[19] 中华人民共和国国家知识产权局

[51] Int. Cl.
G06F 9/44 (2006.01)
G06F 17/30 (2006.01)
H04L 29/06 (2006.01)

[12] 发明专利申请公布说明书

[21] 申请号 200510036392.8

[43] 公开日 2007 年 2 月 7 日

[11] 公开号 CN 1908894A

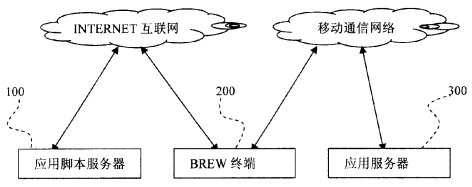
[22] 申请日 2005.8.4
[21] 申请号 200510036392.8
[71] 申请人 腾讯科技（深圳）有限公司
地址 518057 广东省深圳市振兴路赛格科技园二栋东 410 室
[72] 发明人 赵 峰 郭 永
权利要求书 3 页 说明书 13 页 附图 4 页

[54] 发明名称

基于 Brew 的一种动态用户界面的体系结构和实现方法

[57] 摘要

本发明公开了基于 Brew 的一种动态用户界面的体系结构和实现方法，涉及计算机、网络通信技术领域。该方法包括：BREW 终端通过 internet 网络从应用脚本服务器下载更新的 UI 脚本文件；BREW 终端中的 UI 脚本解释层对 UI 脚本文件进行解释，最终生成一个 UI 描述表；BREW 终端中的应用执行层从 UI 描述表入口取得 UI 描述，对 UI 进行绘制，并在 BREW 终端屏幕上显示所绘制的主程序菜单界面。对应的体系结构包括：通过 internet 网络与 BREW 终端进行通讯的应用脚本服务器，应用脚本服务器向 BREW 终端提供更新的 UI 脚本文件。本发明加快了 Brew 应用软件的开发速度；增强了 BREW 应用的扩展性和可维护性。



1、基于 BREW 的一种动态用户界面的实现方法，包括以下步骤：

1001、BREW 终端（200）通过 internet 网络从应用脚本服务器（100）下载更新的 BREW 应用的 UI 脚本文件；

1002、BREW 终端（200）中的 UI 脚本解释层（202）对所述 UI 脚本文件进行解释，对 UI 的配置数据结构进行填充，生成一个 UI 描述表；

1003、BREW 终端（200）中的应用执行层从所述 UI 描述表入口取得 UI 描述，对 UI 进行绘制，并在 BREW 终端（200）屏幕上显示所绘制的主程序菜单界面。

2、根据权利要求 1 所述的基于 BREW 的一种动态用户界面的实现方法，其特征在于，步骤 1001 中进一步包括：

2001、所述 BREW 终端（200）通过移动通信网络，从应用服务器（300）下载 BREW 应用；

2002、当 BREW 终端（200）运行所述 BREW 应用时，BREW 终端（200）采用 TCP 方式通过 internet 网络与应用脚本服务器（100）进行连接，以检查应用脚本服务器（100）中是否存在该 BREW 应用更新的 UI 脚本文件；

若存在，则 BREW 终端（200）下载该更新的 UI 脚本文件替换原有的 UI 脚本文件；

否则，BREW 终端（200）中的 UI 脚本文件保持不变。

3、根据权利要求 2 所述的基于 BREW 的一种动态用户界面的实现方法，其特征在于，步骤 2002 中进一步包括：

3001、BREW 终端（200）与应用脚本服务器（100）进行连接时，应用脚本服务器（100）对 BREW 应用的名称、有效性和安全性进行检查；

3002、在通过安全检查后，再对所述 BREW 应用的版本进行检查，若有更新的版本，则向 BREW 终端（200）发送版本升级的提示信息；

3003、BREW 终端（200）对该提示信息进行确认，并与应用脚本服务器（100）进行通讯，从而获取更新的 BREW 应用的 UI 脚本文件。

4、根据权利要求 3 所述的基于 BREW 的一种动态用户界面的实现方法，其特征在于，步骤 3003 中进一步包括：应用脚本服务器（100）根据 BREW 终端（200）的确认信息所提供的 BREW 终端平台号，选定与该 BREW 终端（200）相匹配的更新的 UI 脚本文件下发给 BREW 终端（200）。

5、根据权利要求 1 所述的基于 BREW 的一种动态用户界面的实现方法，其特征在于，步骤 1002 中 UI 脚本解释层对所述 UI 脚本文件进行解释进一步包括：UI 脚本解释层读取 UI 脚本文件到缓冲区，并对 UI 脚本文件进行词法分析和语法分析。

6、根据权利要求 1 所述的基于 BREW 的一种动态用户界面的实现方法，其特征在于，步骤 1003 中进一步包括：当有事件触发时，应用执行层通过当前 UI 描述表找到主菜单界面下对该事件的处理方法，并把描述指针指向 UI 描述表中与该处理方法对应的索引值，从而获取该状态事件的 UI 描述。

7、基于 BREW 的一种动态用户界面的体系结构，包括通过移动通信网络进行通讯的应用服务器（300）和 BREW 终端（200），所述 BREW 终端（200）可从应用服务器（300）下载 BREW 应用，所述 BREW 终端（200）包括：应用执行层（203）、BREW API 层（204）、OEM 层（205）及 BREW 终端硬件层（206），其特征在于：还包括通过 internet 网络与 BREW 终端（200）进行通讯的应用脚本服务器（100），所述应用脚本服务器（100）向 BREW 终端（200）提供更新的 BREW 应用的 UI 脚本文件。

8、根据权利要求 7 所述的基于 BREW 的一种动态用户界面的体系结构，其特征在于，所述 BREW 终端（200）还包括：UI 脚本描述层（201）和 UI 脚本解释层（202），所述 UI 脚本解释层（202）对 UI 脚本描述层（201）进行解释，生成 UI 描述表，供应用执行层（203）调用并执行。

9、根据权利要求 8 所述的基于 BREW 的一种动态用户界面的体系结构，其特征在于：所述 UI 脚本描述层（201）可由所述应用脚本服务器（100）提供更新的 BREW 应用的 UI 脚本文件来替换。

10、根据权利要求 7 所述的基于 BREW 的一种动态用户界面的体系结构，其特征在于：所述应用脚本服务器（100）对 BREW 终端（200）中 BREW 应用的名称、有效性和安全性进行检查，保证应用脚本服务器（100）与 BREW 终端（200）之间通讯的安全性。

11、根据权利要求 7 所述的基于 BREW 的一种动态用户界面的体系结构，其特征在于：所述应用脚本服务器（100）检测其中是否存在更新的 BREW 应用的 UI 脚本文件，以确认是否能对 BREW 终端（200）中的 BREW 应用版本进行升级。

12、根据权利要求 7 所述的基于 BREW 的一种动态用户界面的体系结构，其特征在于：所述应用脚本服务器（100）根据 BREW 终端平台号，选定与该 BREW 终端（200）相匹配的更新的 UI 脚本文件下发给该 BREW 终端（200）。

基于 Brew 的一种动态用户界面的体系结构和实现方法

[技术领域]

本发明涉及计算机、网络通信和电话技术领域，尤其涉及一种基于 BREW 平台的动态用户界面的体系结构和实现方法。

[背景技术]

BREW--------是无线二进制运行环境(Binary Runtime Environment for Wireless)，是高通公司 2001 年推出的基于 CDMA 网络“无线互联网发射平台”上增值业务开发运行的基本平台。目前在国内，只有中国联通的 CDMA 手机支持 BREW 平台。

“神奇宝典”BREW 业务，是中国联通推出的，基于 CDMA 1X 强大的移动通信功能，在 BREW 技术平台上，提供高速数据下载业务。用户通过下载 BREW 应用软件到 BREW 终端上运行来实现众多功能，用户的手机可以下载各种功能的软件来实现手机的个性化。通过 BREW 接口功能，供应商可以提供成套的完整的资讯、商务、娱乐 BREW 应用软件供用户下载。

“神奇宝典”BREW 业务的核心是“无线数据下载”，BREW 手机可以从网上下载游戏、动漫画、小小说等，也可进行各种在线应用，如联网游戏、收发邮件、证券炒股、信息查询等。

基于 BREW 平台的各种 BREW 应用，都需要经过开发、测试、业务审核、上传高通服务器和做目录等几个环节，才能供 BREW 手机用户下载使用。

在现有技术中，BREW 平台下应用体系结构模型如图 1 所示，BREW 终端（200）如果下载或是更新升级软件只能通过移动通信网络与应用服务器（300）进行通信来实现。其中，应用服务器（300）是由高通公司提供的，软件开发商开发出来的 BREW 应用软件经过严格的测试(为了保证 BREW 应用及 BREW 终端的正常运行而必须进行的测试)后，才能上传到该服务器上，供 BREW 终端（200）下载使用。这样的体系结构就决定了：软件开发商提交一个新的应用上线都要经过很多程序，比如开发、内测、UBT 测试、业务评

审、上传应用服务器、做目录等步骤，才能与 BREW 手机用户见面，从而花费了很多的精力和时间；如果软件开发商希望对已经上线的 BREW 应用进行软件升级，同样需要走相同烦琐的流程。

如图 2 所示，传统的 BREW 终端程序包括应用执行层（203）、BREW API 层（204）、OEM 层（205）和 BREW 终端硬件层（206）；BREW 应用软件中的可执行程序和 UI 脚本文件都固化在应用执行层（203）中。

BREW 终端硬件层（206）由硬件芯片组成，完成特定的功能，比如硬件支持语音通话和短信等功能，位于体系结构的最底层；

OEM 层（Original Equipment Manufacture 原始设备制造商）（205），是由手机开发厂家开发的介于 BREW API 层（204）和 BREW 终端硬件层（206）之间的一个层，负责驱动和管理硬件设备。同时把硬件的使用和操作进行封装，使上层不必关心具体如何管理和操作硬件的细节；提供 OEM 接口供 BREW API 层（204）调用。

BREW API 层(204)是按高通公司的要求统一实现的接口(Application Programming Interface 应用程序编程接口)，方便 BREW 应用开发商在 BREW API 层（204）上进行软件和应用开发，该接口与手机机型和屏幕等硬件参数无关，可以方便进行移植开发等。

应用执行层（203），就是开发商基于 BREW 平台具体开发出的应用，可以在 BREW 平台上运行执行，完成软件功能。

目前的 BREW 应用开发主要存在以下缺点：

1、BREW 应用程序的 UI（User Interface 用户界面）和功能逻辑是固定不能改变的，是固化在应用程序的执行文件中的。如果软件开发者想改变 BREW 应用的 UI 和功能逻辑，或是要修改在 BREW 应用程序中发现的 BUG（程序中微小的错误），除了重新测试提交升级版本之外，再没有其他的办法可用，这就导致 BREW 应用的扩展性和可维护性差。

2、开发中重复工作过多。由于 BREW 应用上线需要通过联通的 UBT 测试，UBT 测试是有一套严格的规范的，尤其是对 UI 部分和功能方面都有很高的要求，所以在 UI 设计开发中有很多重复的工作，比如积分模块，帮助和关于模块的开发，几乎都是每个 BREW 应用必须要面对的工作。每个应用都重新开发相同的模块，增加了潜在的系统不稳定性。

3、对已经上线的 BREW 应用进行升级，也需要经过和新应用上线一样流程，效率很低，影响 BREW 应用升级的速度，BREW 手机用户不能更快体验到升级版软件所带来的新

的乐趣；若发现 BREW 应用中存在新的问题，软件开发商也不能对 BREW 手机用户的要求做出及时的反应。

[发明内容]

本发明要解决的技术问题是提供基于 Brew 的一种动态用户界面的体系结构和实现方法，一方面，避免了在 Brew 应用软件开发中，对基本模块的重复开发工作，加快 Brew 应用软件开发的速度，且能够提高系统的稳定性；另一方面，避免了对已经上线的 BREW 应用进行升级所必经的烦琐流程，增强了 BREW 应用的扩展性和可维护性，使 BREW 手机用户能在第一时间体验到升级版软件所带来的新的乐趣。

本发明是通过下面的技术方案来实现的：

基于 BREW 的一种动态用户界面的实现方法，包括以下步骤：

1001、BREW 终端通过 internet 网络从应用脚本服务器下载更新的 BREW 应用的 UI 脚本文件；

1002、BREW 终端中的 UI 脚本解释层对所述 UI 脚本文件进行解释，对 UI 的配置数据结构进行填充，生成一个 UI 描述表；

1003、BREW 终端中的应用执行层从所述 UI 描述表入口取得 UI 描述，对 UI 进行绘制，并在 BREW 终端屏幕上显示所绘制的主程序菜单界面。

步骤 1001 中进一步包括：

2001、所述 BREW 终端通过移动通信网络，从应用服务器下载 BREW 应用；

2002、当 BREW 终端运行所述 BREW 应用时，BREW 终端采用 TCP 方式通过 internet 网络与应用脚本服务器进行连接，以检查应用脚本服务器中是否存在该 BREW 应用更新的 UI 脚本文件；

若存在，则 BREW 终端下载该更新的 UI 脚本文件替换原有的 UI 脚本文件；

否则，BREW 终端中的 UI 脚本文件保持不变。

步骤 2002 中进一步包括：

3001、BREW 终端与应用脚本服务器进行连接时，应用脚本服务器对 BREW 应用的名称、有效性和安全性进行检查；

3002、在通过安全检查后，再对所述 BREW 应用的版本进行检查，若有更新的版

本，则向 BREW 终端发送版本升级的提示信息；

3003、BREW 终端对该提示信息进行确认，并与应用脚本服务器进行通讯，从而获取更新的 BREW 应用的 UI 脚本文件。

步骤 3003 中进一步包括：应用脚本服务器根据 BREW 终端的确认信息所提供的 BREW 终端平台号，选定与该 BREW 终端相匹配的更新的 UI 脚本文件下发给 BREW 终端。

步骤 1002 中 UI 脚本解释层对所述 UI 脚本文件进行解释进一步包括：UI 脚本解释层读取 UI 脚本文件到缓冲区，并对 UI 脚本文件进行词法分析和语法分析。

步骤 1003 中进一步包括：当有事件触发时，应用执行层通过当前 UI 描述表找到主菜单界面下对该事件的处理方法，并把描述指针指向 UI 描述表中与该处理方法对应的索引值，从而获取该状态事件的 UI 描述。

基于 BREW 的一种动态用户界面的体系结构，包括通过移动通信网络进行通讯的应用服务器和 BREW 终端，所述 BREW 终端可从应用服务器下载 BREW 应用，所述 BREW 终端包括：应用执行层、BREW API 层、OEM 层及 BREW 终端硬件层，还包括通过 internet 网络与 BREW 终端进行通讯的应用脚本服务器，所述应用脚本服务器向 BREW 终端提供更新的 BREW 应用的 UI 脚本文件。

本发明的进一步改进在于，所述 BREW 终端还包括：UI 脚本描述层和 UI 脚本解释层，所述 UI 脚本解释层对 UI 脚本描述层进行解释，生成 UI 描述表，供应用执行层调用并执行。

本发明的进一步改进在于：所述 UI 脚本描述层可由所述应用脚本服务器提供更新的 BREW 应用的 UI 脚本文件来替换。

本发明的进一步改进在于：所述应用脚本服务器对 BREW 终端中 BREW 应用的名称、有效性和安全性进行检查，保证应用脚本服务器与 BREW 终端之间通讯的安全性。

本发明的进一步改进在于：所述应用脚本服务器检测其中是否存在更新的 BREW 应用的 UI 脚本文件，以确认是否能对 BREW 终端的 BREW 应用版本进行升级。

本发明的进一步改进在于：所述应用脚本服务器根据 BREW 终端平台号，选定与该 BREW 终端相匹配的更新的 UI 脚本文件下发给该 BREW 终端。

由于采用了以上技术方案，把 UI 脚本描述层和 UI 脚本解释层从应用执行层中分离

出来，在对 BREW 应用软件进行升级时，BREW 终端只需从应用脚本服务器下载更新的 UI 脚本文件替换掉原有的 UI 脚本文件，即可实现该 BREW 应用软件的升级，这就避免了传统 BREW 应用软件进行升级所必经的如：UBT 测试等烦琐的流程，加快了 BREW 应用升级软件的开发速度，节约了时间，使 BREW 终端用户可以及时方便地体验 BREW 应用升级软件带来的乐趣，同时增加了 BREW 应用的自由度、可扩展性；在进行 BREW 平台应用程序的开发中，可以使用容易书写且易于理解的 UI 脚本来实现，不需要重新编写程序的代码；在跨平台移植 BREW 应用时更加方便，此时仅仅需要修改执行文件中的 UI 脚本解释层，而不需要更新 UI 脚本文件，即可实现 BREW 应用的移植。

[附图说明]

图 1 是现有的 BREW 平台下的应用体系结构模型图。

图 2 是传统的 BREW 终端程序模块图。

图 3 是本发明的动态用户界面结构体系模型图。

图 4 是本发明的 BREW 终端程序模块图。

图 5 是一个标准的 BREW 游戏主菜单界面图。

图 6 是 UI 脚本解释层的工作流程图。

图 7 是动态用户界面实现的流程图。

[具体实施方式]

下面根据附图和具体实施例对本发明作进一步地阐述。

如图 3 所示，本发明的动态用户界面结构体系图是在现有技术的基础之上，增加了应用脚本服务器 100，其中，应用脚本服务器 100 与 BREW 终端 200 之间的通讯是通过 internet 互联网来完成的，应用脚本服务器 100 中存放有更新的 UI 脚本文件供 BREW 终端 200 进行下载。

首先，BREW 终端 200 通过移动通信网络，如 CDMA 网络，从应用服务器 300 上选择下载 BREW 应用到本地；当 BREW 终端 200 运行所下载的 BREW 应用时，BREW 终端 200 通过 INTERNET 网络，使用 TCP 的方式连接应用脚本服务器 100，同时应用脚本服务器 100 对 BREW 应用的名称、有效性和安全性进行检查，保证应用脚本服务

器与 BREW 终端之间通讯的安全性，防止其他组织或者个人非法获取和访问 UI 脚本文件的内容；在通过安全检查后，应用脚本服务器 100 检查其中是否存在更新的 BREW 应用版本；

若有更新的版本，则向 BREW 终端 200 发送版本升级的提示信息；在 BREW 终端 200 对该提示信息进行确认后，应用脚本服务器 100 根据该确认信息中携带的终端平台号，选择与该 BREW 终端 200 相匹配的更新的 UI 脚本文件下发给 BREW 终端 200，BREW 终端 200 接收该更新的 UI 脚本文件，并把原有的 UI 脚本文件替换成该更新的 UI 脚本文件，从而实现了 BREW 应用软件的升级。

若没有更新的版本，则 BREW 终端 200 中原有的 UI 脚本文件保持不变，BREW 应用软件没有得到升级。

本发明的 BREW 终端程序模块如图 4 所示，除了包括传统的 BREW 终端程序模块，即应用执行层 203、BREW API 层 204、OEM 层 205 和 BREW 终端硬件层 206，还包括 UI 脚本描述层 201 和 UI 脚本解释层 202，而在传统 BREW 终端程序模块中，UI 脚本描述层 201 和 UI 脚本解释层 202 的功能是固化在应用执行层 203 中的，因此想对 BREW 应用程序进行升级会存在很大的难度，而且必须提供完整的 BREW 终端升级程序才能实现。本发明是把 UI 脚本描述层 201 和 UI 脚本解释层 202 从应用执行层 203 中分离出来，在进行 BREW 应用软件的升级时，只需替换原有的 UI 脚本描述层 201 即可实现；在 UI 脚本描述层 201 与应用执行层 203 之间增加了 UI 脚本解释层 202，减少了 UI 和逻辑之间的耦合性，更好地实现了 UI 的动态加载，换言之，在修改程序时，不需要把每个层都重写或是修改，仅仅是需要修改 UI 脚本解释层 202 或是修改应用执行层 203，或是修改 UI 脚本文件（UI 脚本描述层 201），对某个层的修改不会影响到其他层的正常工作。

UI 脚本解释层 202 对 UI 脚本描述层 201 进行解释，生成对 UI 的描述，按协议对 UI 的配置数据结构进行填充，最终会生成一个以 UI 的状态为主键的表，该表一直在内存中有效并且保持到应用程序关闭。主键就是用来唯一标识一个记录的关键字。

应用执行层 203 针对表中的描述，绘制应用 UI，当事件或是按键触发时，执行模块就会按照当前的状态查找表中对当前事件或按键的描述，执行相应动作或是跳转到相应的 UI。

在脚本文件中描述一个软件的 UI 和逻辑，由解释层解释，再由执行层执行。这样，我们改变脚本描述就达到了改变软件的 UI 和逻辑的目的，如一个界面由背景，菜单，按钮组成，原脚本文件描述了 5 个菜单，2 个按钮，我们通过修改脚本文件，描述为 6 个菜单和 3 个按钮，就很方便的改变了软件的界面。

如果我们进行软件升级或是软件的跨平台移植，不希望对软件界面进行修改，只需要对应用的 UI 脚本解释层 202 进行修改，而不必修改 UI 脚本文件，把同样的 UI 脚本文件解释成不同操作系统自己所能理解代码，交由应用执行层 203 执行。

BREW 终端的软件安装包中包括目标执行文件*.mod 文件，应用所需的资源文件*.bar，动态加载的 UI 脚本文件*.tsp 文件等。

下面对 UI 脚本描述层 201 和 UI 脚本解释层 202 进行详细的介绍：

1、UI 脚本描述层 201：

Tsp 文件就是实现了 UI 脚本描述层 201 的功能，Tsp 文件就是 UI 脚本文件，该文件具体的描述了 UI 的元素例如按钮、菜单、文字提示、图形、输入框等控件的属性和显示位置等各种属性，该文件还描述了 UI 的状态切换，即对各种事件或按键触发进行动作或是进行响应等。程序启动后就会首先读取脚本数据，重构程序 UI 和逻辑等。

以一个简单的脚本例子可以更好的理解该发明，如图 5 所示，是一个标准的 BREW 游戏主菜单界面图，这个界面图的 Tsp 文件描述如下：

```
<TSP version="1.0">
<name>TestGame</name>
……
```

在这部分主要描述 BREW 应用的版本号，以便以后自动升级的时候比较版本，和 BREW 应用的名称。

```
<Status>C_MAINMENU
    <background>
        <image>
            <path>Res</path>
            <title>R_I_IMAGE<title>
            <x>0</x>
```

```
            <y>0</y>
        </image>
        ……
    </background>
    <menu>
            <text>”开始游戏”</text>
            <x>x_Center</x>
            <y>y_Center</y>
        ……
    </menu>
    <button>
        <text>”进入”</text>
        <x>x_Left_Menu</x>
        <y>y_Menu</y>
        <text>”退出”</text>
        <x>x_Right_Menu</x>
        <y>y_Menu</y>
    </button>
    <onEvent>
        <avk_soft1>
            <onstatus>C_STATUS_START</onstatus>
            <onstatus>C_STATUS_SET</onstatus>
            <onstatus>C_STATUS_ HELP </onstatus>
            <onstatus>C_STATUS_ ABOUT </onstatus>
        </avk_soft1>
        ……
    </onEvent>
    ……
    </status>
```

该脚本简单描述了有4个菜单（开始游戏、游戏设定、游戏帮助、游戏关于）和2个按钮（进入、退出）的UI及各种触发事件的处理逻辑。

2、UI脚本解释层202：

在目标执行文件*.mod文件中，除了包含应用逻辑和计算以外，还包括了自主开发的UI脚本解释层202，UI脚本解释层202对UI脚本描述层201进行解释，供应用执行层203来执行以实现UI的动态加载。

如对背景进行解释，其过程为：

1、根据UI脚本的描述，比如背景是图片还是文字，如果是图片还需要路径信息，是从资源文件*.bar中读取，还是直接从文件中读取。在BREW中，有专用的BAR资源文件打包工具打包，会把图片，音频和文字等打包到一个文件中，读取的时候需要按资源ID来读取，其好处是资源统一管理；但是有的资源可以是动态加载的，为实现动态用户界面，可能需要更新一部分资源文件，但是这时候已经不可能重新打包*.bar文件了，通过动态从应用脚本服务器100下载我们需要的资源图片文件到本地来读取，这就是后者所提到的直接从文件读取。这个文件可以是一个脚本配置文件，可以是一个图片格式的文件，可以是音乐格式的文件，也可以是其他资源文件等。

2、再根据UI脚本中对背景的描述位置，来绘制应用的背景的位置。

3、然后再在背景上绘制菜单项。菜单项一般包含菜单的背景贴图信息和菜单文字显示，文字颜色，绘制位置等信息。

4、接着根据〈BUTTON〉描述绘制出按钮。

5、最后根据onEvent字段处理页面的按键响应逻辑，当事件产生时是如何跳转页面，如何切换状态等。

如图6所示，是UI脚本解释层202的工作流程图。UI脚本解释层202的功能是：对UI脚本文件进行解释和分析，并最终生成一个UI描述表。其具体过程如下：

当BREW应用启动时，UI脚本解释层202把UI脚本文件读入缓冲区，即执行步骤401；对UI脚本文件进行词法分析和语法分析，即执行步骤402和403；最终生成一个UI描述表，即步骤404。

词法分析402是指原文件被分解为词法符号的过程，词法分析按照执行的先后次序分为六步：源字符映射，转义换行替换，空白的划分，目标字符映射以及相邻字符串连

接，从词法分析中得到一个正确的单词序列。

语法分析 403 的主要任务就是，根据程序设计语言的语法规则，将词法分析器所提供的单词符号串分析成各种语法范畴。从单词到短语，从短语到语句，从语句到程序段或程序，分析和确定给出的单词符号串是否组成一个正确的程序。

若分析中发现有错，则进行相应的出错处理。其中，这里的错误可以分为两级错误。

第一级：严重的系统错误，提示用户后，按任意键退出应用程序。

第二级：低级别的错误，提示用户后，返回到离当前界面比较近的或是功能菜单的入口处。

综上所述，如果能得到正确的脚本信息，就能通过自带的解释器来解析 UI 信息，重构应用程序的 UI 和逻辑。

UI 脚本解释层 202 解释的过程就是填充 UI 描述表的过程，把脚本写到缓冲区，通过字符串查找和内存接口把解释出来的 UI 描述项填充到相应的 UI 描述表项。

UI 描述表项如下所示：

每一条 UI 描述记录都由以下参数组成：

char iRunStatus //UI 的状态 ,也是应用执行模块查表的各界面的入口和主键值。

char iBackground //UI 的背景模式 ，表示是否更新背景、还是整屏重绘还是局部更新等。

char iImage //UI 的贴图，是否贴图标志。

char iPath //UI 贴图的来源是从资源文件还是直接从文件中读取图片。

char *title // 如果从资源文件读取，该字段为资源文件中图片的索引值；如果为直接文件读取则为文件 URL。

char ix //要贴图片的左上角的 X 坐标。

char iy //要贴图片的左上角的 Y 坐标。

char iMenu //是否有菜单项。

char *pText //菜单显示的名称。

char ix //菜单文字的显示位置的 X 坐标。

char iy //菜单文字的显示位置的 Y 坐标。

char iButton //是否有按钮标志。

char *pText //按钮的显示名称。

char ix //按钮文字的显示位置的 X 坐标。

char iy //按钮文字的显示位置的 Y 坐标。

char iOnEvent //事件或是按键处理。

char iCondition //事件 ID 或按键值。

char iAction //事件或按键响应描述。

若有完备的上述 UI 描述表项，用户界面就可以重构。UI 描述表由上述的结构体类型的结构体数组组成。一个简单的描述表如下：

1 C_MAINMENU 106

2 C_APPSTART 156

3 C_APPHELP 206

4 C_APPABOUT 256

……

105 C_APPEXIT 626

106 iBackground 0//UI 整屏重绘。

107 iImage 1//贴图

108 iPath RES//图片在资源文件中的索引

109 title R_I_BACKGROUND //该字段为资源文件中图片的索引值；

110 ix 0//要贴图片的左上角的 X 坐标

111 iy 0//要贴图片的左上角的 Y 坐标

112 iMenu 4//4 个菜单项

113 pText “开始游戏”//UNICODE 编码格式，菜单显示的名称

114 ix 50//菜单文字的显示位置的 X 坐标

115 iy 50//菜单文字的显示位置的 Y 坐标

……

124	iButton	2//2 个按钮
125	pText	“进入”//UNICODE 编码，按钮的显示名称
126	ix	10//按钮文字的显示位置的 X 坐标
127	iy	120//按钮文字的显示位置的 Y 坐标
128	iOnEvent	AVK_SOFT1//左软键，事件或是按键处理
129	iCondition	C_STATUS_MAINMENU//事件 ID 或按键值
130	iAction	C_STATUS_APPSTART//事件或按键响应描述

……

156	iBackground	//Next Status

……

其中，“1 C_MAINMENU 106”表示文件的开头，文件开头作为描述表的入口，本身也是一个小表，C-MAINMENU 等 C 开头的大写字符串为状态标志，标志每一个页面的状态描述。 数字 106 表示行号，意思是，我们首先找到要找的页面，比如是 MAINMENU， 这样我们就查到了 106 ，紧接着我们转到 106 来具体获取该页面的描述。

动态用户界面的实现流程具体如图 7 所示，其具体描述如下：

BREW 终端运行 BREW 应用程序，即执行步骤：程序启动 501；接着屏幕上会闪出一个页面，即执行步骤：启动闪屏 502；一般该页面仅仅是显示该应用程序的 LOGO 和开发商信息等。UI 脚本解释层 202 对 UI 脚本文件进行解释和分析，生成一个以 UI 状态为主键的描述表。

应用执行层 203 从 UI 描述表入口获取 UI 描述，比如背景，菜单项和按钮的描述，即执行步骤：转入 UI 描述表入口 503 和取得 UI 描述 504；接着绘制程序 UI，进入主程序菜单界面，即执行步骤：根据 UI 描述绘制页面 505。

接下来，执行步骤：判断是否有事件触发 206。

若没有按键或是事件被触发，则屏幕一直显示主程序菜单界面，处于空闲的状态，即执行步骤：空闲 507；

若有按键或是事件被触发，应用执行层 203 首先根据当前状态描述中对事件或按键的处理描述，调用相应的功能函数，并进行状态切换，即应用执行层 203 获取当前要跳转到的页面状态，也即是获取 iRunStatus 参数后，会根据该参数对存放在内存中的 UI 描述表进行查表，当查找到该参数对应的状态描述项后，就把当前状态修改为参数所对应的状态，并根据修改后的此当前状态的 UI 描述，重绘整个 UI，实现 UI 状态机的可靠切换，即执行步骤 508。

在执行完步骤 508 之后，跳转回到步骤 503，按流程继续循环上述的过程。

例如：当 BREW 应用程序启动，接着会出现闪屏，然后应用执行层 203 从 UI 描述表的表头查询到 C_STATUS_MAINMENU 的数据索引为 106，就把描述表指针指向 106 处，此时，应用执行层 203 获取主菜单界面的 UI 描述，根据 UI 描述表中对主菜单的背景属性、菜单属性、按钮属性对主菜单对 UI 界面进行绘制，即屏幕上显示主菜单界面。

当按钮事件发生时，比如 AVK_SOFT1 被按下，这时，应用执行层 203 通过当前描述表指针找到主菜单界面下对该事件的处理方法：跳转到 C_STATUS_APPSTART 页面；测试描述指针又跳转到表头处，搜索到 C_STATUS_ARRSTART 的索引值：156，则把描述指针指向描述表的索引值为 156 的地方，获取该状态页面的 UI 描述，这样就实现了 UI 状态机的切换。

采用本发明所描述的体系结构和实现方法，不仅能够很好的实现 BREW 应用程序的动态用户界面加载，而且还能够克服当前 BREW 应用软件开发中存在的几个弊端。

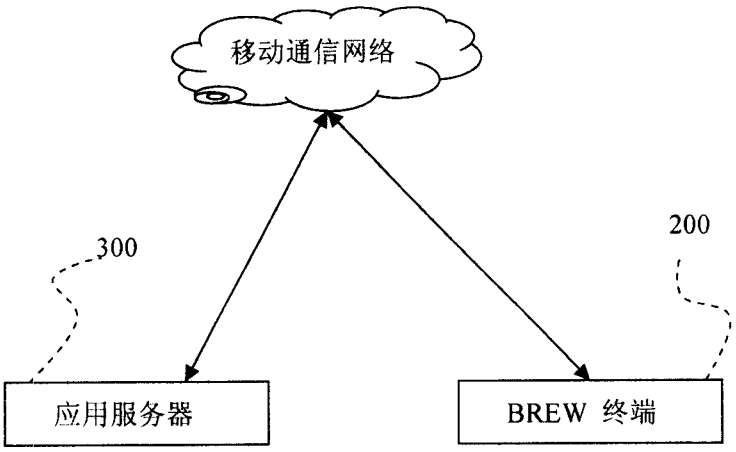

移动通信网络
300
200
应用服务器
BREW 终端

图 1

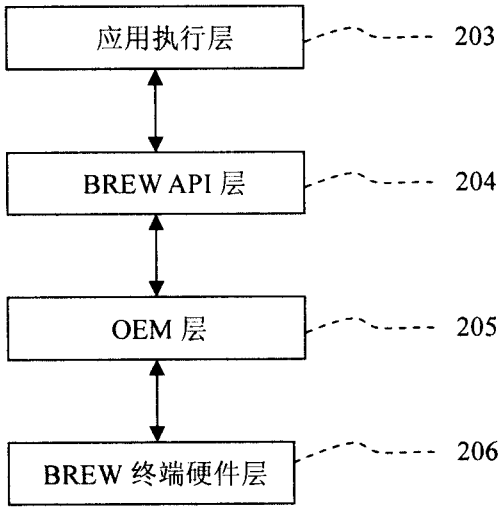

应用执行层
203
BREW API 层
204
OEM 层
205
BREW 终端硬件层
206

图 2

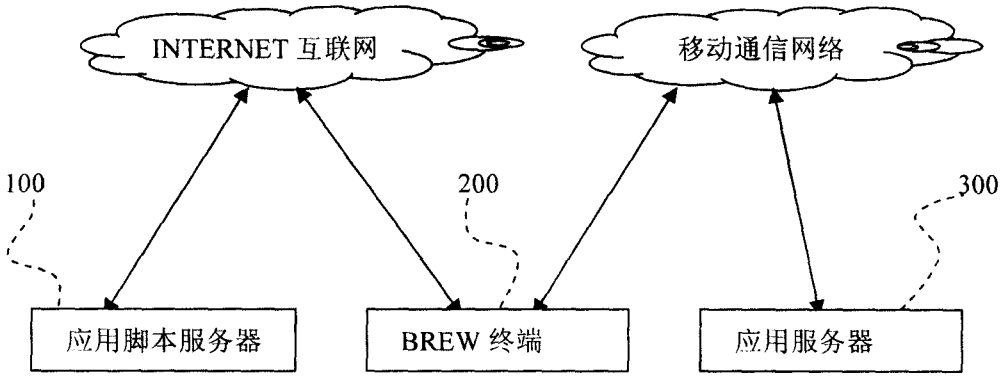

INTERNET 互联网
移动通信网络
100
200
300
应用脚本服务器
BREW 终端
应用服务器

图 3

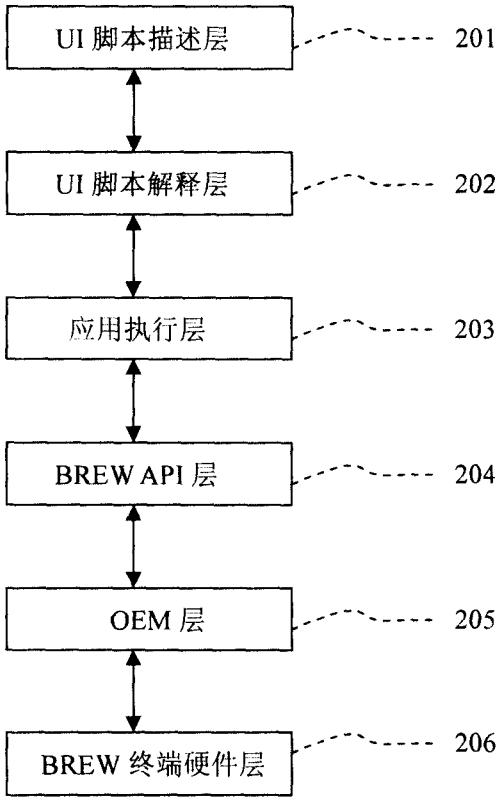

UI 脚本描述层
201
UI 脚本解释层
202
应用执行层
203
BREW API 层
204
OEM 层
205
BREW 终端硬件层
206

图 4

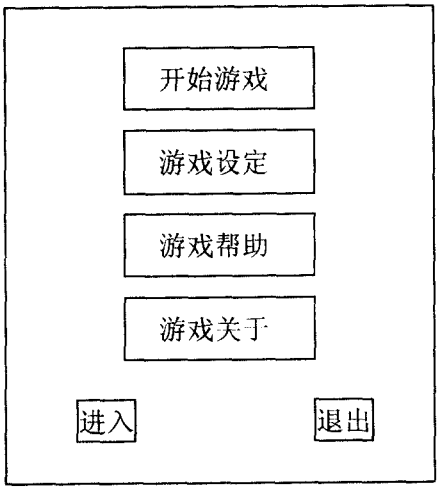

开始游戏
游戏设定
游戏帮助
游戏关于
进入
退出

图 5

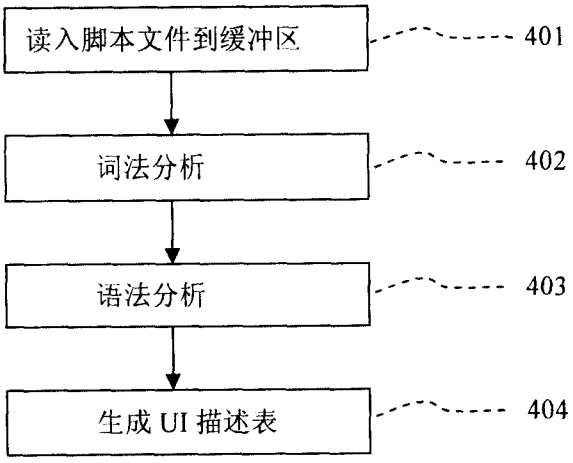

读入脚本文件到缓冲区
401
词法分析
402
语法分析
403
生成 UI 描述表
404

图 6

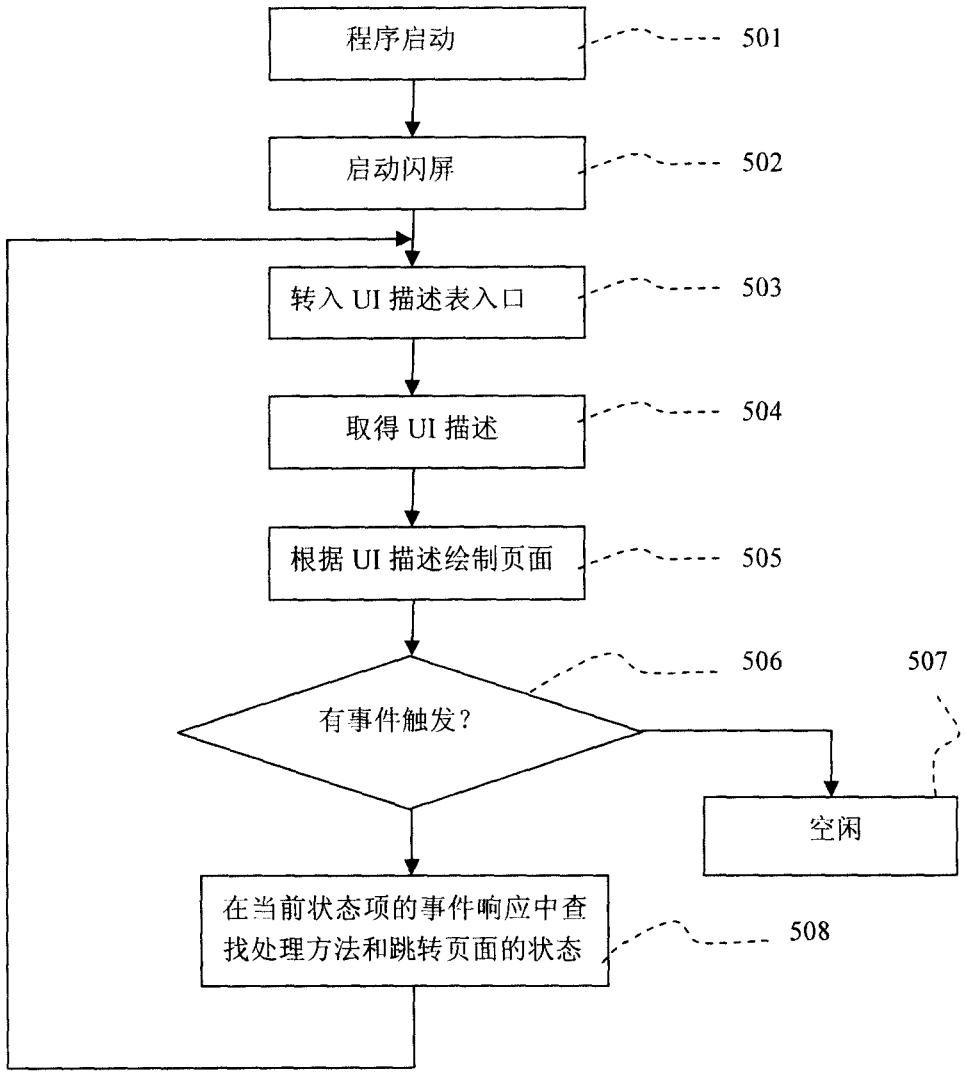

程序启动
501
启动闪屏
502
转入 UI 描述表入口
503
取得 UI 描述
504
根据 UI 描述绘制页面
505
有事件触发?
506
507
空闲
在当前状态项的事件响应中查找处理方法和跳转页面的状态
508

图 7