

Methods for Realization of Software Configuration Information

软件配置信息设计 实现方法

张萍 易东云 (湖南长沙国防科大理学院数学与系统科学系 410073)

摘要: 通过软件的配置信息, 实现软件的不同功能和部分数据的获取, 可以提高其通用性、灵活性, 满足不同用户的各种需求, 是软件设计实现的一个重要方法。本文总结和介绍了软件配置信息设计实现的几种方法, 结合实际开发工作中的经验, 分析了各自的特点和适用性, 对解决软件开发中的此类问题有一定帮助作用。

关键词: 软件 配置信息 文件 注册表 数据库

1 前言

软件的配置信息对于一个完整的软件来说, 是必不可少的, 它有助于实现软件各种功能的组合与控制, 丰富用户使用软件的交互性手段, 增强了软件的通用性、灵活性、可维护性和可操作性, 是一个影响软件生命力的重要因素。

配置信息现在不再是单纯的软件操作的对象, 它已经成为软件的功能、界面, 甚至是代码的一部分扩展融合。例如, ActiveX 控件可以嵌入用户文档, 具有执行代码和界面的功能, 也可以作为配置信息的载体。同时, 它也不再是某个软件的问题, 随着 ActiveX、COM、DCOM、网络技术、互联网的发展, 使得配置信息扩展为众多软件之间协同工作的纽带和桥梁, 通过具有开放性、标准化的配置信息实现全球的网络协同计算已成为现实。由于商用软件配置信息具有的开放性、标准化和国际化等特点, 对软件开发和技术发展具有一定的推动作用, 数据库在这方面的贡献尤为突出。

配置信息实现的主要方法有 Windows 系统提供的初始化文件、注册表、数据库等几种方式, 也可以由开发者制定适合具体软件环境的数

据格式来记录和操作配置信息。使用模式随着软硬件技术性能的发展, 由最初的固态信息(缺乏配置), 发展为本机外部数据配置, 进一步扩展为网络模式。以下对这些方法和模式进行简要的介绍和分析。

2 初始化文件

初始化文件即 ini 文件; 在 Windows 系统中可以看到大量的 ini 文件, 保存着各种软件的配置信息, 系统本身一些配置信息就保存在 Win.ini 和 System.ini 文件中, 这是一种最常用的配置信息设计方法, 从 Win3.1 直到最新的 Win2000、XP 都支持。标准 Win32 函数和 Visual C++ 的 MFC 类库中对 ini 文件操作提供了丰富的函数, 在程序设计和实现中容易掌握。

Ini 文件本身是文本文件, 具有标准的格式, 所有软件都是统一的, ini 文件的格式如下:

```
[section1]
    key1=int
    key2=string
[section2]
    key1=int
```

key2=string

方括号中的 section 为字段名, key1 为键名, 即变量名, 等号后依据键所取的值为整型或字符串, 在一行的开始为 `;` 则是注释语句。

Win32 函数提供了访问系统 Win.ini 和用户自己的 ini 文件信息的两套函数, 函数名前缀统一是 Get/Write(读/写)Profile/PrivateProfile(系统/私有), 指定 ini 文件名、段名、键名, 就可以获取和设置相应的信息。

例如, GetProfileInt 函数获取 Win.ini 文件中指定的段和键相应的整型数, GetPrivateProfileInt 获取在用户自己某个 ini 文件中相应的整型数, WriteProfileInt 设置在 Win.ini 中相应的整型数。

此外 Win32 函数还提供了字符串的操作和段名、键名的管理检索等函数。

MFC 通过 CWinApp 类实现了支持与执行程序同名的私有 ini 文件操作, 所以其函数没有 Private 前缀, 不用指定文件名, 但提供的 Get/Write ProfileBinary 函数实现了 Win32 函数不支持的数据串操作(在 VC 资料中没有给出, 实际可用)。

对于配置信息较少的软件,采用ini文件是很方便的,使用函数简便,不需要复杂的配置管理,使用文本编辑器即可进行编辑。它的缺点是对于信息量比较多的软件不便使用,数据只有一个层次结构,不能支持稍复杂的数据格式;由于可以随意编辑,无法保证不被篡改,安全性不高。

查看 Visual C++ 的开发资料,可以看到并不推荐使用 ini 文件,而建议将信息放入注册表,实际上 MFC 类库 CWinApp 类的 GetProfileInt 等 ini 操作函数就是这么实现的,在程序开始处调用了 SetRegistryKey 函数之后,使用 MFC 的 ini 函数,信息实际保存在了系统当前用户的注册表键 HKEY_CURRENT_USER 中,这样可以实现系统每个用户使用同一个软件的不同配置,但如果不调用 SetRegistryKey 函数, MFC 才会将信息保存在 Windows 系统目录下的和执行程序同名的 ini 文件中,这样可以实现每个用户的相同配置,因此,使用时要根据具体需要来选择 MFC 的程序设计功能。需要注意的是, MFC 将工具条、最近使用的文件列表等内容也记录在你的信息存放位置(注册表或 ini 文件)中。虽然 ini 文件不推荐使用,但其简便灵活的特点在程序设计实现上还是具有一定优势。

3 注册表

Windows 系统的注册表中包含了有关计算机的各种信息,如软件的配置信息、系统配置、专用用户设备信息和环境配置信息等。注册表是针对所有 32 位硬件/驱动程序以及 32 位软件设计的数据文件,主要用于帮助系统对硬件、软件、用户环境以及 Windows 界面进行控制。注册表是由 System.ini 和 Win.ini 以及各应用程序的 ini 文件演变而来,它突破了使用文件的大小限制,利于系统的统一管理、扩展和各类软硬件

的升级。注册表也是记录相对较少的初始化和配置数据的很好方法,在 Win32 函数中提供了丰富的注册表操作函数。

注册表是以层次组织存储的信息,这个树状信息结构的每个条目称为键,一个键可以包含任意数目、层次的子键,也可以包含称为“值”的数据条目,以这种形式,注册表存储了各种信息,注册表主要按以下五个键组织: HKEY_LOCAL_MACHINE、HKEY_CLASSES_ROOT、HKEY_CURRENT_CONFIG、HKEY_USERS、HKEY_CURRENT_USER,

HKEY_LOCAL_MACHINE 和 HKEY_USERS 对应了所有用户,而 HKEY_CURRENT_CONFIG 和 HKEY_CURRENT_USER 对应当前用户,因此软件如果在用户限制上有要求,就要注意使用不同的键。

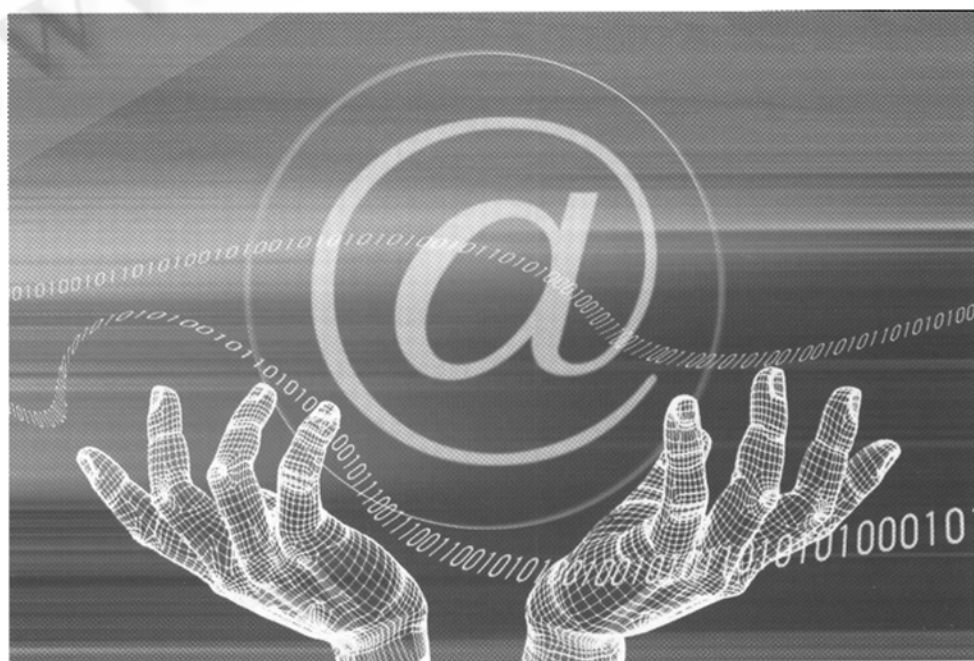
Win32 注册表函数基本都以 Reg 为前缀,由于结构层次复杂,因此函数参数较多,使用略有些不便。例如: RegOpenKey: 打开一个键或子键; RegEnumValue: 获取注册表的值信息; 除此之外,还提供了其他创建、删除、查询、检索等函数,可以完成所有注册表操作功能。

借助操作系统的管理,注册表具有一定安

全性机制,理论上具有无限大空间,开发函数功能齐全,强大,多级层次结构相对于 ini 文件的一个层次具有优势,便于实现较为复杂的配置信息管理。NT 系统注册表还在 HKEY_PERFORMANCE_DATA 键中记录了实时性能数据信息,可以实现对系统的监视。缺点是开发函数相对于 INI 文件有些复杂,由于注册表涉及操作系统信息,一旦被破坏,有可能导致系统崩溃,需要认真操作。另外庞大的注册表对系统性能也有一定影响,尽管注册表本身对应用程序能存储的数据类型和大小几乎没有技术上的限制,但对系统效率、开销上产生巨大的注册表使用应有一些限制。设计者可以将部分配置和初始化数据存储注册表中,其他大量信息以文件或数据库的方式存储在其他地方。

注册表信息实质上还是基于文件的,它保存在系统目录中的几个文件中,操作系统也提供了编辑的工具,还有一种 Reg 文件,类似 ini 文件格式,可以直接输入到注册表中,这种机制有一定灵活性,但也是不安全因素。

比较上面两种方法,都有一定的局限性,在信息量大的情况下不宜使用。单一的树型结构难以描述自然界客观存在的复杂关系模



型, 因此在许多开发项目中都使用了数据库方法。

4 数据库

数据库技术在目前互联网各项服务中得到了广泛应用, 技术成熟度值得信赖, 其适应大规模数据的能力非常强大。除此之外, 从小型的DBase、FoxPro到大型的Oracle、SyBase等, 各种数据库开发平台为我们提供了丰富的选择空间, 并且具有良好的通用性和兼容性, ODBC开放数据库接口, 标准SQL数据库语句, 使同样的程序代码可以在不同类型的数据库上执行, 能够适应各种平台, 具有很强的移植性。随着其广泛应用, 数据库编程已经成为程序设计中发展迅猛的一项内容。

和软件的设计一样, 数据库的设计是一个重要工作, 有一套完整的理论, 数据库管理系统(DBMS)只是一个设计、使用的平台, 根据实际需要和数据结构, 对数据库模式进行定义。好的设计对于可维护性和操作性具有重要意义。数据录入相对只是一个很小的工作。传统的ODBC编程是利用高级语言(如C语言)用ODBC的API来实现, 但过程比较复杂, 各种参数不易理解, 且直接获取返回的数据较困难。VC的MFC类库对ODBC的API进行了封装, 进行了部分简化。除此之外C++ Builder也是一个非常好的开发工具。

另外数据库一般也提供其相应的嵌入式开发系统, 例如Oracle公司的PRO系列, 它是该公司提供的在高级程序设计语言中嵌入SQL语句来访问数据库的一套预编译程序, 目前可以支持C/C++、Ada等多种高级语言, 在VC等开发工具中使用。一般执行效率要比ODBC高一些, 但无法移植到异构数据库上。

目前在数据库开发方法上有许多新技术得到了广泛应用, 例如Oledb、ADO(ActiveX Data Objects)、DAO(Data Access Objects)等, 在此就

不详细介绍了。

数据库的配置管理由于有DBMS的存在, 使工作简化了许多, 象MS Access、FoxPro和SQL等根据选择的数据库模式使用相应的管理系统; 也可以通过程序设计实现用户特殊要求的配置管理。

数据库开放式的接口, 是软件开发的一个优势, 在数据库不断发展的过程中, 和分布式、客户机/服务器以及专家系统、人工智能等技术不断结合, 在安全性、可移植性等方面有了提高, 成为开发的标准, 良好的开发支持, 完善的设计平台, 是软件设计开发的成功保证。缺点是程序运行时至少还不能脱离数据库的驱动程序, 还不能完全满足实时性能的要求, 所以只能用于程序开始和结束阶段。

5 自定义数据文件

目前流行的软件换肤(Skin)技术(通过更改配置信息改变软件的界面)是自定义数据文件作为配置信息的一个典型代表。由于文件格式由设计者自行定义, 因此也要开发相应的配置管理工具, 在这方面的工作量比较大。

主要设计方法和原则:

(1) 为了解决软件版本升级和维护扩展等带来的一系列问题, 最好在自定义数据文件的开头保留有文件头, 记录软件的版本信息;

(2) 确定程序开发中读写自定义数据文件的程序设计方法, 主要是通过文件操作来进行;

(3) 要便于实现信息的检索和查找;

(4) 考虑文件格式和程序运行中保持在内存中的配置信息的一致和统一;

(5) 为保证可靠性和安全性, 可以通过加密、压缩等技术来实现。

6 客户机/服务器模式和互联网环境

除了配置信息的设计实现方法外, 其使用

模式对于软件的开发也有一定影响, 最简单的就是本地模式, 在本机上只使用本机上的配置信息。

随着客户机/服务器模式的兴起, 各种信息又划分为客户端和服务端, 分别为不同的功能进行配置和管理。使用模式进行了延伸扩展, 作为服务的一方, 要收集客户的配置信息, 提供相应服务功能。软件开发也要有所区别, 这给软件行业的发展带来了变化。

典型的代表就是软件许可证和序列号等信息的实现方法, 需要到互联网的另一端去查询。互联网的快速发展, 产生了许多新的模式, 需要不断探索和实践, 也将带动着软件行业再次产生巨大变化。

7 结束语

配置信息的多样性带来了软件功能的完善, 本文总结了这些设计实现方法, 有助于在进行软件开发时提供一个可供参考的技术帮助。由于本人实际工作经验和程序设计的水平不高, 在技术理论角度等方面考虑还不周全, 希望大家多批评指正。 ■

参考文献

- 1 (美)Microsoft公司, 《MSDN Library Visual Studio 6.0版》, 1998。
- 2 (美)Microsoft公司, 《Win32函数手册》, 清华大学出版社, 1996。
- 3 张海藩, 《软件工程导论》, 清华大学出版社, 1992。
- 4 清汉计算机工作室, 《Visual C++ 6.0数据库与网络开发实例》, 机械工业出版社。