计算机科学1995 Vol. 22№. 4

用对象实现构件的原型开发方法

姚淑珍
(北京航空航天大学计算机系 北京100083)

摘 要 Based on general properties of object-oriented Languages, the paper analyzed many advantages of objects representing components and gives a prototyping method with components implemented by objects. This method involves outer interface of a component, "relationship" function between components and optimization plan in the process of prototype generation.

关键词 Prototyping, Object-oriented.

1. 引 言

原型开发方法是针对传统的瀑布式开发方法弊端,为充分获取用户需要而提出来的[1]。原型开发方法是指开发者获取用户最初需求后,利用高级软件工具和开发环境快速建立起目标系统最初版本提交给用户试用,用户可对此提出评价和补充意见,改进后的新版本或作为目标系统模型,或被完善成目标系统[2]。原型开发模式强调充分利用领域知识——即知识重用,因此说重用是实现原型开发支持环境的主要手段。

虽然重用思想被广泛采用,但具体应用中会暴露出种种问题,主要为:

·原型法适于描述非确定性需求,但对于这种需求,很难决定其中哪些成分对应库中构件,以及什么样构件(构件名称)。

·对不完全适合的构件进行调整的费用可能比开发费用还要大,调整后或重新开发的构件应如何处理。

以上问题要求系统在构造原型时,能提供给用户友好的启发式界面,引导用户调整需求描述以及周期性地提取用户知识。我们提出的原型开发方法具有联机帮助和择优机制以协助用户查找最佳构件,同时还提供调整与重新编制两种手段,并将中间构件存放在用户区。开发结束时,对其整理,转入系统构件库中。整个原型生成过程中构件的表示形式至关重要。我们在分析了面向对象的基本特征之后,认为用对象来实现构件最能体现领域知识的重用性。

2. 面向对象语言的基本特征

重用技术主要分两种:生成技术和合成技术。生成技术机制是用一个生成程序接受可重用模式,据此生成新程序。当接受的是代码模式时,这种生成程序就是应用程序生成器;当接受的是规则模式时,这种生成程序就是变换系统[3]。前者适于开发应用领域知识相对明确的情况,后者适于能形式化地描述系统需求的情况。无论是应用程序生成器还是变换系统,所接受的用户需求都必须是完备、正确的,它们都不支持原型开发模式,因为原型开发模式主要解决的是如何全面、确切地理解用户需求并结合用户领域知识,由构件组装、改进成目标系统,以此进行原型开发更能体现开发过程的快速和所开发原型的可扩充性的特点。

利用面向过程语言实现构件库时,构件使用者必须完全按构件名及其参数模式访问构件,这虽然保证了使用申请和服务二者间的兼容性,但同时却降低了构件使用的灵活性,从而限制了构件的重用范围。相反,面向对象语言在提供服务方面具有更大的灵活性,在面向对象语言中,使用申请发给一个对象,真正的服务形式由该对象来决定,因此,在使用申请中可以动态地改变对象类型,或者由系统按申请模式自动选择服务的构件[4]。在这里构件具有较高的抽象级,可以是一类操作,这在组织构件库方面有利于知识的分类和对构件库可控性的保证。

面向对象语言除具有以上特性外,它所提供的

抽象、封装、动态联编和可继承性都有助于构件库的实现。

软件工程的进展总是和抽象程序的提高紧密相连。广义上来讲，抽象是对复杂的现实世界的简明表示，它强调我们所关心的，忽略不重要的，即只关心高层次上的设计思想。层次越高，则抽象性越强。在面向对象语言中不仅可把对象的外部操作与内部实现分开，而且可用类达到更高层的概括和抽象，为开发者提供了更自然的解决问题方式。

封装是一种信息隐藏技术，封装体应具有友好的外部界面，说明封装体间的相互作用和关系，内部信息则是隐藏的。在面向对象语言中，对象和对象类都是良好的封装体，对象类设置成实例后，除继承共性外，还定义仅由该对象所私有的特性，因此，对象封装比类封装更具体、更细致。封装概念同样提供如何将一个问题的各个成分组装在一起的求精过程，用户可以对象为基本封装体。对其组装得到不同视图，即封装体视图[4]。用封装体表示构件将明显使构件可控性大大增强。

在面向对象语言中多态性体现在同名操作可在不同时间保存、取用以及返回不同的类型值。主要表现在：运算符重载、函数名重载以及虚函数与动态联编。后者允许在运行时(而不是编译时)才按照具体数据类型和参数来确定选用哪一个函数，增强了软件开发过程的灵活性和重用性。继承性可扩充并调整对象行为，使得构件在不改变最初实现细节下适应于各种应用领域，即保证了构件的可扩充性，综上所述，面向对象语言的诸多特性使得用它来实现构件库最为合适，同时面向对象的设计思想也使得领域知识以最自然的方式表达出来。

原型开发方法中应明确的三个关键问题是：1)如何设计构件的接口协议?2)如何管理构件库?3)如何用构件合成原型?

3. 构件的接口协议

构件不仅要提供给用户一个友好的界面，从构件表示领域知识角度来看，它在计算机中的存贮方式上又应体现知识的层次关系，因此构件接口应包括外部接口和内部接口两种。外部接口用来描述对象的规格说明及开发信息，它应包括以下内容[4]：

·查找项：对构件进行检索的信息，包括作者名、开发日期、最新更改日期和构件名称等。

·基本信息项：描述构件的基本特征，包括构件种类、规模、版本、源代码文件名、测试代码文件名和数据结构等。

·功能描述项：对功能进行概述。

·用户接口项：带有操作语法和信息语义的操作集合。

·使用格式项：实例

构件库组织完全遵守内部接口协议。为表示知识的层次性，我们用多个无环有向图来组成构件库。有向弧的终止点是起始点的具体化，两者间联系程度用二元函数 ω 表示[5]，ω 含义为：

$$\omega(x_1,x_1)=0$$

$$\omega(x_1,x_2)=\omega(x_2,x_1)$$

$$\omega(x_1,x_2)=\min_k\{\omega(x_1,x_{k1})+\sum_{i=1}^{i-1}\omega(x_{ki},x_{k(i+1)})+\omega(x_{kj},x_2)\},k=1,2,\cdots,n$$

若将无环有向图中任一条有向弧(x_1,x_2)都规定加权函数值 $\omega(x_1,x_2)$，则可通过上式求得图中任两点的 ω 值，并以此作为两点关系程度的度量。可结合经验数据来规定 ω 取值，这样，构件的内部接口相应地可采用以下描述格式：

类号	类名	功能名	子类名	与子类 ω 值

4. ω 函数值的获取

ω 函数是知识间相互联系的量化表示，它启发人们如何进行联想查找和最优构件组合，所以创建构件库时，一方面确定组成构件，另一方面应明确构件间关系，即 ω 函数值。初始值可在建库时由领域专家确定。这里我们提出一种基于模糊理论的专家评判方法[6]。在这种方法中，首先确定评价指标 $U=\{u_1,u_2,\cdots,u_n\}$ 和评语集 $V=\{v_1,v_2,\cdots,v_m\}$，然后用下面模糊矩阵来表示多专家对多评价指标的每一评语的百分比：

$$\underset{\sim}{R}=\begin{bmatrix} r_{11} & r_{12} & \cdots & r_{1m} \\ r_{21} & r_{22} & \cdots & r_{2m} \\ \cdots & \cdots & \cdots & \cdots \\ r_{n1} & r_{n2} & \cdots & r_{nm} \end{bmatrix}$$

利用模糊合成运算 $A*\underset{\sim}{R}$ 求出 $B=(b_1,b_2,\cdots,b_m)$，$b_j=\sum_{i=1}^{n}a_ir_{ij}(j=1,2,\cdots,m)$ 作为多专家评价结果。

例如，给定 $U=\{u_1,u_2,u_3\}$，其中：u_1：子类对父类的实际继承程度；u_2：子类相对于父类的未扩充程

度；u_3：子类对于父类的未更改程度。V＝{1，2，3，4}，其中，1，2，3，4分别表示程度级别为1级、2级、3级和4级。对实际继承程度评价时，有50%的专家认为属于1级关系，40%的专家认为属于2级关系，10%的专家认为属于3级关系，没有人认为属于4级关系，则对实际继承程度的评价为(0.5，0.4，0.1，0.0)，类似地得到对扩充程度的评价为(0.4，0.3，0.2，0.1)，对更改程度的评价为(0.0，0.1，0.3，0.6)，这样就得到一个评价的模糊矩阵：

$$\underset{\sim}{R}=\begin{bmatrix}0.5 & 0.4 & 0.1 & 0.0\\ 0.4 & 0.3 & 0.2 & 0.1\\ 0.0 & 0.1 & 0.3 & 0.6\end{bmatrix}$$

在指定了评价中三个评价指标的权向量A＝(0.5，0.2，0.4)后就可以求出向量B，归一后得(0.33，0.29，0.18，0.20)，这表明有33%的专家认为该子类与父类关系属1级，29%的专家认为该子类与父类关系属2级，18%的专家认为属3级，20%的专家认为属4级，综合来看，把它们之间的关系定为1级最合理。

5. 构件库的管理

从以上接口协议可以看出，知识表示是一个无环有向图集合，每个有向图代表一类知识，中间结点是对象，叶结点是中间结点提供的操作，在每个有向图的起始点都有一个伸向“用户定义”节点的弧，用于存放本次开发用户临时编制并在开发过程中可能重用的构件。软件开发结束时可对这些“用户定义”的构件进行审查及测试，从而决定是清除掉，还是作为标准可重用构件保存下来。

构件库管理系统应提供对构件库的增、删、改和查询等操作。构件库的组织可以在应用领域确定下来以后，由专家通过对领域知识的提取、分类和归纳来完成，也可吸收已有知识或在原型开发过程中逐渐积累起来，后者必须有专门的质量保证小组对构件进行测试和验证后才允许加入到构件库中。构件库的改删操作也要有这种质量保证手段。当对构件库进行增删改操作时，节点间的ω值也要做相应调整。另外，构件库还应附带一个同义词表，以达到申请和服务的安全统一。

6. 原型合成

利用构件合成原型过程可分为以下几步：

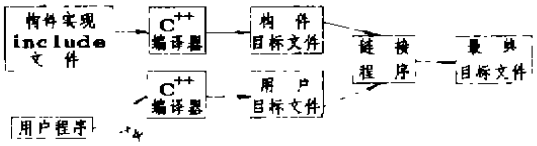
1. 从需求中识别出所有对象实体及其动作，

2. 对每一实体按其类别映射到构件库中相应有向图上。

3. 通过检索构件的外部接口择优选择可供使用的构件。

4. 检查构件能否直接引用，a. 若能，则进行必要的设置；b. 否则，自己设计并放到“用户定义”节点下。

5. 按以下方式合成：a. 开发一主程序；b. 编译所有用户定义的构件程序；c. 链接可重用C++构件目标文件和用户目标文件，整个过程如下：



上图中采用构件实现，旨在限制用户对构件进行修改。先分别编译再链接的目的是为了以后对构件的改进不影响原来对它的引用[4]。

第3步中择优的对象是这样一些构件，这些构件从属于一个有向图且都满足用户某一使用申请要求，只是各有特性，比如对象的数据结构表示不同等。可以人为地按构件的外部接口选择理想构件，也可以由原型开发环境辅助自动完成。下面简述一下原型开发环境的这一实现机制。用户的每个操作都可能对应一个构件候选组，多个操作就对应多个构件候选组，每个候选组中取一个构件就组成了用户操作的映射构件集，择优原则就是保证映射构件集最为“集中”(“集中”极限就是它们都从属于同一对象)。可以扩展ω函数作为衡量“集中”的准则。扩展ω函数规定如下：

对于n个构件$a_1，a_2，\cdots，a_n$，规定

$$\omega(a_1，a_2，\cdots，a_n)=\sum_{i=1}^{n}\sum_{j=1}^{n}\omega(a_i，a_j)$$

假如用户操作的映射构件集有K个，对应的扩展ω函数值为$\omega(a_{11}，a_{12}，\cdots，a_{1n})$，$\omega(a_{21}，a_{22}，\cdots，a_{2n})$，$\cdots$，$\omega(a_{k1}，a_{k2}，\cdots，a_{kn})$，则对应的$\omega_l(\omega_l=\min_{i=1，\cdots，k}(\omega(a_{i1}，a_{i2}，\cdots，a_{in})))$的映射构件集$(a_{l1}，a_{l2}，\cdots，a_{ln})$为最佳的构件集。

7. 结　论

我们所开发的原型开发支持环境已具雏形，用

(下转第78页)

调整其目标函数，使之均衡一致，不断地使决策达到满意的效果。由于设计与制造分属两个不同的知识库所控制，在设计阶段，要满足产品性能和功能要求必须制定较为严格的公差；在制造阶段，为了降低对设备和操作要求并减小制造成本，希望有较大的公差范围；在装配阶段，为了便于工装，希望零件具有较好的互换性，对公差要求十分严格。面向制造的设计就可以权衡和协调这些关系，对产品特性及工艺规划进行优化，使产品容易制造和装配，减少检验工序工作量。从而简化产品设计，降低成本，减少试制费用，缩短产品研制时间，使设计更趋标准化。

3. 变换矩阵

从设计状态到制成满足一定功能要求的产品状态存在着一定对应关系，我们用一个变换矩阵 A 来描述，可以表示成：

$$[FR]=[A][DP] \quad (1)$$

其中，DP 表示设计参数集，FR 表示产品功能要求，它们既有参量的描述，又有概念型的语义描述。

(1)式提供了从特征空间到功能空间的映射，矩阵 A 中各元素提供了各种特征参数与功能要求之间的数值和语义关系，它是设计信息和加工、装配、检测信息的集成。通过对变换矩阵 A 的完备性分析及相干性检查，对相互关联和牵制的设计参数进行重新定义和优化调整，尽量使设计参数相对独立地实现其对应功能要求，消除冗余过程，达到各个过程的协同性，从而找到一种最大限度地降低产品制造复杂度的途径。同时，据 FR、A 可以推导出 DP，在进行新产品设计时，由用户提出产品的功能要求，结合以前产品的设计经验，搜索类似产品的变换矩阵，则可以使零件参数化，形成设计草图和较为粗糙的方案，供设计者参考，这样就节省了许多重复性的工作，降低设计工作的劳动强度，设计者还不断提炼变换矩阵，集类似产品的优点于一体，较容易地设计出让用户满意的产品。

五、结束语

并发设计自动化环境的研究意义在于：探讨如何协调多领域、多专业专家求解设计问题的思维过程和合作求解模式，为 CAD/CAM 集成，并发设计在 CIMS 中应用提供了一种基本理论和技术方法。本文提出的并发设计系统结构和求解策略的特点是：(1)多领域知识的集成化表达；(2)为达到最高目标要求，进行多种突发事件综合决策以及不同观点和结论的协调、控制；(3)并发设计系统结构单元之间以及系统与外部环境之间动态连接并进行信息交换，实现了决策自动化的目的。(参考文献转第50页)

(上接第68页)

户界面基本上是改进后的 C++ 语言成分集合。首先系统对其中确定成分调出相应构件，非确定成分引导用户查找最为满意构件。这种方法对用户要求水平过高，从根本上摆脱不了语言的约束。现在我们进行的工作是进一步提高界面友好性，试图结合人工智能知识，接收用户的自然语言描述，分析出其中关键成分，与构件进行模式匹配，找出最理想构件[7]。这种思想的实现将会使原型法应用更趋于普及化。

参考文献

[1] 姚淑珍等，原型法及其支持技术，计算机工程，1992.5

[2] Sharram Hekmatpour, Darrel Ince, Software Prototyping, Formal Methods And VDM Workingham: Addison-Wesley Pub, Comp., 1988

[3] 林琪超编，软件开发环境，上海交通大学出版社，1991.5

[4] Deng-Ji Chen and P. J. Lee, On the Study of Software Reuse Using Reusable C++ Components, *Systems Software* 1993;20

[5] 姚淑珍，一种实用的原型开发支持环境，软件世界，1993.3

[6] 吴怀钊等著，模糊数学与计算机应用，电子工业出版社，1988.12

[7] Roy Rads et al., Software reuse: from test to hypertext, Software Engineering Journal, Sep. 1992