第16卷 第12期
2006年12月

计算机技术与发展
COMPUTER TECHNOLOGY AND DEVELOPMENT

Vol.16 No.12
Dec. 2006

基于 Visual Prolog 6.3 内核的专家系统实现

申晓勇,雷英杰,王 坚
(空军工程大学 导弹学院,陕西 三原 713800)

摘 要:针对目前构建专家系统开发环境的局限性造成开发周期长的问题,提出了利用传统语言调用逻辑程序设计语言的动态链接库构建专家系统的方法,通过两类语言优势互补,实现高效开发专家系统。详细论述了 Visual Prolog 6.3 动态链接库的生成过程、调用方式、调用约定以及编码转换,并介绍了利用 VB.net 如何调用 Visual Prolog 6.3 生成的动态链接库构建专家系统。通过实例验证了该方法的可行性,并具有良好的应用前景。

关键词:专家系统;Visual Prolog;动态链接库;谓词;事实数据库
中图分类号:TP182 **文献标识码**:A **文章编号**:1673-629X(2006)12-0091-03

Implementation of Expert System Based on Visual Prolog 6.3 Kernel

SHEN Xiao-yong, LEI Ying-jie, WANG Jian
(Missile Institute of Air Force Engineering University, Sanyuan 713800, China)

Abstract: To solve the limitation of expert system development environment which results in the longtime exploitation, an approach is put forward to build an expert system by transferring logical programming language DLL through common programming language, which make use of the reciprocity of the two kinds of language to reduce the cycle of exploitation. Discusses the creating process, transfer mode, transfer promise and code conversion of Visual Prolog 6.3 DLL, and introduces how to transfer Visual Prolog 6.3 DLL through VB.net to build an expert system. Proves the feasibility and well foreground of this method through this instance.

Key words: expert system; Visual Prolog; DLL; predicate; fact database

0 引 言

专家系统的三个要素为:知识库、推理机与用户界面。目前专家系统的开发周期都很长,其中问题的关键就是开发环境的局限性。用传统设计语言开发,对界面设计、知识数据库维护方便,而事实、规则的表示以及推理机的设计则非常困难,其工作量和复杂程度之高造成开发周期变长;若用逻辑程序设计语言开发,则可免去推理机的开发工作,使得程序量大大减少,但是其界面开发、数学计算以及知识数据库管理是此类语言的致命缺陷,经常不得不使开发人员忍痛割爱。

Visual Prolog 语言是当今新一代开发智能化应用的强有力工具,是人工智能与专家系统领域最著名的逻辑程序设计语言[1]。它具有模式匹配、递归、回溯、对象机制、事实数据库和谓词数据库等强大功能,支持系统级编程、文件操作、字符串处理、位级运算、算术与逻辑运算,以及与其它语言的接口,适合于专家系统、规划和其它 AI 相关问题的求解[2]。

如果用传统语言,譬如.net, VC, VB 等来完成数学计算、数据库管理与用户界面的开发,而用 Visual Prolog 实现逻辑推理,就能互补优势。而关键就是两种语言如何连接的问题。正是针对目前这种实际情况,文中提出一种基于 Visual Prolog 内核,即通过其它语言调用 Visual Prolog 生成的 DLL 实现逻辑推理来构建专家系统。

1 Visual Prolog 6.3 DLL 生成

Visual Prolog 6.3 是最新一代完全面向对象的逻辑程序设计语言,它和传统的面向对象语言一样也有对象、类、多态、继承以及数据封装性等面向对象的概念[3]。在 Visual Prolog 6.3 中,对象模型的语义实体是对象、对象类型和类,有关这些实体的概念是接口、类的声明及实现[3]。接口是一组命名的谓词声明,描述了对象之间的"界面",是从一个对象之外进入到对象内部的入口,关键词用 Interface;对象由类产生,一个类包含类的声明和类的实现,关键词分别用 Class 和 Implement;在类的实现中必须对接口和类的声明中的每个公共谓词和构造函数提供定义,也可以声明并定义私有对象实体和私有类实体。

创建一个 diagnose.dll 文件的步骤如下:

(1)创建工程。

在 UI Strategy 项中选择控制台模式,因为不需要 GUI,免去了很多没用的代码;在 Target Type 项中选择

收稿日期:2006-03-15
作者简介:申晓勇(1982-),男,陕西佳县人,硕士研究生,研究方向为智能信息处理;雷英杰,教授,博士生导师,研究方向为智能信息处理与军用软件工程等。

DLL,其它项默认即可。

(2) 编译初始化。

通过编译,生成后缀名为.cl,.pro和.ph的五个文件,其中*.c文件描述类的声明;*.pro文件描述类的实现;*.ph文件是程序包的头文件(package headers),而程序包是类和接口的集合。

(3) 谓词及其子句的定义和设计。

动态链接库中定义有两种谓词:导出谓词(export predicates)和内部谓词(internal predicates)。导出谓词可以被外部或其它模块调用,内部谓词在定义它们的DLL程序内部使用,内部谓词的实现与传统Prolog没有区别。对调用而言关键就是如何声明导出谓词,Visual Prolog 6.3要求在文件diagnoseexport.cl中对导出谓词进行声明,而在diagnoseexport.pro中编写导出谓词的子句。导出谓词声明格式如下:

getstring:(integer Num,string IN) - > string Out procedure (i,i) language stdcall as "getstring"

这是一个有两个输入参数,一个返回值的谓词,也可叫做函数;它与内部谓词的区别在于增加了语言规范language stdcall(下一节详细介绍)以及别名as "getstring"来实现外部调用。

(4)目标段的定义和实现。

每一个DLL必须有一个入口点,在diagnose.pro的目标段中通过DLL=diagnose::new(),DLL:init()负责初始化和结束工作,每当一个新的进程或者该进程新的线程访问DLL时,或者访问DLL的每一个进程或者线程不再使用DLL或者结束时,都会调用目标段。

(5)编译生成。

通过编译,可以在exe目录下生成一个需要的diagnose.dll文件,把此文件拷贝在主程序的目录下即可运行。

为什么没有文件模块定义文件(*.def)也可以实现函数导出呢?模块定义文件是一个或多个用于描述DLL属性的模块语句组成的文本文件,每个DEF文件必须包含以下模块定义语句:EXPORTS语句。EXPORTS语句列出被导出函数的名字,将要输出的函数修饰名罗列在EXPORTS之下,这个名字必须与定义函数的名字完全一致,如此就得到一个没有任何修饰的函数名了。实际上在diagnoseexport.pro中通过#export diagnoseexport.pro编译器命令已经实现了以前需要添加*.def文件来实现的功能。

2 DLL调用

Windows系统平台上提供了一种完全不同的较有效的编程和运行环境,你可以将独立的程序模块创建为较小的动态链接库文件,并可对它们单独编译和测试。在运行时,只有当EXE程序确实要调用这些DLL模块的情况下,系统才会将它们装载到内存空间中。这种方式不仅减少了EXE文件的大小和对内存空间的需求,而且使这些DLL模块可以同时被多个应用程序使用,从而实现不同语言间的调用。

DLL模块中包含各种导出函数,向外界提供服务。DLL可以有自己的数据段,但没有自己的堆栈,使用与调用它的应用程序相同的堆栈模式;一个DLL在内存中只有一个实例;DLL实现了代码封装性;DLL的编制与具体的编程语言及编译器无关。

2.1 调用方式

DLL调用方式分为静态调用和动态调用两种。

(1)静态调用方式。这是一种隐式调用,由编译系统完成对DLL的加载和应用程序结束时DLL卸载的编码,简单实用,但不够灵活,只能满足一般要求。

把产生动态链接库时产生的.LIB文件加入到应用程序的工程中,想使用DLL中的函数时,只需说明一下。程序员在建立一个DLL文件时,链接程序会自动生成一个与之对应的LIB导入文件。该文件包含了每一个DLL导出函数的符号名和可选的标识号,但是并不含有实际的代码。LIB文件作为DLL的替代文件被编译到应用程序项目中。当程序员通过静态链接方式编译生成应用程序时,应用程序中的调用函数与LIB文件中导出符号相匹配,这些符号或标识号进入到生成的EXE文件中。所有被应用程序调用的DLL文件都会在应用程序EXE文件加载时被加载到内存中。可执行程序直接通过函数名调用DLL的输出函数,调用方法和程序内部其它函数是一样的。

(2)动态调用方式。这是一种显式的调用,由编程者用API函数加载和卸载DLL来达到调用DLL的目的,使用上较复杂,但能更加有效地使用内存,是编制大型应用程序时的重要方式。

在应用程序中用LoadLibrary或MFC提供的AfxLoadLibrary显式地将自己所做的动态链接库调进来,动态链接库的文件名即是上面两个函数的参数,再用GetProcAddress()获取想要引入的函数。自此,你就可以像使用如同本应用程序自定义的函数一样来调用此引入函数。在应用程序退出之前,应该用FreeLibrary或MFC提供的AfxFreeLibrary释放动态链接库。程序员可以决定DLL文件何时加载或不加载,显式链接在运行时决定加载哪个DLL文件。

2.2 调用约定

调用约定决定在DLL中定义的函数参数传送时入栈和出栈的顺序,由调用者还是被调用者把参数弹出栈,以及编译器用来识别函数名字的修饰约定。函数调用约定有多种,经常用到的是Stdcall调用约定和C调用约定两种。

(1)Stdcall调用约定。函数的参数自右向左通过栈传递,被调用的函数在返回前清理传送参数的内存栈。

(2)C调用约定。按从右至左的顺序压参数入栈,由调用者把参数弹出栈。对于传送参数的内存栈是由调用

者来维护的,实现可变参数的函数只能使用该调用约定。

2.3 编码转换

由于采用不同的编译系统,这就涉及到编码方式标准及其转化的问题,目前有两种不同的编码方式标准 ANSI 和 UNICODE。ANSI 中的字符采用 8bit,而 UNICODE 中的字符采用 16bit。在软件开发中往往需要既支持 ANSI 又支持 UNICODE,这就要求在类型转换的时候,重新改变字符串的类型和使用字符串上的操作函数。

为此,经常遇到的编码转换方式有:

(1)宏定义方式。标准的 C 运行期库、Windows 以及.net 系列语言都提供宏定义的方式,只需在声明外部函数时添加关键词 UNICODE 即可。

(2)函数转换方式。譬如 VB 通过字符处理函数 StrConv(string,conversion)对两种编码进行相互转换;Visual Prolog 通过系统谓词 aSCIIZ_2_vb_String(string,string)把 ANSI 编码方式转换为 UNICODE。

3 实例验证

采用以上方法构造一个某型飞机的故障诊断系统,通过测试飞机各项参数的高低异常状态形成状态码,分析故障原因。

该系统用 VB.net 设计用户界面,同时对事实数据库以及参数值进行管理;测试数据形成状态码,然后调用 Visual Prolog 6.3 生成的 DLL,通过推理机返回故障码并判断故障类型,给出解释。

系统实现结构如图 1 所示。

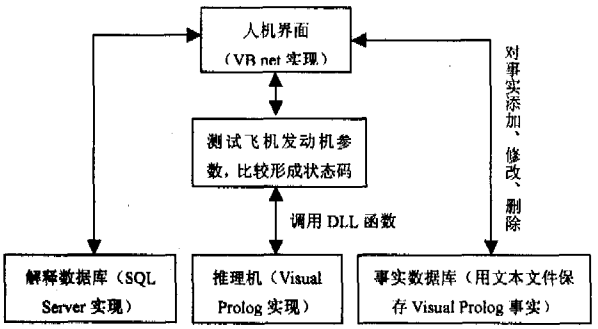

图 1　某型飞机故障诊断系统实现结构图

系统主要实现代码如下:

(1)DLL 函数调用。该系统采用静态调用方式,函数声明如下:

```
Public declare Unicode Function getstring Lib "diagnose"Alias "getstring"(ByVal integerin As Integer, ByVal stringin As String) As String
```

其中"diagnose"是要调用的 DLL 文件名,函数 getstring 申明之后在 VB.net 中的调用方法和程序内部其它函数是一样的。

(2)DLL 文件的内部结构。DLL 文件中自定义的代码主要在 diagnoseExport.pro 中,其代码如下:

```
#export diagnoseExport          %该编译器命令列出被导出函数的名字
implement diagnoseExport        %类的实现开始
class facts - mydatabase        %定义类事实
    rule:(integer,string,string).
clauses                         %子句段
  getstring(Num,String) = Out: - %调用函数 getstring 的定义
    retractall(_,mydatabase),%清空事实数据库
    file::consult("fa.txt",mydatabase),%将存储事实的文件 fa.txt 载入事实数据库中
    rule(Num,String,Out1),      %事实匹配,匹配成功则绑定变量 Out1 并返回
    string5x::aSCIIZ_2_VB_String(Out1,Out),%编码转换
    !.                          %若成功则截断,并返回故障代码,否则回溯直到结束
  getstring(_,_) = "00000000000000000000000000000000".%没有事实匹配则绑定一常量
end implement diagnoseExport    %类的实现结束
```

(3)事实数据库。

对于事实库用哪一种语言实现根据事实表示方法而定。如果事实表示方法采用一阶谓词逻辑,则事实库用 Visual Prolog 程序实现,这时程序量最少;如采用框架或面向对象的事实表示,则事实库宜用其它语言实现[4]。故该系统的事实库用 Visual Prolog 语言实现,将所有的事实保存在 fa.txt 文件中,通过 VB.net 打开文件对事实数据库进行维护。

4 结　语

通过以上论述以及相应的程序示例,可以看出这种新方法使两类语言完美结合,充分发挥各类语言的优势,缩短了开发专家系统的周期,而且使 Visual Prolog 语言广泛应用于具体工程,加快了其发展速度;同时在网络智能软件开发领域[5]以及计算机系统仿真和决策支持领域具有良好的应用前景。

参考文献:

[1] 雷英杰,张　雷,邢清华,等. Visual Prolog 语言教程[M]. 西安:陕西科学技术出版社,2002.

[2] 雷英杰,邢清华,孙金萍,等. Visual Prolog 编程、环境及接口[M]. 北京:国防工业出版社,2004.

[3] 雷英杰,邢清华,王　涛,等. 人工智能(AI)程序设计(面向对象语言)[M]. 北京:清华大学出版社,2005.

[4] 赖朝安,孙延明,郑时雄. 结合 C++与 Prolog 语言快速开发专家系统[J]. 计算机工程与应用,2002(3):30-32.

[5] 张　白,崔尚森. 应用 JNI 实现 Java 与 Prolog 的优势互补[J]. 交通与计算机,2004,22(6):119-121.