可重用构件及其描述语言*

全炳哲 余 江 金淳兆

(吉林大学计算机科学系,长春 130023)

摘要 本文讨论可重用构件应该具备的特性,并介绍一种可重用构件描述语言 Recos. Recos 支持面向对象设计,而且提供功能抽象和类属机制.

关键词 可重用性,可重用构件,面向对象,规格说明语言.

软件重用技术是提高软件生产率,缓解软件危机的重要手段. 根据可重用实体的不同性质,我们可以把重用技术分为合成技术和生成技术. 合成技术中重用的实体是重用构件,生成技术中重用的实体是转换规则或生成器.

在合成技术中,首先要解决的问题是要确定可重用构件的形式. 目前采用的可重用构件大体可分为子程序,Ada 包和面向对象设计中的类. 对于子程序构件没有较好的构件调整和组装机制. Ada 包作为构件时,可用类属机制作为构件调整的机制,包之间的引用关系作为组装机制. 面向对象设计中继承既是构件的调整机制,也是组装机制.

1 可重用构件

为提高软件开发的生产率,我们应该提供易于重用的构件. 我们认为可重用构件应该具备如下五个特性:

1. 用面向对象设计方法设计可重用构件:面向对象设计方法可以使问题自然映射到软件结构,因而可使软件易于理解. 类是抽象数据类型的一种实现,而且是一种程序模块. 这种模块之间的耦合度松散,有利于提高模块独立性.

2. 可重用构件应具有较高的抽象程度:构件的重用程度是指构件在开发各种软件时可以重用的难易程度. 构件越具体,其重用程度越低. 为提高构件可重用度,应该尽量把构件一般化,使软件开发人员重用这些构件的设计思想.

3. 构件应易于调整:可重用构件应该具有较高的通用性. 然而,开发特定软件时,必须把这些构件具体化,以用于确定的环境. 因此,必须提供构件具体化的机制,即调整机制.

* 本文1991年5月3日收到,1991年9月13日定稿

本文得到国家863计划的支持. 作者全炳哲,33岁,讲师,主要研究领域为软件重用技术,软件设计方法,软件工程环境. 余江,34岁,讲师,主要研究领域为软件重用技术,软件设计方法,软件工程环境. 金淳兆,57岁,教授,主要研究领域为软件重用技术,软件设计方法,软件工程环境.

本文通讯联系人:全炳哲,长春 130023,吉林大学计算机科学系

4. 构件应易于组装:基于可重用构件的重用途径,一般来说是从可重用构件库中选出若干构件组装所需要的软件.因此,由构件组装软件的难易程度是影响软件重用根本实用化的重要因素之一.因此,为了易于组装,除构件之间具有松散的耦合度外,还应提供便于组装的机制.

2 Recos 语言

Recos(REusable COmponent Specification)语言是一种可重用构件的描述语言.它是强类型的面向对象描述语言,同时也是一种程序设计语言.因此,Recos 语言不仅用于描述可重用构件,而且也可以用它编写目标软件.

2.1 可重用单位

Recos 语言中可重用构件分为类和模块两种.类是某抽象数据类型的一种实现,由它可以生成具体对象.类的描述由定义部分和实现部分组成.定义部分描述该构件的外部接口,包括继承哪个类,引用哪些构件,移出哪些常量,数据类型和规程(method)等信息.实现部分描述该类中的数据类型,示例变量和各个规程等的实现细节.Recos 中类,继承和消息发送机制来支持面向对象设计技术.

面向对象设计技术作为一种数据抽象的拓广得到越来越广泛的重视.数据抽象技术是以数据为中心设计软件的技术.这种技术有利于提高软件的易修改性和易理解性.但是,在实际软件设计中,不仅需要有数据抽象机制,而且常常需要有功能抽象机制.例如,输入/输出功能,排序算法,编译程序中的语法分析和语义分析等属于功能.用数据抽象机制描述这些功能性的构件反而显得不自然.因此,如果软件设计中只允许采用类机制,则往往是不方便,甚至有些情况下是不合理.我们认为软件开发过程中应该尽量用面向对象设计技术,但是确属功能性的部分应该用功能抽象.另外,数据抽象也不一定全部用类机制来实现,当由某个类只需生成一个对象时,应该采用类似 Ada 语言中包机制来实现数据抽象.这样可以使"解"和"问题"更加接近.为此,Recos 语言中引入了模块概念.它类似于 Ada 语言中的包机制.模块的描述也分为两部分,即定义部分和实现部分.模块内部可以说明其数据类型,内部变量和若干相关的子程序(Recos 中也称它们为规程).它和类不同在于模块只限于描述某个特定的对象,而不能用于生成对象.在 Recos 中,为了尽可能地重用已有信息,同时为了便于组装,允许模块继承其它模块.

2.2 类型问题

作为工程化的语言,为了提高软件可靠性,应该是强类型的.面向对象语言中可用类实现记录型数据和数组,但是程序中所有记录型数据和数组都用类描述有时会增加软件复杂性,降低软件结构的清晰性.例如,链表的实现问题等.为此 Recos 语言除了提供基本类型(整型,实型,字符型和布尔型)外,还提供了记录型、数组型和指针型.另外,Recos 中把类作为一种数据类型来处理,用类定义的变量称为对象变量.作为一种强类型的面向对象语言,为了不仅要体现强类型的特点,而且要充分体现面向对象语言的灵活特性,Recos 语言的赋值规则允许为某个对象变量赋值由该对象所对应的类的子类生成的对象.

2.3 可重用构件的级别

Recos 语言中重用构件可以带有类属参数,这些类属参数可以是常量参数、类型参数和规程参数.我们把带有类属参数的构件称为模板构件.当用户使用模板构件时,首先需要确定类属参数的具体细节.为了编写抽象级别较高的构件,Recos 还提供了抽象规程的机制.抽象规程是指由相应子构件中完成其实现细节的规程.因而,抽象规程的定义中没有规程体的描述.抽象规程只用于描述某个类或模块的外部接口.通过具有抽象规程的构件可以描述一类构件的规范.利用抽象规程机制和赋值相容规则,我们可以编写出相当通用的重用构件.

利用类属和抽象规程机制我们可以描述三个级别的重用构件.其一是构件中的常量和类型都是类属的,而且相应规程也都是类属或抽象的.这种构件的重用是设计思想的重用.另外,利用这种构件之间的移入表和继承关系,可以描述软件初步设计,其中规程的算法可用注解形式的 PDL 来描述.其次是构件中某些部分是类属的构件,对这种构件可以重用其设计思想之外,还可以重用某些变量和规程的具体实现.第三种是不包含类属参数的构件,这种构件的重用是具体实现的重用,即代码的重用.

2.4 已有子程序库的重用

目前,人们开发了相当丰富的通用子程序.Recos 程序中可以嵌入用某种程序设计语言编写的代码,用它可以直接重用已有的子程序库.用模块机制也可以重新组织已有子程序库,以利于重用.

3 举 例

下面我们举栈的例子说明如何用 Recos 语言描述重用构件.程序 1 是用 Recos 语言描述的抽象栈.

```
class definition:stack;
inherit:Class;
export:elemtype,init,empty,push,
       pop,top;
parameter:elemtype;
type
  elemtype=deferred;
procedureinit();
function empty():boolean;
procedure push(elem:elemtype);
procedure pop();
function top():elemtype;
end stack;

class implementation:stack;
procedure init()
  deferred;
    ...
function top():elemtype
  deferred;
end stack;
```

程序 1

```
module definition:aModule;
inherit:Module
import:stack;
export:proc1,proc2;
parameter:proc1;
procedure proc1();
procedure p2(aStack:stack);
end aModule;

module implementation:aModule;
var
  elem:stack.elemtype;
procedure proc1()
  deferred;
procedure proc2(aStack:stack)
begin
  ...
  if (not aStack! empty()) then
  elem:=aStack! top();
  aStack! pop();
  end;
  ...
end;
end aModule;
```

程序 2

stack 类继承系统中已定义的类 Class，它是所有类的抽象. 定义部分有移入表和移出表. 元素类型 elemtype 是类属类型. 对栈的操作有 init，empty，push，pop 和 top. 若某个操作需要有返回值，则用函数来定义，否则用过程来定义，但在构件的定义部分中只需要列出被移出的函数或过程的头部，规程的具体实现需在相应的实现部分中完成. stack 类的实现部分中，把所有规程定义为抽象规程，这是由于这些规程的实现方法随具体实现不同而不同. 例如，可以用链表或数组实现栈. stack 类是不考虑具体实现，只考虑栈的一般性质的抽象栈. 假设 stack 有两个子类，一个是用链表实现的无界栈 unboundedStack，另一个是用数组实现的有界栈 boundedStack. 程序 2 是使用 stack 类的一个模块 aModule 的描述. 模块 aModule 继承了系统已定义的模块 Module，它是所有模块的抽象. 由于模块 aModule 中将用到 stack 类，因此需要把 stack 放在移入表. proc1 是一个类属规程. 类属规程和抽象规程的实现体都用保留字 deferred 代替，但是组装目标软件时，必须用实际实现体来替换类属规程体，而抽象规程不需要这种替换. 在 proc2 过程规程中，aStack ! empty()等是消息发送.

规程的说明形式类似于 Pascal 语言中的函数和过程，但是由于 Recos 语言的赋值规则，用 Recos 编写的规程比其它传统过程式语言中的子程序更加灵活. 例如，我们可以用如下方法使用模块 aModule.

```
var
  a:unboundedStack;
  b:boundedStack;
  ...
  aModule! proc2(a);
  aModule! proc2(b);
```

虽然在规程 proc2 的定义中形式参数 aStack 的类型用 stack 来规定，但是由于赋值规则，调用 proc2 时实际参数类型可以是 stack 类的某个子类. 因此，我们可以把 proc2 看成是一种代码框架.

结束语：Recos 语言是一种支持面向对象设计的可重用构件描述语言，也是一种类型化的面向对象程序设计语言. 利用类属，抽象规程机制和赋值规则，可以编写出抽象级别较高的构件，而且通过类属机制可以把较抽象的构件调整为具体的构件. 本文中我们主要讨论了可重用构件的特点和说明方法，但是没有讨论可重用构件的检索问题和目标软件的组装问题. 我们已在 Sun 工作站上开发了基于 Recos 的软件重用支撑系统 SoftReuse. 该系统提供基于知识的构件查询工具，[1]系统库管理及软件组装工具，[2]Recos 到 C 语言的转换工具[3]等. 我们用 SoftReuse 系统开发了用于显示电网实时状态的图形工具和一个 Pascal 子语言的编译程序，它们分别运行在 DOS 和 UNIX.

参考文献

1　余江，全炳哲，金淳兆. 可重用构件查询专家系统 KCQ. 第四次全国软件工程会议论文集，北京，1991：118－122.

2　全炳哲，余江，金淳兆. 基于可重用构件的软件组装. 小型微型计算机系统，1991：**12(9)**：38－42.

3　姜延峰，全炳哲，金淳兆. 面向对象语言到过程式语言的转换. 小型微型计算机系统，1991：**12(10)**：20－25.

REUSABLE COMPONENTS AND ITS SPECIFICATION LANGUAGE

Quan Bingzhe, Yu Jiang and Jin Chunzhao

(*Department of Computer Science, Jilin University, Changchun* 130023)

Abstract This paper discusses the characteristics of the reusable components and introduces a new reusable component specification language (Recos). Recos supports object—orienteddesign and it provides function abstraction and generic mechanism.

Key words Reusability, reusable component, object—oriented, specification language.