

# PowerBuilder 应用程序的编译发布技术研究

张益星, 罗 敬

(湖南工程学院 计算机科学与技术系, 湖南 湘潭 411101)

**摘 要:** PowerBuilder 发布的可执行应用系统一般包括三个部分: 可执行文件、动态库文件、外部资源文件, 编译发布 PowerBuilder 应用程序的目的是使发布后的应用程序能脱离开发环境独立运行。文中研究了基于 Windows 平台的 PowerBuilder 9.0 应用程序编译发布技术, 提出四种打包模型, 介绍了在 PowerBuilder 9.0 中创建工程的步骤、编译格式的选择、可执行文件的建立、资源文件的使用、动态库的选择。实践证明, 根据打包模型发布的应用系统满足了不同的应用要求。

**关键词:** PowerBuilder 9.0; 应用程序编译; 应用程序发布

中图分类号: TP311

文献标识码: A

文章编号: 1673-629X(2007)08-0043-05

## Research on Compiling and Deploying Technology of PowerBuilder Application

ZHANG Yi-xing, LUO Jing

(Dept. of Computer Science and Technology, Hunan Institute of Engineering, Xiangtan 411101, China)

**Abstract:** The PowerBuilder deploying application system includes three parts: an executable file, dynamic libraries and resources. The purpose of the compiled and deployed PowerBuilder application is to release from the development environment to run standalone. Researches into the compiling and deploying technologies of PowerBuilder application based on Windows, and explains in detail how to create project in step in PowerBuilder, choose a compiler format, create an executable file, use PowerBuilder resource files, and choose dynamic libraries. Practice shows that the application system based on the packaging model can meet various demands of application.

**Key words:** PowerBuilder 9.0; compiling of application; deploying of application

### 0 引言

PowerBuilder 是一个面向对象的图形化开发环境, 在这个开发环境中用户能够同时而且高效率地创建、预览和部署多个应用。当用户完成了应用的编程和调试工作之后, 最后一步就是应用的编译和发行, 这涉及到工程对象的创建, 生成可执行的应用程序, 以及发行环境的配置等多方面的问题。PowerBuilder 提供的编译工具可以将应用程序编译成可直接在系统下运行的应用程序, 文中是基于 Windows 平台上 PowerBuilder 9.0 的编译环境来研究 PowerBuilder 的编译发布技术。

### 1 创建工程

在 PowerBuilder 中, 用户可创建多种类型的应用。对于传统的 Client/Server 应用, 用户需要创建一个可执行的应用分发给最终用户。若用户创建的是分布式

应用, 则需要建立一个客户端可执行文件, 并为事务服务器建立一个服务器端可执行文件或服务器组件。此外, 用户还可建立一个代理对象、HTML 文件或 Java 类<sup>[1]</sup>。

用户在通过模板向导创建新应用时可同时创建一个工程, 创建工程的步骤如下:

(1) 执行菜单命令 File→New, 选择 Project 选项卡中的 Application Wizard 向导。

(2) 在 Specify Destination Library 对话框中选择存储工程的库文件, 单击 Next 按钮。

(3) 在 Specify Project Object 对话框中给工程命名, 单击 Next 按钮。

(4) 在 Specify Executable and Resource Files 对话框中给可执行文件命名, 选择资源文件, 单击 Next 按钮。

(5) 在 Specify Build Options 对话框中选择编译类型: Full Build(完全编译)和 Incremental Build(只重新编译新增部分)。Prompt for Overwrite 选中, 表示当覆盖文件时出现提示, 单击 Next 按钮。

(6) 在 Generate Machine Code 对话框中选择应用

收稿日期: 2006-10-31

基金项目: 湖南工程学院科研项目(120664)

作者简介: 张益星(1964-), 男, 湖南湘潭人, 讲师, 主要研究方向为分布式应用和系统集成。



程序是否生成机器代码,单击 Next 按钮。

(7)在 Specify Dynamic Library Options 对话框中选择确定应用程序是否生成动态库,单击 Next 按钮。

(8)在 Ready to Create Application 对话框中单击 Finish 按钮,显示如图 1 所示的工程画板界面,最后在工程画板中还可以对前面的设置进行修改。

(9)执行菜单命令 Design→Deploy Project,即可进行编译应用程序。

## 2 建立可执行应用系统

PowerBuilder 开发出的可执行应用系统一般包括以下三个部分:可执行文件(即 EXE 文件)、动态库文件(PBD 或 DLL 文件)、外部资源文件(如位图、图标等)。

### 2.1 编译格式的选择

PowerBuilder 对生成可执行文件提供了两种编译格式:伪码(Pcode)和机器码(Machine Code)<sup>[2]</sup>。

#### 1)伪码。

伪码是支持所有 PowerBuilder 平台的一种解释性语言,封装在伪码中的各种对象与其在库文件中具有相同的格式。伪码最大的优点是方便性和可移植性。

#### 2)机器码。

PowerBuilder 同样也可生成和编译成可执行代码或动态链接库。机器码关键的优点是具有很高的执行速度。

两种编码格式各有优点,可根据如下原则进行选择:

#### (1)速度。

如果你的首要目标是优化执行速度,同时应用也要求加强脚本的处理,则应选择机器码。若应用的代码大量地使用循环、浮点运算或函数调用,则使用机器码会比伪码执行得更快。伪码同样也有速度优势,即编译生成要比机器码快。这对于快速生成测试用途的可执行应用是非常方便快捷的。

#### (2)大小。

伪码生成的文件要比机器码生成的文件小。在应用所分发的机器中,若文件大小是主要问题,则应该放弃机器码的速度而选择伪码。

#### (3)可移植性。

伪码对于跨平台的应用是非常有用的,即可将同一 PBD 分发给使用 Windows 和 UNIX 平台的用户。但是,在不同的平台上,用户还需分别生成各自平台的可执行文件。

机器码只能用于指定的平台,这意味着用户必须为每个平台生成一套可执行文件和动态链接库。具体

选择何种代码,可根据运行的环境、应用的大小等多种情况综合考虑。

### 2.2 可执行文件

对任何 PowerBuilder 应用来说,都至少需要一个编译好的可执行文件(EXE)。这个文件包含了最基本的、能使应用系统在各种操作系统下执行的程序代码,用户可以通过双击 EXE 文件图标来执行应用程序。

### 2.3 资源文件

除了如窗口、菜单等 PowerBuilder 对象之外,应用在执行时还可能用到其他资源,如位图和图标等,这些资源可以作为可执行应用的一部分编译,也可以将它们作为单独的资源文件(PBR 文件)保存<sup>[3]</sup>。

#### 1)单独分发资源。

若一个资源不包括在可执行文件或动态链接库中,则应用在执行时将在搜索路径中搜索这些资源。在发行时,用户需将应用所需的资源分发到最终用户的搜索路径中。

这里所指的搜索路径包括:当前目录、Windows 目录、Windows 系统目录、所有由 PATH 指定的目录。

#### 2)利用资源文件。

另外,也可通过 PowerBuilder 的资源文件来发行资源,而不是单独发行。资源文件是应用所需资源的清单,在资源文件中列出了这些资源所在路径及文件名。PowerBuilder 建立可执行文件或动态链接库时可根据资源文件中列出的资源,将它们连入可执行文件或动态链接库中。这种方法适合建立一些大型的应用。

PowerBuilder 的资源文件是 ASCII 码的文本文件,其中列出了所需资源(位图、图标等文件)的文件名及其所在位置。创建和使用资源文件的步骤如下<sup>[4]</sup>:

(1)使用任意一种文本编辑器(如:记事本)新建一个文本文件,在此文件中列出所有资源名(如: bmp, cur, ico, rle, wmf 等文件),要求每项资源列为一行。若资源不在当前路径下,则应给出完整路径。

(2)在工程画板中,为可执行文件和动态链接库指定所需的资源文件名。

此时,用户就不需要单独发行资源了,因为资源已经在应用中了。如果诸如图片的资源被应用在执行时引用,则 PowerBuilder 将首先在可执行文件中搜索。若搜索失败,则在应用定义的资源文件中查找。若再次失败,则在文件的搜索路径中寻找。

### 2.4 动态库文件

若应用系统使用多个库文件,在编译时,可将除应用对象之外的库文件编译成动态链接库,应用系统在运行时会动态调用不包含在可执行文件中的其他对



象。这使用户可将应用系统编译成便于管理的几个小单元,同时也可减小可执行文件的大小<sup>[1]</sup>。

使用动态库有以下优点:

- (1)模块化:利用动态库技术可将应用分解为几个模块,易于管理。
- (2)易维护:将整个应用分解为几个组件向用户分发,当版本需要升级时,不必重新分发全部的应用,只分发有关的动态库即可。
- (3)可重用:动态库可由几个应用共享。
- (4)灵活性:动态对象只在执行时动态调用。
- (5)高效性:由于对象是在执行期间动态调用,对于一个大型的应用来说,可以提高内存的利用效率。

因此,在组织应用系统时,应当充分利用动态库技术,把在整个应用中利用率较低的对象,或可能被其他应用调用的组件作为动态对象放在动态库中。在可执行文件中只放少量的、使用频率最高的对象。

2.5 选择打包方式

完成前面的过程之后,所面对的是选择编码格式,将哪些内容打包发行,以及选择何种打包模型。这些问题关系到发行的成功与否,是编译发行应用的关键环节。

2.5.1 打包文件

无论用户选择哪种编码格式,在 PowerBuilder 中生成的应用都将包括可执行文件、动态链接库和资源文件。

- (1)可执行文件。若只是创建两层应用并分发给最终用户,则应该生成可执行文件而不是一个服务器组件;若创建一个分布式应用,则应该分别生成客户端和服务器的可执行文件。一个可执行文件最少应包括能使应用正常运行的代码。
- (2)动态链接库。为了不使全部应用打包成一个大的可执行文件,用户可将应用中的一些对象打包成一个大的或多个动态链接库文件。所生成的动态链接库文件要根据所选的编码格式决定,如表 1 所示<sup>[5]</sup>。

表 1 编码格式

编码格式	生成的动态库文件
机器码	在 Windows 下生成扩展名为 DLL(动态链接库)的文件。这些动态库文件与其他标准的共享库文件类似,惟一不同的是这些动态库不能被外部程序调用
伪码	生成 PBD(PowerBuilder 动态库)文件,这些动态库文件与应用执行时链接的 DLL 相类似。但并非能与 DLL 互换,因为它们内部有不同的格式。用户在发行同一应用时不能同时使用两种动态库(DLL 和 PBD)文件

动态库文件不包括任何开始代码,它们不能独立执行,而只能在应用执行需要某对象时,才会在动态库中查找(假如应用不能在可执行文件中找到);而应用的代码只能编译成可执行文件,并通过双击图标启动。

(3)资源文件。当使用资源时,必须将这些资源作为应用的一部分发行,一个 PowerBuilder 应用可使用多种类型的资源,并且不同的对象使用的资源也不同,如表 2 所示。

表 2 资源类型

对象名	可使用的资源
窗口和用户对象	图标(ICO 文件) 图片(BMP,GIF,JPEG,RLE 和 WMF 文件) 光标(CUR 文件)
数据窗口	图片(BMP,GIF,JPEG,RLE 和 WMF 文件)
菜单(在 MDI 窗口中)	图片(BMP,GIF,JPEG,RLE 和 WMF 文件)

2.5.2 打包模型

从前面的设置中已经知道可将应用编译成多种类型(可执行文件、可执行文件和资源文件、可执行文件和动态库等),为了正确选择打包的模型,下面给出几种常用的打包模型,可根据实际情况选择<sup>[5]</sup>。

(1)一个单独的可执行文件。

在此模型中,用户将所有对象和资源都包括在可执行文件中。这意味着,发行时可只发行一个文件,这种模型适用于简单小巧的应用,是最简单的一种模型,此模型的示意图如图 1 所示。

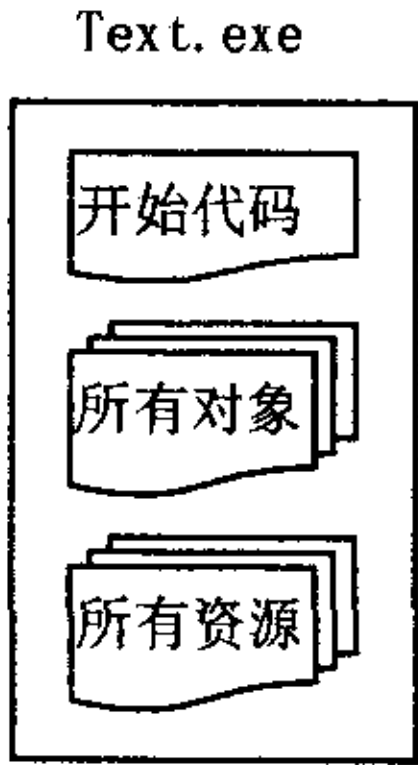


图 1 单独可执行文件模型

(2)一个可执行文件和资源文件。

在此模型中,所有对象和大多数资源都包括在可执行文件中,发行时除要发行可执行文件外,还须为某些特定的资源发行一些文件,这种模型同样适用于简单小巧的应用。但它不同于前一种模型的是,它对于维护经常改变的资源是非常方便的。也就是说,可只对软件修订所需的资源发行一些文件,而不必再次发行可执行文件。另外,用户还可使用此模型来处理某些大型且不经常使用的资源,或必须被其他应用共享的资源。此模型的示意图如图 2 所示。

(3)一个可执行文件和动态库文件。

在此模型中,应用被分成可执行应用和一个或多个动态库文件(DLL 和 PBD)。当使用此模型时,用户可以选择不同的方法来组织对象和资源,如表 3 所示。

这种模型,适用于大多数应用,因为它赋予用户在组织、发行、维护访问更大的灵活性。例如,它允许用户修正应用系统的个别部分(打包在动态库中的部分)



并单独发行,而不必再次分发应用的其他部分。如图 3 所示。

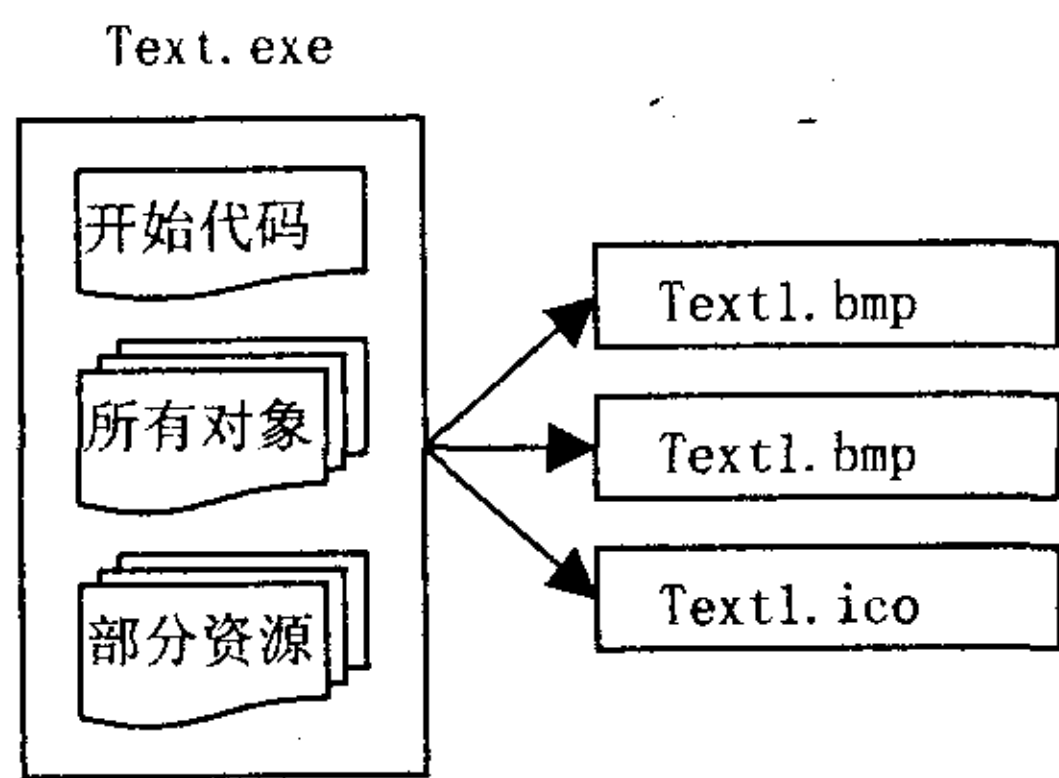


图 2 可执行文件和资源文件模型

表 3 组织对象和资源的方法

对象类型	组织方法
对象	将全部对象打包进动态库,或者将少量经常访问的对象打包进可执行文件,而将其余的对象打包进动态库
资源	将大多数或全部资源打包进动态库,或者将大多数或全部资源打包进可执行文件

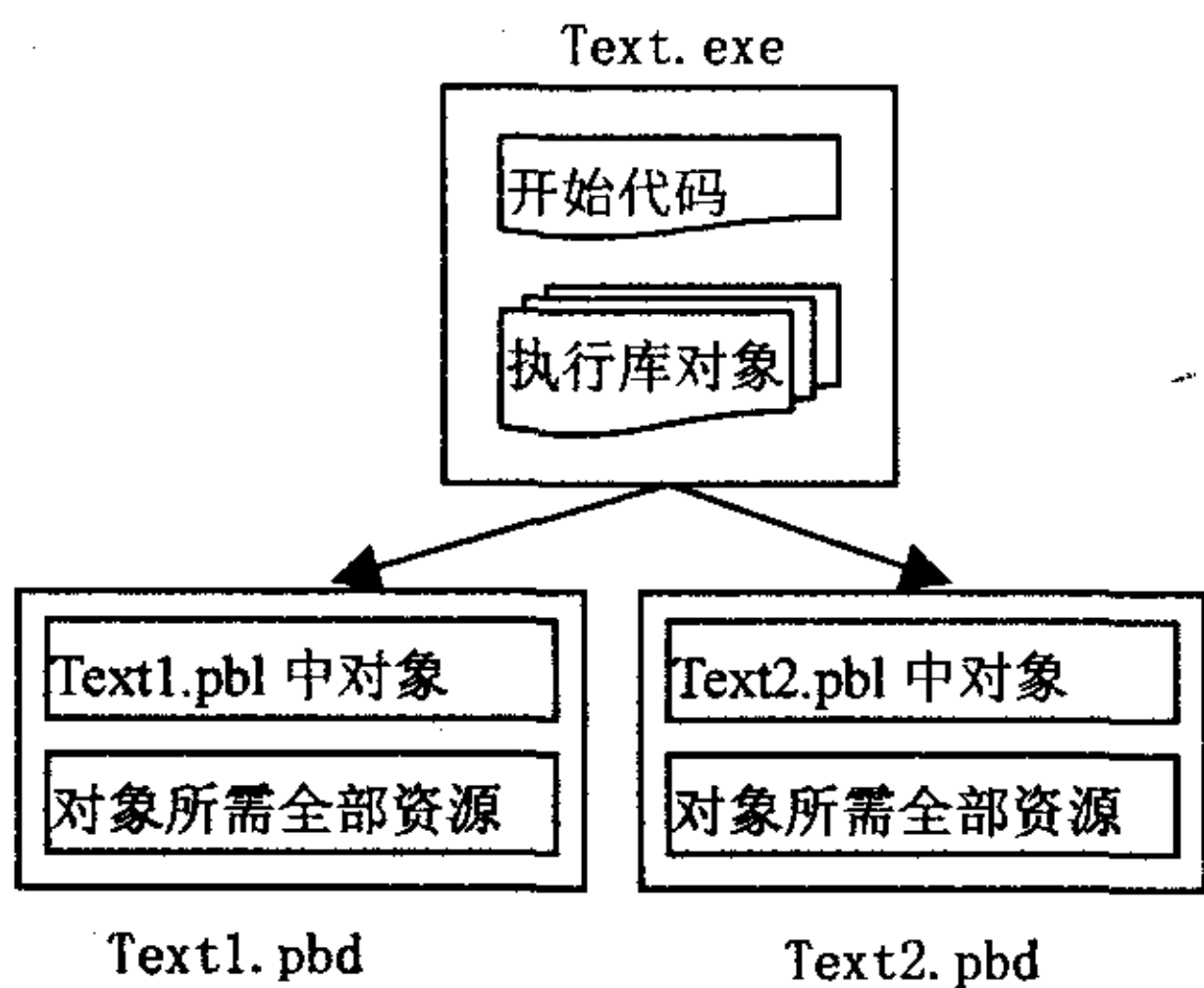


图 3 可执行文件和动态库文件模型

(4)一个可执行文件、动态库文件和资源文件。

此模型除发行包括在可执行文件、动态库文件中的资源外,还需单独为某些特定的资源发行一些文件,如图 4 所示。

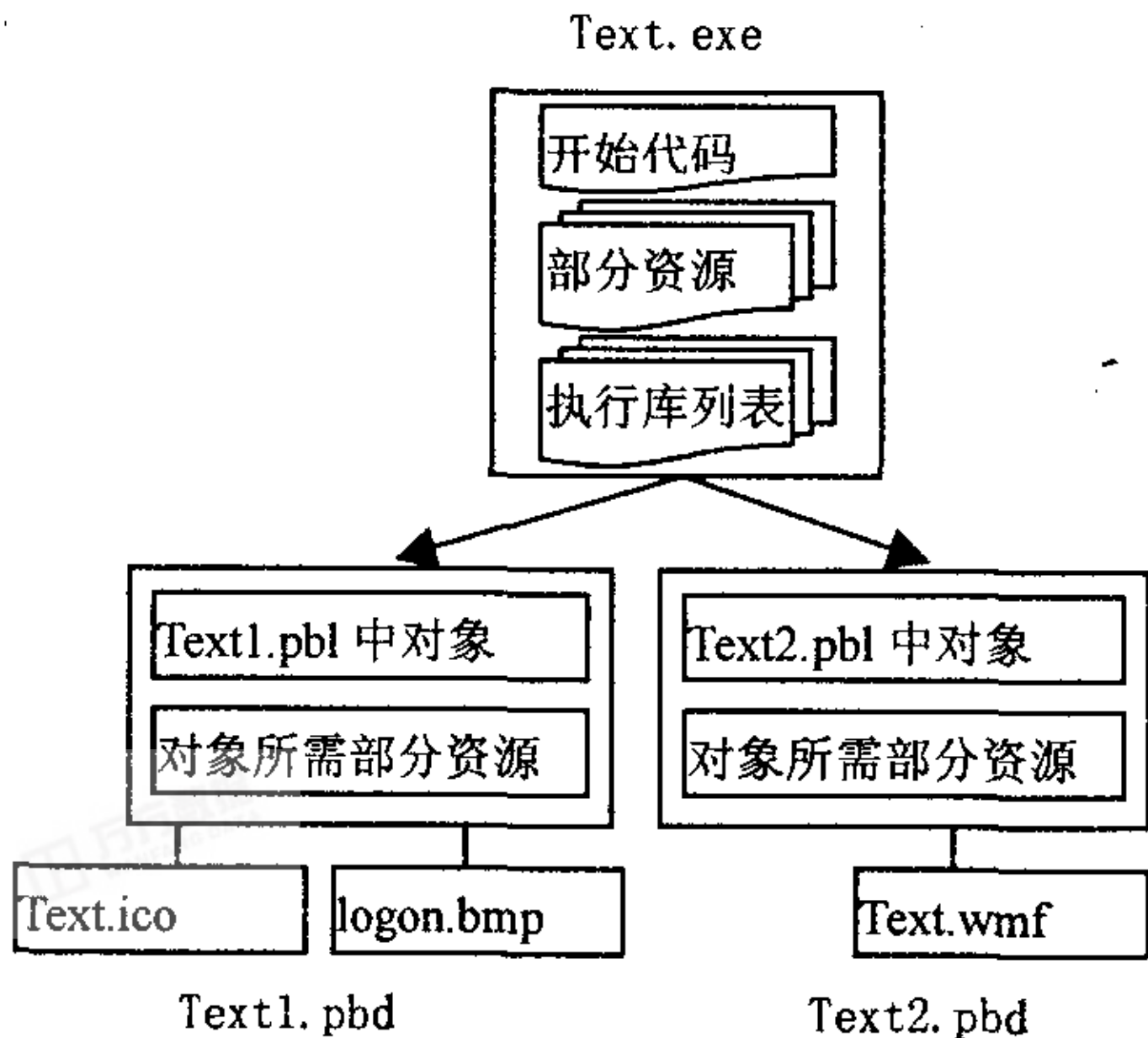


图 4 可执行文件、动态库文件和资源文件模型

此模型适用于要求灵活处理各类资源的应用系统、可利用此模型来处理部分资源被修正、与其他应用共享资源或文件很大而且不经常使用的情况。

2.6 编译生成和测试

进行以上各项选择之后,可在工程画板中选择编

码方式,并将资源文件与可执行文件或动态链接库连接起来,最后选择编译生成所要发行的可执行应用。若编译通过,还必须在发行前对生成的可执行应用进行测试。测试的步骤如下:

(1)离开 PowerBuilder 的开发环境,返回到操作系统环境下。

(2)确保生成的可执行应用能够访问 PowerBuilder 的发行库文件。一种最简单测试应用可执行与否的方法是将编译好的文件(包括可执行文件、动态库,以及数据库文件等)拷贝到 Sybase/Shared/PowerBuilder 下。

(3)运行应用的可执行文件。

通过以上测试的可执行应用,应该说已经可以发行了。但真正要使这个可执行应用能在各种用户的机器上运行,还需发行一些 PowerBuilder 的环境文件。

3 应用程序的发行

发行 PowerBuilder 应用程序时,只在用户的机器上安装应用程序的可执行文件(包括动态库文件和资源文件)是远远不够的。为保证正常运行,还应向用户发行脱离 PowerBuilder 开发环境所必需的环境文件,包括 PowerBuilder 运行时库、安装数据库接口,配置 ODBC 数据源,安装网络驱动程序,以及必要时修改操作系统的配置等。

3.1 PowerBuilder 运行时库

PowerBuilder 开发环境中包括一组脱离开发环境运行 PowerBuilder 应用程序所需的运行时库,称为 PowerBuilder Development Kit,这是一组动态链接库(DLL)。发行应用程序时,必须同时向用户提供这组运行时库,并安装在合适的路径下(通常应放置在应用程序所在的目录下或系统的搜索路径下)<sup>[1]</sup>。

不同的 PowerBuilder 版本使用的动态链接库不同。用 PowerBuilder 9.0 开发的应用程序,一起发行的基本动态链接库文件有:PBVM90.DLL, PBDWE90.DLL, PBODB90.DLL, LIBJCC.DLL, LIBJSYBHEAP.DLL 等。PBVM90.DLL 是 PowerBuilder 9.0 的虚拟机,任何由 PowerBuilder 开发的程序在发布时都必须携带虚拟机,否则将不能运行指定的应用程序;PBDWE90.DLL 在使用了数据窗口的应用程序中是必需的;PBODB90.DLL 是 PowerBuilder 与 ODBC 交互的接口文件。当应用程序使用了管道、超文本控件等对象时还需发行 PBRTC90.DLL 文件。

虽然上面介绍了运行 PowerBuilder 应用程序所必需的几个文件,但当有特殊需要时,很难确切地知道需要哪些 PowerBuilder 运行时库。PowerBuilder 9.0 新增



了一个运行时库打包工具 Runtime Packager,利用此工具可以很方便地把 PowerBuilder 运行时需要的文件打包,然后和应用程序一起发行。

3.2 安装数据库接口

当应用程序需要访问数据库时,在为用户安装应用程序的同时还必须为其安装好数据库接口文件。安装数据库接口文件包括两方面的内容:第一,安装 PowerBuilder 提供的专用数据库接口或 ODBC 驱动程序(根据应用程序要访问的数据库而定);第二,安装数据库厂商提供的数据库驱动程序。表 4 列出了访问大型数据库所需的 PowerBuilder 专用接口文件,这些文件应该安装在应用程序所在的目录或系统的搜索路径中。

表 4 Windows 下各数据库使用的专用接口

数据库管理系统	PowerBuilder 专用接口文件
Informix 9	Pbin990.dll
Oracle 8. X	Pbo8490.dll
Powersoft ODBC 接口	Pbodb90.dll,Pbodb90.ini
SQL Server 2000	Pbmss90.dll
Adaptive Server Enterprise(11. X,12. X)	Pbsyc90.dll

3.3 配置 ODBC 数据源

如果应用程序使用了 ODBC 数据源,在为用户安装应用程序的同时还必须为其安装和配置 PowerBuilder ODBC 驱动程序 Pbodb90.dll 和 Pbodb90.ini,这两个文件应该安装在应用程序所在的目录或系统的搜

索路径中。另外,还需要修改 ODBC 初始化文件 odbinst.ini 和 odbc.ini,这两个文件通常在 Windows 目录下,如果用户机器上没有这两个文件,可从开发环境中复制<sup>[1]</sup>。

4 结束语

文中就基于 Windows 平台的 PowerBuilder 9.0 应用程序编译发布的关键技术进行了分析,并给出了四种编译成可执行文件的打包模型。详细介绍了在 PowerBuilder 9.0 中创建工程的步骤、编译格式的选择、可执行文件的建立、资源文件的使用、动态库的优点、应用程序的发行环境等,为脱离开发环境运行 PowerBuilder 应用程序打下了坚实的基础。

参考文献:

[1] 樊金生,张翠肖,沙金,等. PowerBuilder9.0 实用教程[M]. 北京:科学出版社,2004.

[2] 柯建勋,张涛. PowerBuilder8.0 进阶篇[M]. 北京:清华大学出版社,2002.

[3] 杨昭. PowerBuilder9.0 对象与控件技术详解[M]. 北京:中国水利水电出版社,2003.

[4] 张遂芹. PowerBuilder9.0 系统开发实例[M]. 北京:中国水利水电出版社,2003.

[5] 吕睿烜,李勇,温为民. PowerBuilder9.0 开发精解[M]. 北京:电子工业出版社,2003.

(上接第 42 页)

息,然后找到该用户所在的聚类,并把该类用户所具备的共有属性(类用户的 Web 关联规则、用户兴趣)推荐出来。

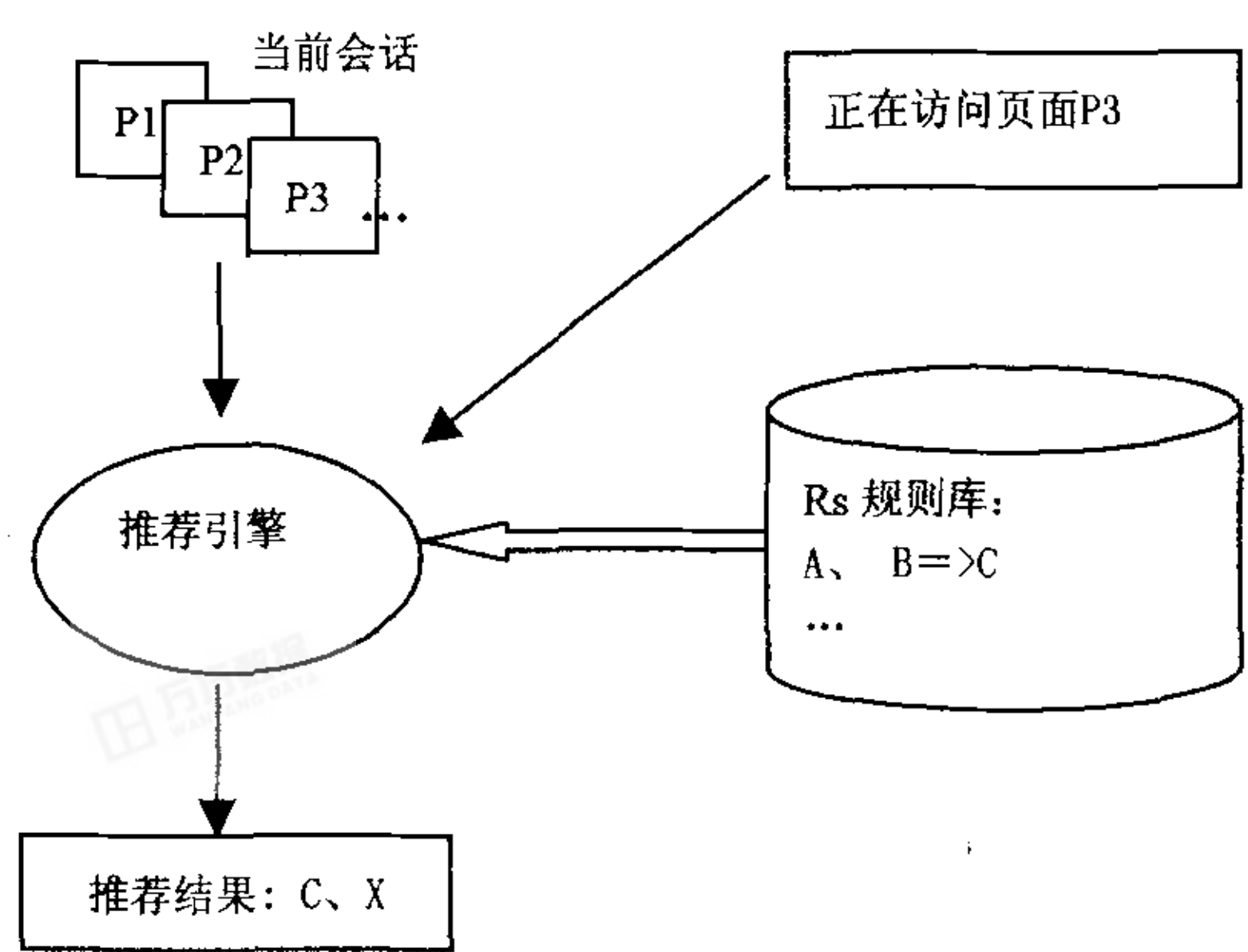


图 2 关联引擎推荐图

4 结束语

Web 关联规则作为智能站点知识库的重要组成部分,对实现用户使用的个性化具有重要的意义,文中综合考虑了关联规则挖掘过程中的用户的实际访问序

列,消除反向关联规则对系统知识库的影响,提高了系统的效率和准确性。

参考文献:

[1] Han Jiawei, Kamber M. Data Mining Concepts and Techniques[M]. 北京:机械工业出版社,2001.

[2] Pal S K, Talwar V, Mitra P. Web Mining in Soft Computing Framework: Relevance, State of the Art and Future Directions[J]. IEEE Transactions on Neural Networks, 2002, 13 (5):1163-1169.

[3] 史忠植. 知识发现[M]. 北京:清华大学出版社,2002.

[4] Lu Jieping, Liu Yuebo, Ni Weiwei, et al. A fast interactive sequential pattern mining algorithm[J]. Wuhan University Journal of Natural Science, 2005, 11(6):3136-3139.

[5] 冯玉才,冯剑林. 关联规则的增量更新算法[J]. 软件学报, 1998, 8(4):301-306.

[6] 汤亚玲,崔志明. 基于遗传算法的 Web 行为挖掘研究[J]. 微电子学与计算机, 2006, 23(8):168-170.

[7] 金阳,左万利. 有序概念格与 WWW 用户访问模式的增量挖掘[J]. 计算机研究与发展, 2003, 40(5):635-683.

[8] 朱玉全,孙志挥,赵传申. 快速更新频繁项集[J]. 计算机研究与发展, 2003, 40(1):94-99.