

基于构件技术的农业信息系统处理平台的研究*

张 建, 李 淼, 方 微, 黄永聪, 宋海生

(中国科学院 合肥智能机械研究所, 安徽 合肥 230031)

摘 要: 在对构件技术进行深入探讨的基础上, 给出构件化农业信息系统处理平台的设计思想, 提出了适合它的结构模型和构件描述语言、构件制作的原则、构件制作的算法, 并且结合具体的实例加以阐述。

关键词: 构件; 模型; 描述语言; 组装; 构件化软件开发方法; 平台

中图法分类号: TP391 文献标识码: A 文章编号: 1001-3695(2005)04-0058-04

Research and Implementation of Agricultural Information Manage Platform Based on Component

ZHANG Jian, LI Miao, FANG Wei, HUANG Yong-cong, SONG Hai-sheng

(Hefei Institute of Intelligent Machines, Chinese Academy of Sciences, Hefei Anhui 230031, China)

Abstract: Put forward the designing idea of the platform which is used to process agriculture information based on the in-depth study in component technology, presented its corresponding structural model and component description languages, as well as the principle and algorithm of component produce. Moreover, they were illustrated with concrete instance.

Key words: Component; Model; Description Language; Assembling; Component-based Software Development; Platform

1 引言

农业专家系统(AES)是农业信息技术研究中的一个重要部分。这项应用研究工作在 20 世纪 70 年代末期开始于美国, 我国从 80 年代初开始于中国科学院合肥智能研究所, 随后迅速引起了国内其他研究单位、大学以及农业生产管理部门的高度关注。在国家“863”计划的大力支持下, 取得了一系列的研究成果, 目前已经在全国 23 个省市得到应用。

农业专家系统是根据领域专家的知识 and 人工智能的推理机制相结合, 形成的一个实际应用系统, 用来求解农业生产管理方面的实际问题, 在一定程度上取代农业专家的作用。从专家系统的结构角度, 我们可以给出专家系统的定义: 一个专家系统是由一个四元组组成。ES = (P, R, L, K)。其中, P 是要解决的问题, R 为系统的推理控制策略, L 是学习机制, K 为知识库。

专家系统的这个定义比较明确, 其优点是四个部分相互独立, 任何一部分实现的具体方法不受其他部分的限制, 它确切地刻画了专家系统的本质, 在一定意义上体现出专家系统的结构, 当然, 作为一个实际使用的完整系统, 专家系统还应包括知识获取、人机接口和解释子系统等, 用于把专家的知识转换、加工为计算机内部表示, 使人和计算机(专家系统)用非常自然的方式对话。

经过近 30 年的研究与应用, 随着专家系统和软件技术的发展, 人们也看到了农业专家系统应用中存在的弱点, 如农业信息交流标准化问题、农业领域知识表示的局限性、重复开发现象严重、平台适应性差等问题。其研究内容随之发生了三点

值得注意的变化: 专家系统的概念不断向外延伸, 专家知识需要与各类实时发生的信息相结合, 形成完整的农业信息处理系统; 在大量资源数据电子化、形成数据库的情况下, 知识库与数据库的相互补充与利用显现出了重要的作用, 知识库与数据库之间需要桥梁; 在农业知识需要不断更新的情况下, 需要有利的技术手段或工具来支持和保障开发者和使用者获得知识的时效性和便捷性。

农业信息处理平台是针对上述第三个问题而提出的研究课题, 它的主要目的是利用软件技术, 为农业信息的处理提供软件工具。其核心思想有两点: 利用构件技术, 构造可自由组合与拆卸的农业信息处理件开发工具; 对农业专家系统的结构进行改造, 开放知识的信息源, 将静态的知识库变为动态更新。

在农业信息处理平台中, 利用专家系统和软件研究领域的多项技术, 提供一套具有骨架结构、用于生产管理决策的通用外壳程序和数据库通用接口、用于描述复杂的农业知识表示方法、知识库自动生成、推理机控制规则、可视化知识获取、农情资料数据管理、农作物知识库模板等一系列实用工具包, 使用者只需在框架内填充农业领域的知识以及信息, 连接上相应的资源数据库, 便可生成相应的各类应用系统。该平台的优点主要是: 构件技术为建造各类农业应用系统提供一个标准构架和一些标准构件及其接口, 通过构件技术来扩大知识库、加快农业应用系统的开发和更新速度, 同时可以使许多源出不同的、在解决农业生产领域中某一些问题具有独特优势的各类功能性构件在同一个环境下共同工作, 可以提高农业应用系统研制速度, 减少开发费用和不必要的重复劳动。因此它具有较高的研究、应用和实用推广价值。随着专家系统研究的深入、应用范围的扩大和软件技术的发展, 农业信息处理平台的研究已

成为专家系统领域研究的一个重要部分。

2 构件技术分析

构件的中心思想是将单个的应用程序分隔成多个部分,即构件(Software Component)。应用程序由多个这样的构件遵循一定的标准打包而得到的。每一个构件都可以在不影响其他构件的情况下被升级,这使得应用程序可以随时向前发展进化,这种方法被称为构件构架技术(Component Architecture Technology)。

2.1 构件概念的发展

构件是软件复用的基础,也是软件开发人员非常关注的问题。在不同的技术条件下构件的概念大体经历了三个阶段:

(1) 20 世纪 70, 80 年代构件主要是指可复用的程序代码片断,现在一般将它们称为代码件。这一时期如何利用已有的源程序代码、子程序和类库来提高软件开发效率是研发人员首要考虑的问题,主要包括以下几种形态: 子程序。作为构件的子程序包括程序设计语言提供的标准过程或函数,语言编译器提供的处理特定平台或硬件功能的标准子程序,专用领域的通用子程序(如数学函数库 MATLAB)以及开发人员在开发过程中积累的具有复用价值的子程序代码。 程序包。它将逻辑相关的实体或计算资源组成一个集合,是一种比子程序粒度更大的构件。 模板。它也称为参数化数据类型,是比函数或类更高一级的抽象,如在 Visual C++ 语言中包含大量标准模板函数和模板类。

(2) 20 世纪 90 年代初期构件主要指分析件、设计件、代码件、测试件,是可以被复用的软件实体,如 Microsoft 的 MFC, Borland 的 VCL, SUN 的 Java 类库,也称为软件资产。同早期的认识相比,这一定义将构件概念的外延大大拓宽了,主要包括以下几种形态: 设计模式(Design Pattern)。它记录了软件设计时经常使用的一些被实践证明是成功的思想、结构和方法。目前对设计模式的研究主要集中在 OO 领域,OO 设计模式是一种比单个对象和类粒度更大的结构,描述了对对象或类之间如何组织和交互来解决特定的问题。 框架(Framework)。它是一个可复用设计,由一组抽象类及其实例间协作关系组成。框架构件的典型例子是 Taligent 公司发布的一套支持快速应用开发的工具集 CommonPoint,其中包括上百个面向对象框架。

软件体系结构(Software Architecture)。它力图实现比框架粒度更大的系统级的复用。

(3) 20 世纪 90 年代中后期随着分布对象、Internet 等技术以及基于构件的软件开发模式的发展,国际学术界对构件的认识逐渐又有了新的变化。构件主要是指具有规范接口和确定的上下文依赖的组装单元,软构件能够被独立部署和被第三方组装。构件包含以下几个共同的因素: 构件是二进制功能单元。要想使基于构件的软件开发模式成为主流开发模式离不开大量软件开发商的参与和构件市场的形成,有效保护构件开发商智力投资是非常重要的,构件源码的公开显然不利于智力投资的保护,因此二进制代码就成为构件形态的首选。二进制代码隐藏了构件内部实现细节,也有利于减轻构件理解的难度,避免由于内部的修改而可能带来的错误。 符合构件模型(或具有规范接口)。符合同一构件模型是构件制作和组装的

基础,构件接口是复用者理解构件和构件组装的桥梁。一般来说,构件接口是构件模型的一个组成部分,因此符合一个得到业界广泛认可的构件模型是构件走向实用化的一个不可或缺的条件。 允许不同构件开发商开发的构件进行组装。构件要具有可插拔(Plug&Play)性,便于替换。

2.2 构件的分类

构件分类方法多种多样,下面将构件按多个侧面进行分类。

(1) 根据构件复用的方式,分为两种构件。 黑匣构件 Black,这种构件不需要知道其内部怎么工作,只需从外部使用它,能够达到即插即用; 白匣构件 White,需要经过修改才能重用。其中黑匣构件是发展方向。

(2) 根据使用的范围,分为通用构件 Common 和专用构件 Special。

(3) 根据构件粒度的大小,分为小型 Small,基本数据结构类构件,如窗口、按钮、控件、菜单等;中型 Medium,功能构件,如录入、查询、删除;大型 Large,子系统级构件,如人事管理,财务管理。

(4) 根据功能用途,分为系统构件 System,在整个构件集成环境和运行环境都使用的构件;支撑构件 Support,在构件集成环境、构件管理系统中使用的构件;领域构件 Domain,为专用领域开发的构件。

(5) 根据构件的结构,分为原子构件 Atom 和复合构件 Assemble。

(6) 根据构件重用时的状态,分为动态构件 Dynamic,运行时可以动态嵌入、链接的构件,如对象链接和嵌入(OLE)、动态链接库(DLL);静态构件 Static,如源代码、系统分析构件、系统设计构件等。当前常用到的是动态构件。

2.3 构件模型

构件模型是构件本质特征及构件间的关系的抽象描述。构件的本质特征是构件与软件复用相关的那些属性,主要包括三个部分: 构件对外提供的服务; 构件实体内部的实现策略,包括构件实体的实现语言或开发环境,构件内部哪些部分可以直接复用,哪些部分需要复用者加以实现等; 构件需要外界提供的服务,这一部分和构件间的关系描述反映了构件的组装信息。简单地说构件模型是为开发者定义软件构件而建立的体系结构和 API(Application Programming Interface)集,使开发者可通过软件构件的动态组合来建立应用系统。构件模型由构件与容器两种主要成分构成,其中容器用于存放和安排构件,实现构件间的交互,容器也可以作为另一个容器的构件使用。

目前,构件技术中广泛使用的构件模型主要有三种形式: 微软公司提出的 COM/DCOM; SUN 公司提出的 JavaBeans/ EJB; OMG 组织提出的 CORBA。前两种多用于 Windows 系列操作系统下,后一种多在 UNIX 系统下应用。因为我国目前 Windows(NT) 操作系统的用户较多,COM/DCOM 的应用较为广泛。

2.3.1 CORBA 模型

CORBA(Common Object Request Broker Architecture, 公共对象请求代理体系结构)是由对象管理组 OMG(Object Management Group)开发的。OMG 的中心任务是建立基于对象技

术并且是商业上可行的一个体系结构和一组规范,用于集成的分布式应用。其首要的目标是要达到在分布异构环境下基于对象构件的重用、可移植性和可互操作性。为达到这一目的,OMG 通过接口和协议规范来定义一个对象管理体系结构 OMA(Object Management Architecture),以支持基于分布互操作对象的应用。

2.3.2 JavaBeans/EJB 模型

SUN 公司 1996 年推出 JavaBeans 的构件模型。JavaBeans 是一个用 Java 语言编写的可移植的、与平台无关的构件模型。这样的构件可以在所有平台中使用。同时,JavaBeans 还提供了强大的无缝的连接方式,如通过 JavaBeans for ActiveX,可以使开发者在含有 ActiveX 控件的应用程序中建立 JavaBeans 构件,实现与 ActiveX 控件的无缝连接。2001 年 SUN 公司在 JavaBeans 的基础上,提出了应用服务器的构件模型与支撑环境 EJB(Enterprise JavaBean),并在此基础上进一步发展了多层企业级应用的开发与部署框架 J2EE。J2EE 是 SUN 公司推出的一种全新概念的模型,与传统的互联网应用程序模型相比有着不可比拟的优势,原因在于 J2EE 应用程序模型的一个主要优点就是在中间层的多层应用程序。

2.3.3 COM/DCOM 模型

COM(Component Object Model, 构件对象模型)是 Microsoft 创建的为解决构件之间进行通信的标准。COM/DCOM 自 1993 年首次发布以来,自身经历重大的变化,但是它的基本思想和结构一直保持相对稳定,它是当前比较流行的构件模型的标准。COM/DCOM 为构件的互操作定义了一种二进制构件模型,是一种底层的构件软件体系结构。对象与接口是 COM 规范中最核心的部分,COM 构件提供给客户的是以对象的形式封装起来的实体,客户程序与 COM 构件进行交互的是 COM 对象。COM 对象包括属性与方法,对象的状态反映了对象的存在,也是区别于其他对象的要素;COM 对象提供的方法就是对象提供给外界的接口,客户必须通过接口才能获得服务,所以接口是对象与外界进行交互的唯一途径。因此,封装特性是 COM 对象的基本特征。

3 平台中构件技术的应用

开放系统技术的基本原则是在系统开发中使用标准接口,同时使用各类符合接口标准的应用构件。它为一个系统的开发扩展提供了一个稳定的基础环境,也为系统(子系统)之间的互操作提供了保证。开放系统技术具有在保持系统效率的前提下降低开发成本,缩短开发周期的潜能。

平台根据构件的技术规范,设计并实现了即插即用的接口标准以及构件的登记、注册和组装。构造了基于模式化知识表示的编译构件;基于案例的知识库推理构件;MapInfo 地理信息系统的后台服务构件;数据访问对象构件;可以使源出不同的功能软件在同一个环境中发挥各自的优势,保证系统的开放性和通用性。

3.1 构件注册

与任何一个应用程序一样,所有的构件都有一个名称,平台在工作的时候,也就是需要对各构件进行调用的时候,它首先需要知道的是在它的环境中有哪些构件可供它调用。因此

平台必须为用户提供两个方面的服务: 对构件进行注册;获取注册表信息。

在 Windows 操作系统中注册构件的方法一般有以下两种:

(1) 直接使用 Windows 提供的对象注册工具 Regsvr32.exe 对该文件进行登录注册。使用该控件时,根据系统注册表定位控件。

(2) 自己编写注册文件。由于 DLL 知道它所包含的构件,因此 DLL 可以完成这些信息的注册。通过 LoadLibrary 装入相应的 DLL,包含构件的 DLL 将输出 STDCALL DLLRegisterSvr(); STDCALL DLLUnregisterSvr(); 通过 GetProcAddress 获取以上函数的地址,并调用它们,从而实现对某个构件的注册和注销。

我们利用方法(2)的原理在开发平台中设计了一个注册/注销控件功能窗体,它向用户的系统注册表添加/移去 ActiveX 控件(*.ocx)、进程内构件服务器(*.dll)、进程外构件服务器(*.exe)。主要步骤如下:

注册。用户选定所需要注册的控件文件所在的目录,将控件注册到系统注册表中,并提示是否注册成功。

注销。在控件列表框中,找到需要注销的控件,从系统注册表中注销掉,并提示是否注销成功。

重新注册。对控件重新注册,可以用于控件的升级。

查找所有已注册的控件。将本机内所有已注册的控件显示在列表框中,该列表框将显示出控件的 ProgID 以及相应的 Path。

3.2 构件参数的浏览和获取

在该平台中,提供了一种知识库管理与开发语言(KML),用户所需要调用的构件将通过 KML 进入知识库的编辑和推理过程。

应用构件注册后,开发工具中的构件参数浏览器通过注册表来获取控件的 CLSID 或 ProgID,并调用 CoCreateInstance 函数,生成控件的实例,返回一个指向构件的 Iunkown 接口的指针。一般情况下是通过此 Iunkown 接口的 QueryInterface 函数来查询该构件所支持的接口指针。但由于开发工具在调用该构件之前,并不知道 QueryInterface 的实现,因此,它无法知道一个构件所支持的所有接口以及它所支持的函数。所以开发工具了解构件所支持接口的唯一方法是获取构件中所有的参数。这一点与 C++ 有很大的不同,在那里,调用构件的客户可以通过头文件来得到接口中函数的所有参数。为了使开发工具能够对它所不能识别的接口进行处理,需要使用者拥有该控件的技术文档,然后编写处理此接口的代码。这种方法一方面可能由于技术文档不全、没有或遗失而不可行;另一方面让使用者在不了解接口实现的情况下编写处理接口的代码几乎是不可能的。为此,我们借助 COM 提供的构件类型库和调度接口来解决构件参数的获取和构件的动态调用。

利用类型库的特点,我们在开发工具中编制了一个浏览控件窗体。通过浏览控件的类型库信息,获取其方法及参数信息,为接收构件参数及调用方法做好准备。这个窗体包括了两个方面的功能: 显示本机所有已经注册的构件; 获取所需要调用的构件中方法及参数的信息。

具体实现步骤如下:

(1) 打开包含构件的文件。用户在利用开发工具向欲开发的农业专家应用系统中添加专用构件的第一步是打开那些包含构件的文件(OCX, DLL, EXE) , 装入类型库, 获得 ITypeLib 接口指针。

(2) 列出包含构件的文件(OCX, DLL, EXE) 中所有的构件。供用户选取所需要的构件, 并获取该构件的名称 CoClass 及说明信息。

(3) 列出所选构件中的所有实现的接口, 供用户选择, 通过指定 GUID 来获取 ITypeInfo 接口指针, 并获取该接口的相关信息 pInterfaceInfo。

(4) 列出所选接口中成员的名称, 获取函数描述结构。

(5) 列出所选成员中所有参数的类型, 获取参数描述结构。

3.3 构件的动态调用

通过上述的浏览控件获取到了所选构件的 CLSID 或 ProgID 以及方法名和参数信息后就可以调用控件的方法了。

调用控件方法又可分为静态和动态两种。静态调用控件是在应用程序发行前就把要调用的控件包含在程序中, 然后通过编译、链接后发行。这种方法显然不够灵活, 不能适应我们的要求, 因此需要对控件进行动态调用。所谓动态调用就是在应用程序开始运行后能够根据用户需要或命令动态加载控件, 并调用它的方法。实现控件的动态调用存在不少难点, 控件类型不一, 有的不带窗体, 有的包含窗体, 如何在主程序中载入控件窗体、该控件窗体处于何处、窗体如何响应事件、控件如何与主程序进行数据交换?

考虑到以上难点, 实现动态调用的方法有两种:

(1) 用 IE 的方法。该方法能自动发送消息到注册的 COM 构件, 从而生成控件实例。它的一个主要优点是对技术细节要求较少, 因而编程量小, 但是控件窗体的位置及其与开发平台中的数据交换是一个主要的难点。

(2) 用自动化调度接口方法。这种方法绕开了控件窗体的载入及其对事件的响应, 直接对生成控件对象的内部函数进行调用。该方法的主要优点是由于其避开了窗体生成与事件响应, 深入到控件内部获取它的信息, 并对其进行控制; 另外由于它对控件生成、调用的技术细节要求较多, 尽管编程量大, 但由于独立于 IE 控件, 可以不受 IE 控件变动的影响, 同时也由于深入到内核, 具有明显的技术优势。掌握了控制构件的技术细节后, 一方面可以生成独立的控件测试器; 另一方面也可以根据需要嵌入控件窗体, 支持控件事件等。可以说这种方法现阶段编程量大, 但对构件技术的深入了解打下了良好的基础。

自动化不是独立于 COM 而是建立在 COM 基础上的。一个自动化服务器实际上就是一个实现了 IDispatch 接口的 COM

构件。一个自动化控制器则是一个通过 IDispatch 接口同自动化服务器进行通信的 COM 客户。自动化控制器不会直接调用自动化服务器实现的各种函数, 而是通过 IDispatch 接口中的成员函数实现对服务器中函数的间接调用。自动化使用解释性语言和宏语言访问 COM 构件更为容易, 因为它仅关注构件运行时的类型检查, 这一点是以牺牲速度和编译时的类型检查为代价的。开发工具利用它的特点, 通过自己编写的代码来生成构件接口, 从而替代编译器生成构件接口的工作。

在 IDispatch 中有两个最主要的函数, 即 GetIDsOfNames 和 Invoke。GetIDsOfNames 将读取一个函数的名称并返回其调度 ID 或称作 DISPID。为了执行这个函数, 自动化控制程序将把函数的 DISPID 传给 Invoke, 并且由 Invoke 来调用相应的函数。编译器将在编译时生成对 IDispatch 成员调用的代码, 此代码将是静态的。但是传给 Invoke 的参数将决定所调用的函数是什么, 这些参数同其他参数一样, 是动态的, 可以在运行时灵活变化。我们就是利用这个原理, 实现了构件的动态调用。

4 结束语

软件复用和开放系统技术本身还是一项需要不断提高的技术, 我们在将这些技术应用于开发工具中还存在一些问题需要解决: 构件的动态包容问题。目前对构件包容是通过接口描述语句以指针地址和数据黑板的方式进行, 这是一个静态的过程。我们计划设计一个构件包容器, 它将所有已提供的构件功能动态地提供给构件的插拔者, 供随时使用。 开发工具自身的构件化还需要加强解决继承性和通用性的问题。

参考文献:

[1] 杨芙清, 等. 软件复用及相关技术[N]. 计算机世界, 1999-03-15.

[2] 李森, 张建, 等. 关于开放式农业专家系统平台的研究[C]. 99 智能计算机接口与应用进展, 1999. 416-421.

[3] Zhou Yuan-chun, Li Miao, *et al.* Dynamic Call of the Component on AESP-Agricultural Expert Systems Platform[C]. Beijing: Intemation Symposium on Intelligent Agricultural Information Technology, 2000. 241-244.

[4] 李小欧, 李森, 等. 基于 ActiveX 文档的农业专家系统网上推理技术[J]. 微电子学与计算机, 2002, 19(1): 49-53.

[5] 贺卫红, 何映, 曹毅. 异构环境下分布组件的对象互操作性的研究[J]. 计算机应用研究, 2004, 21(2): 79-82.

作者简介:

张建(1954-), 男, 研究员, 主要研究方向为人工智能与软件工程; 李森(1955-), 女, 研究员, 主要研究方向为人工智能中的知识与信息处理等; 方微(1978-), 女, 助理研究员, 主要研究方向为数据库与组件技术; 黄永聪(1981-), 男, 在读研究生, 主要研究方向为模式识别与人工智能; 宋海生(1978-), 男, 在读研究生, 主要研究方向为模式识别与人工智能。

(上接第 17 页)

[37] aniel M, Willsky A. A Multiresolution Methodology for Signal-level Fusion and Data Assimilation with Applications to Remote Sensing [J]. IEEE, 1997, 85: 164-180.

[38] Riedi R, *et al.* A Multifractal Wavelet Model with Application to Network Traffic[J]. IEEE Tran. Inform. Theory, 1999, 45: 992-1018.

[39] Fosgate C, *et al.* Multiscale Segmentation and Anomaly Enhancement of SAR Imagery[J]. IEEE Tran. Image Processing, 1997, 6: 7-20.

[40] Irving W, Novak L, Willsky A. A Multiresolution Approach to Discrimination in SAR Imagery[J]. IEEE Tran. Aerosp. Electron. Syst. ,

1997, 33: 1157-1169.

[41] Kim A, Krim H. Hierarchical Stochastic Modeling of SAR Imagery for Segmentation/Compression[J]. IEEE Transactions on Sigal Processing, 1999, 47: 458-468.

[42] Kolaczyk E, Huang H. Multiscale Statistical Models for Hierarchical Spatial Aggregation[J]. Geogr. Anal. , 2001, 33: 95-118.

作者简介:

温显斌, 副教授, 博士研究生, 主要研究方向为信号与图像处理、多分辨率分析技术; 田铮, 教授, 博士生导师, 主要研究方向为信息与图像处理技术、动态序列分析。