包程序的文件夹。

15.根据权利要求 11 所述的计算机设备，其特征在于，所述对安装包程序编译打包得到安装包 apk 文件，包括：

获取安装包程序的源码，及与源码对应的依赖库源码；

定位依赖库源码目录，通过执行预设的第一指令，生成依赖库下的第一自动打包文件；

定位源码的根目录，通过执行预设的第二指令，生成源码下的第二自动打包文件；

在源码的根目录下依次进行目录清理及编译，得到未签名的 apk 文件；

获取 shell 脚本中设置的存储路径所对应的签名文件，将签名文件、第一自动打包文件、及第二自动打包文件增加至未签名的 apk 文件，得到安装包 apk 文件。

16. 一种存储介质，其特征在于，所述存储介质存储有计算机程序，所述计算机程序包括程序指令，所述程序指令当被处理器执行时使所述处理器执行以下操作：

获取存储在共享服务器上的自动打包脚本；

通过自动打包脚本判断本地文件中是否存在编译配置缓存文件；

若本地文件中存在编译配置缓存文件，读取编译配置缓存文件，并根据编译配置缓存文件设置当前配置参数；

若本地文件中不存在编译配置缓存文件，初始化当前配置参数；

将当前配置参数配置至安装包程序中，对安装包程序编译打包得到安装包 apk 文件，将安装包 apk 文件上传到预先设置的指定服务器地址。

17.根据权利要求 16 所述的存储介质，其特征在于，所述读取编译配置缓存

文件，并根据编译配置缓存文件设置当前配置参数，包括：

获取编译配置缓存文件中所包括编译的用户信息、编译的版本号、编译的时间、及安装包的适用环境；

将编译的版本号自动加 1；

将编译的时间置为当前系统时间；

将编译的用户信息置为当前编译用户信息；

将安装包的适用环境置为自动打包脚本中编译环境标识符对应的适用环境，得到更新后的编译配置缓存文件；

将当前配置参数置为更新后的编译配置缓存文件。

18.根据权利要求 17 所述的存储介质，其特征在于，所述所述初始化当前配置参数，包括：

获取系统中当前编译用户信息，将编译的用户信息置为当前编译用户信息；

将编译的版本号置为 1；

获取当前系统时间，将编译的时间置为当前系统时间；

获取 shell 脚本中编译环境标识符对应的适用环境，将安装包的适用环境置为编译环境标识符对应的适用环境；

根据编译的用户信息、编译的版本号、编译的时间、及安装包的适用环境，得到初始化的当前配置参数。

19.根据权利要求 17 所述的存储介质，其特征在于，所述将当前配置参数配置至安装包程序中，包括：

获取本地存储的未进行编译的安装包程序；

将当前配置参数中所包括的编译的用户信息、编译的版本号、编译的时间、及安装包的适用环境，对应传入安装包程序的文件夹。

20.根据权利要求 16 所述的存储介质，其特征在于，所述对安装包程序编译打包得到安装包 apk 文件，包括：

获取安装包程序的源码，及与源码对应的依赖库源码；

定位依赖库源码目录，通过执行预设的第一指令，生成依赖库下的第一自动打包文件；

定位源码的根目录，通过执行预设的第二指令，生成源码下的第二自动打包文件；

在源码的根目录下依次进行目录清理及编译，得到未签名的 apk 文件；

获取 shell 脚本中设置的存储路径所对应的签名文件，将签名文件、第一自动打包文件、及第二自动打包文件增加至未签名的 apk 文件，得到安装包 apk 文件。

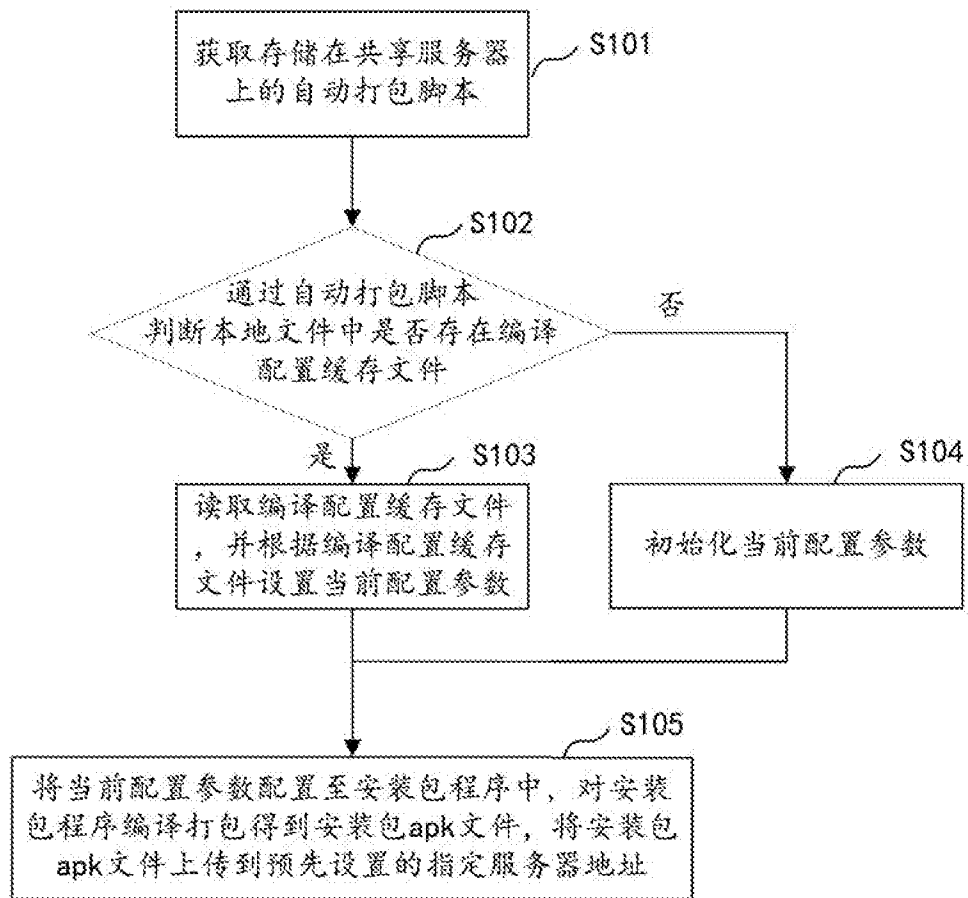


图 1

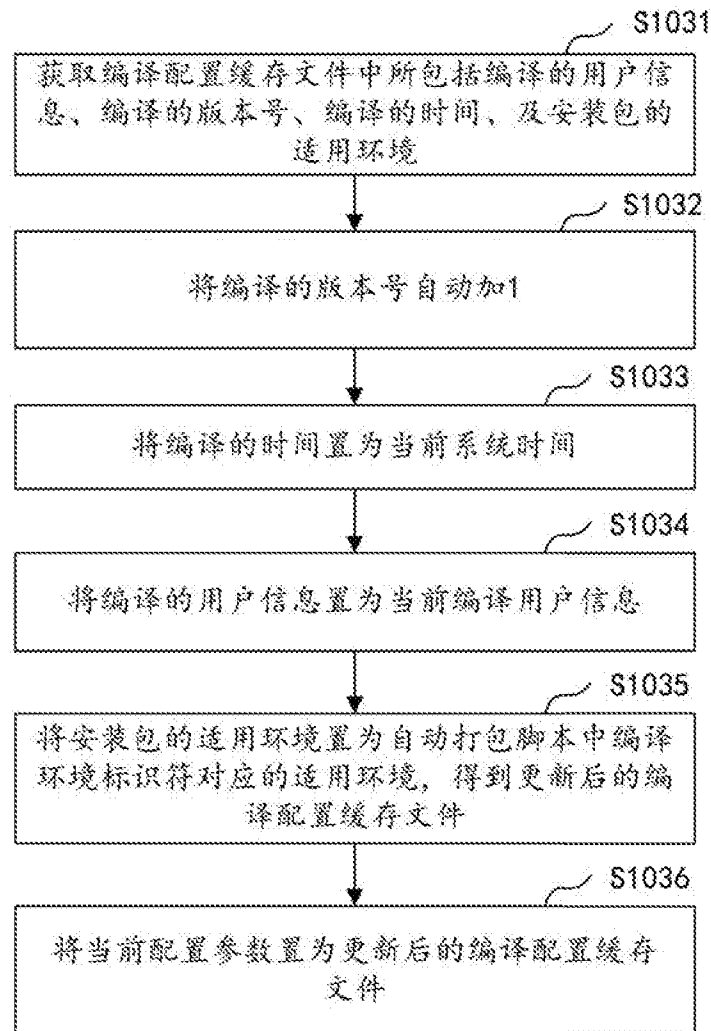


图 2

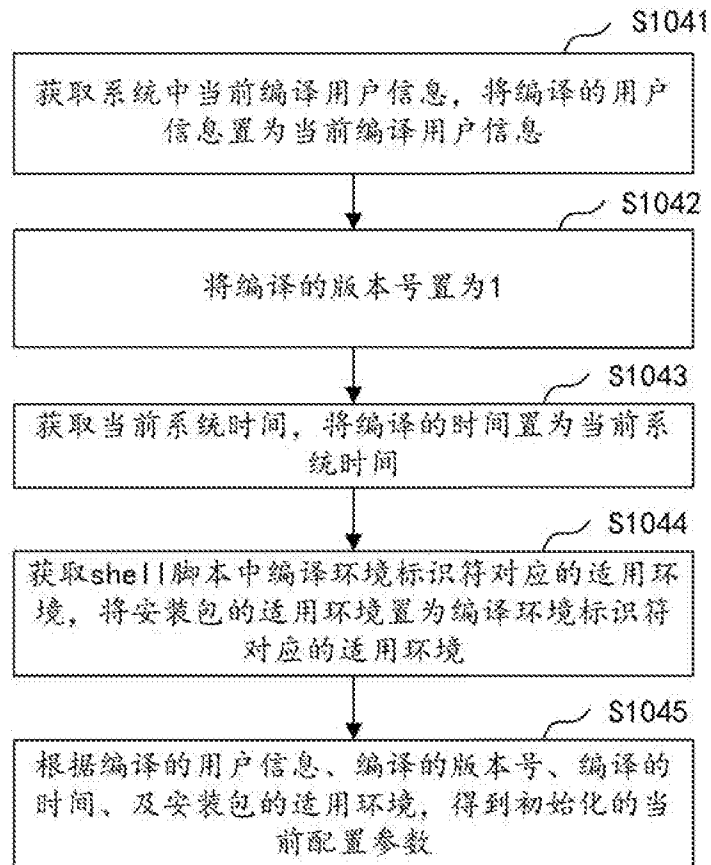


图 3

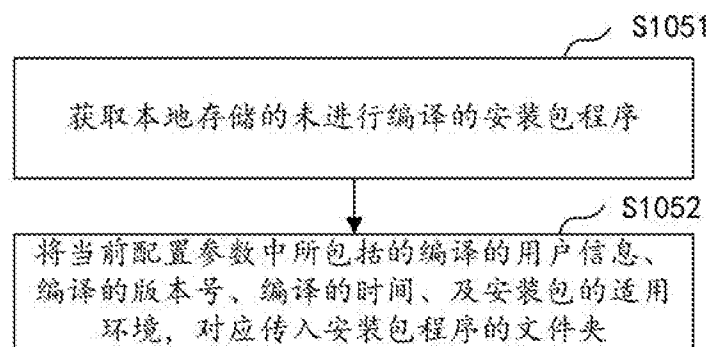


图 4

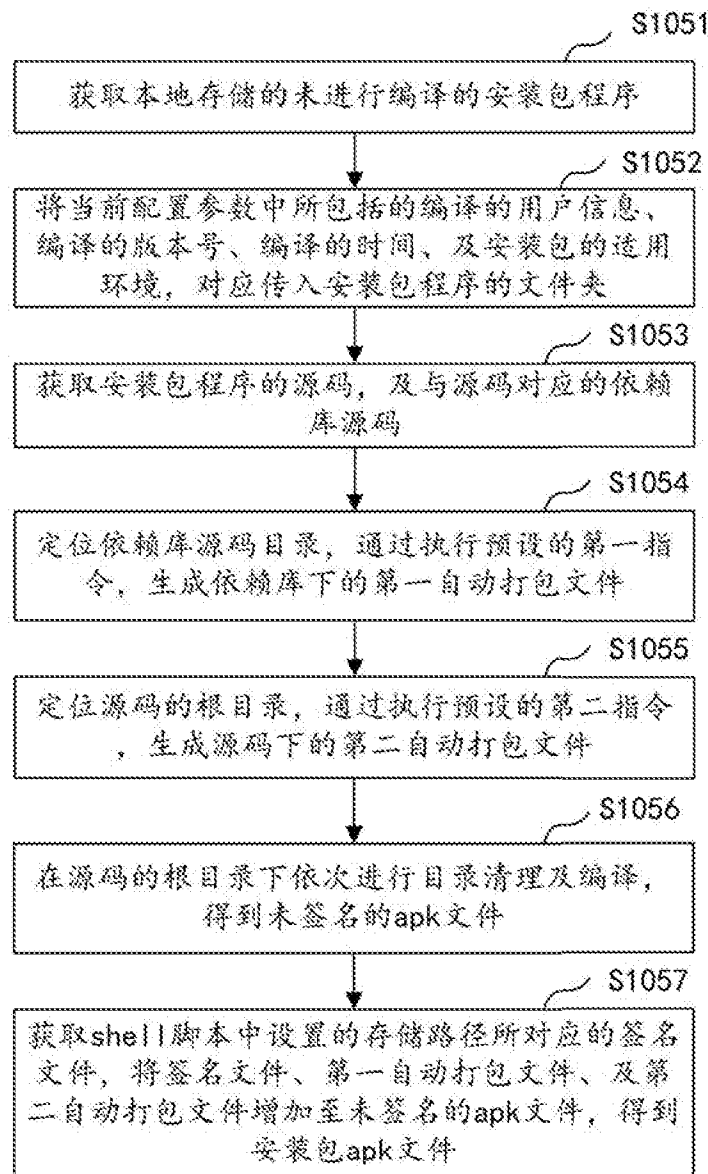


图 5

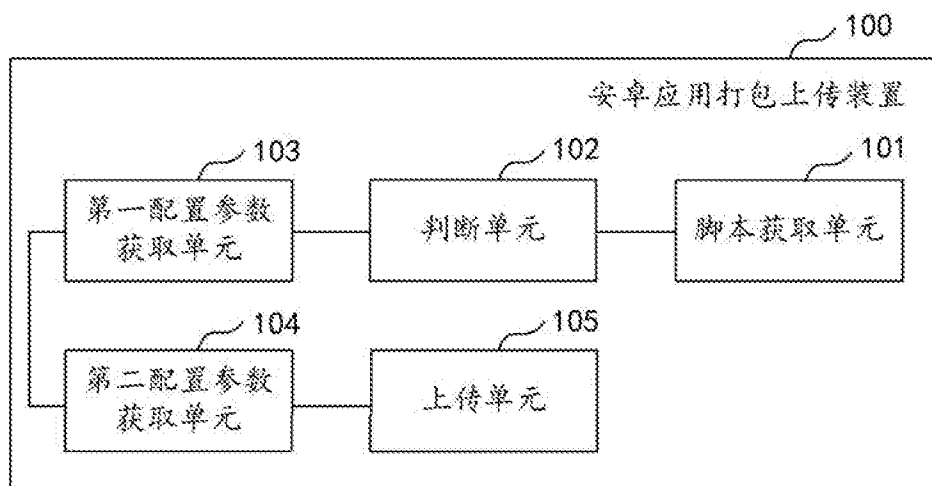


图 6

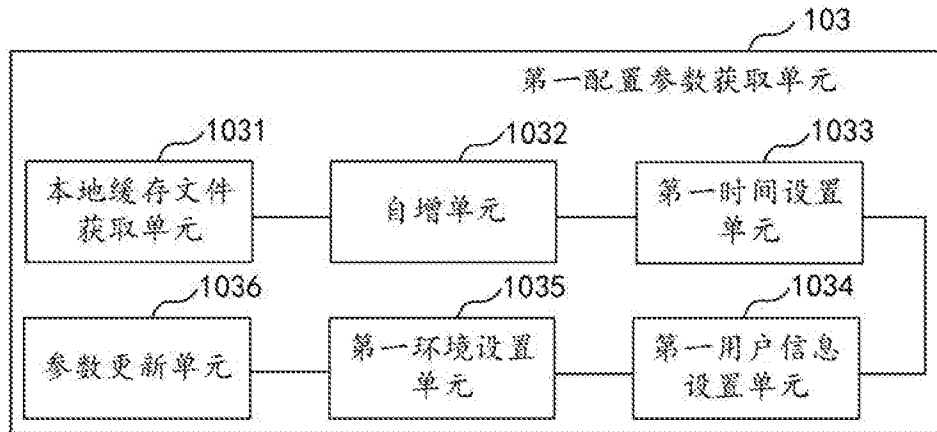


图 7

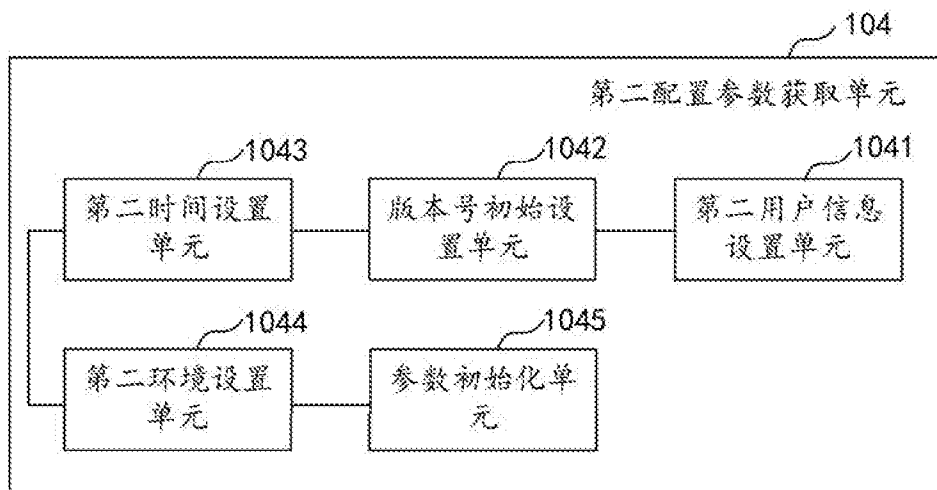


图 8

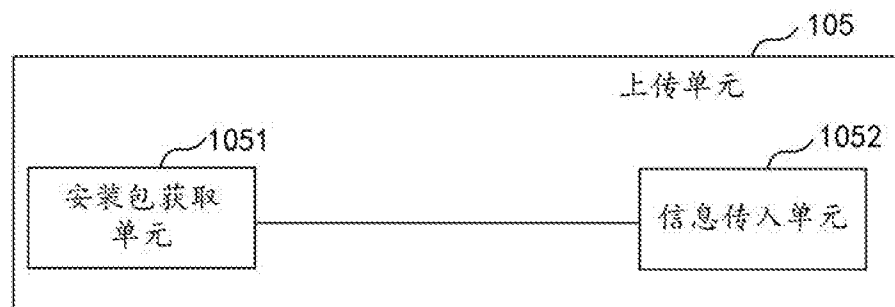


图 9

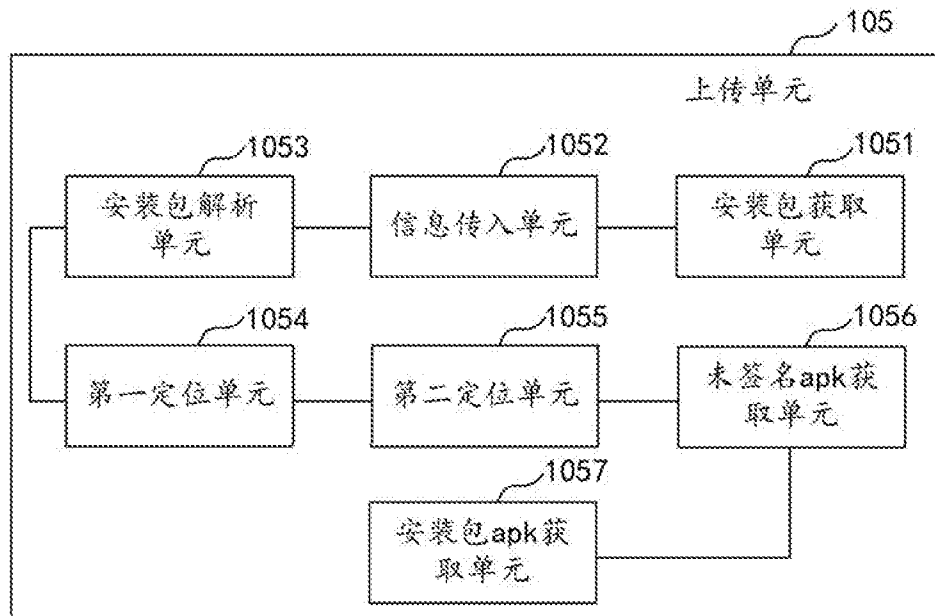


图 10

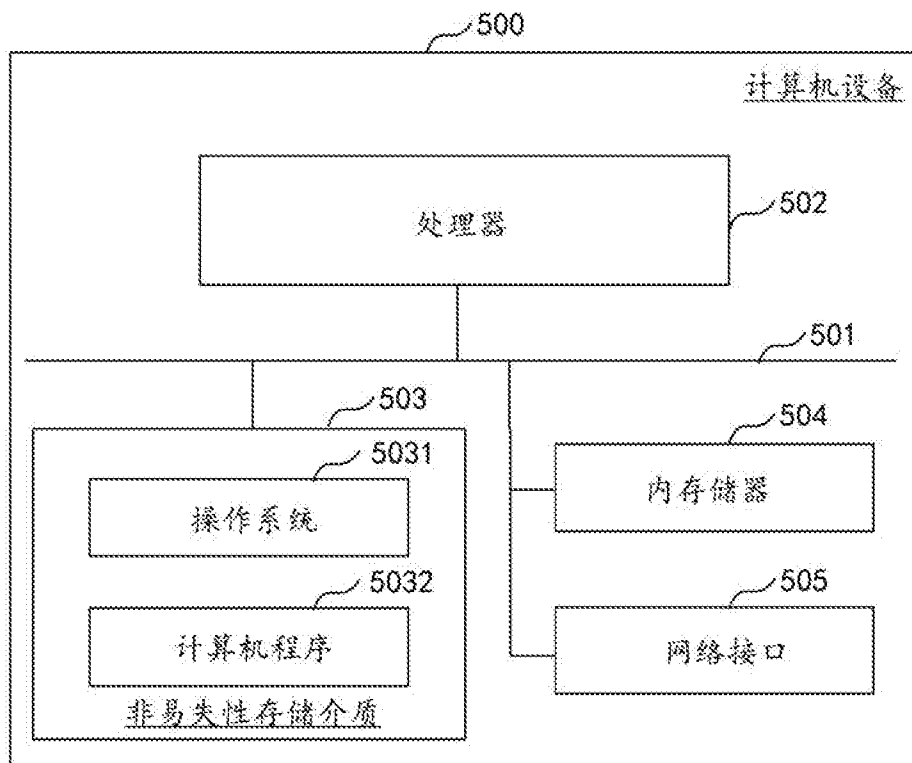


图 11

INTERNATIONAL SEARCH REPORT

International application No.

PCT/CN2018/085250

A. CLASSIFICATION OF SUBJECT MATTER

G06F 9/44(2018.01)i

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

G06F

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

CNABS, CNKI, SIPOABS, DWPI: 配置, 缓存, 安装包, 打包, 参数, configuration, cache, apk, package, parameter

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	CN 104516755 A (BOYAA ONLINE GAME DEVELOPMENT (SHENZHEN) CO., LTD.) 15 April 2015 (2015-04-15) entire document	1-20
A	CN 105468428 A (PHICOMM (SHANGHAI) CO., LTD.) 06 April 2016 (2016-04-06) entire document	1-20
A	US 2017337372 A1 (TRUSTLOOK INC) 23 November 2017 (2017-11-23) entire document	1-20



Further documents are listed in the continuation of Box C.



See patent family annex.

* Special categories of cited documents:

“A” document defining the general state of the art which is not considered to be of particular relevance

“E” earlier application or patent but published on or after the international filing date

“L” document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

“O” document referring to an oral disclosure, use, exhibition or other means

“P” document published prior to the international filing date but later than the priority date claimed

“T” later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

“X” document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

“Y” document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art

“&” document member of the same patent family

Date of the actual completion of the international search

13 June 2018

Date of mailing of the international search report

09 July 2018

Name and mailing address of the ISA/CN

State Intellectual Property Office of the P. R. China (ISA/
CN)
No. 6, Xitucheng Road, Jimenqiao Haidian District, Beijing
100088
China

Facsimile No. (86-10)62019451

Authorized officer

Telephone No.

INTERNATIONAL SEARCH REPORT
Information on patent family members

International application No.

PCT/CN2018/085250

Patent document cited in search report			Publication date (day/month/year)	Patent family member(s)	Publication date (day/month/year)
CN	104516755	A	15 April 2015	None	
CN	105468428	A	06 April 2016	None	
US	2017337372	A1	23 November 2017	None	

国际检索报告

国际申请号

PCT/CN2018/085250

A. 主题的分类 G06F 9/44 (2018.01) i 按照国际专利分类 (IPC) 或者同时按照国家分类和 IPC 两种分类		
B. 检索领域 检索的最低限度文献 (标明分类系统和分类号) G06F 包含在检索领域中的除最低限度文献以外的检索文献 在国际检索时查阅的电子数据库 (数据库的名称, 和使用的检索词 (如使用)) CNABS, CNKI, SIPOABS, DWPI: 配置, 缓存, 安装包, 打包, 参数, configuration, cache, apk, package, parameter		
C. 相关文件		
类 型*	引用文件, 必要时, 指明相关段落	相关的权利要求
A	CN 104516755 A (博雅网络游戏开发深圳有限公司) 2015年 4月 15日 (2015 - 04 - 15) 全文	1-20
A	CN 105468428 A (上海斐讯数据通信技术有限公司) 2016年 4月 6日 (2016 - 04 - 06) 全文	1-20
A	US 2017337372 A1 (TRUSTLOOK INC) 2017年 11月 23日 (2017 - 11 - 23) 全文	1-20
☐ 其余文件在C栏的续页中列出。 ☒ 见同族专利附件。		
* 引用文件的具体类型: “A” 认为不特别相关的表示了现有技术一般状态的文件 “E” 在国际申请日的当天或之后公布的在先申请或专利 “L” 可能对优先权要求构成怀疑的文件, 或为确定另一篇引用文件的公布日而引用的或者因其他特殊理由而引用的文件 (如具体说明的) “O” 涉及口头公开、使用、展览或其他方式公开的文件 “P” 公布日先于国际申请日但迟于所要求的优先权日的文件 “T” 在申请日或优先权日之后公布, 与申请不相抵触, 但为了理解发明之理论或原理的在后文件 “X” 特别相关的文件, 单独考虑该文件, 认定要求保护的发明不是新颖的或不具有创造性 “Y” 特别相关的文件, 当该文件与另一篇或者多篇该类文件结合并且这种结合对于本领域技术人员为显而易见时, 要求保护的发明不具有创造性 “&” 同族专利的文件		
国际检索实际完成的日期 2018年 6月 13日		国际检索报告邮寄日期 2018年 7月 9日
ISA/CN的名称和邮寄地址 中华人民共和国国家知识产权局 (ISA/CN) 中国北京市海淀区蓟门桥西土城路6号 100088 传真号 (86-10)62019451		授权官员 康健 电话号码 (86-10)62411639

国际检索报告
关于同族专利的信息

国际申请号

PCT/CN2018/085250

检索报告引用的专利文件			公布日 (年/月/日)	同族专利	公布日 (年/月/日)
CN	104516755	A	2015年 4月 15日	无	
CN	105468428	A	2016年 4月 6日	无	
US	2017337372	A1	2017年 11月 23日	无	