[19] 中华人民共和国国家知识产权局

[51] Int. Cl.
G06F 9/52 (2006.01)

[12] 发明专利说明书

专利号 ZL 02159487.2

[45] 授权公告日 2006 年 7 月 12 日

[11] 授权公告号 CN 1264090C

[22] 申请日 2002.12.31 [21] 申请号 02159487.2

[71] 专利权人 上海科泰世纪科技有限公司

地址 201203 上海市浦东新区郭守敬路 498 号 17 号楼 2 层

[72] 发明人 陈 榕 叶忠强 梁宇洲

审查员 李 楠

[74] 专利代理机构 北京同立钧成知识产权代理有限公司

代理人 余 丽 刘 芳

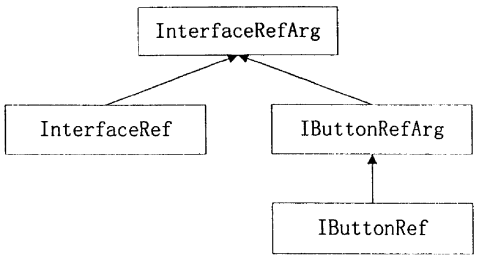
权利要求书 3 页 说明书 8 页 附图 2 页

[54] 发明名称

调用构件对象功能的智能指针的封装方法

[57] 摘要

一种调用构件对象功能的智能指针的封装方法，当接口智能指针做为构件对象方法的调用方传递给被调用方的参数时，不进行引用计数的增加和减少；当接口智能指针被构造、析构或赋值时，增加、减少引用计数；类智能指针包括与构件对象实现的接口相对应的成员变量，该成员变量用于调用构件对象实现的接口方法；类别智能指针包括与类别中所有接口相对应的成员变量，该成员变量用于调用继承了该类别的构件对象实现的该类别所有接口的接口方法。 本发明实现了智能管理引用计数和构件创建；通过类智能指针调用构件对象实现的所有接口功能，降低了编程的复杂度；类别智能指针类中的成员变量与接口相对应，实现了创建对象时的“多态”。



1、一种构件端接口智能指针的实现方法，其特征在于：

用一个类实现第一种类型接口智能指针，该第一种类型接口智能指针只做为构件对象的调用方，向该构件对象的被调用方接口功能传递参数时，不进行引用计数的增加或者减少；

用一个类实现第二种类型接口智能指针，该第二种类型接口智能指针用于构造、析构和赋值被调用方的构件对象接口功能时，还增加或者减少引用计数；

该第一、二种类型接口智能指针均用于创建构件对象。

2、根据权利要求 1 所述的构件端接口智能指针的实现方法，其特征在于：当构造一接口智能指针对象时，且当一个非空的接口指针从一个内存位置拷贝到另一个内存位置时，增加一个引用计数。

3、根据权利要求 1 所述的构件端接口智能指针的实现方法，其特征在于：当析构一接口智能指针对象，或当两个智能指针对象之间进行赋值时，对于已经包含非空接口指针的内存位置，在重写该内存位置之前，先减少一个引用计数。

4、一种直接访问构件对象所有功能的智能指针的封装方法，其特征在于：类智能指针设有与构件对象实现的接口相对应的成员变量，该成员变量用于调用构件对象实现的接口功能；当构造、析构和赋值被调用方的接口功能时，对所述的接口智能指针中的所述成员变量进行加操作或减操作，以进行引用计数；当向被调用方接口功能传递参数时，对所述接口智能指针的所述成员变量不做任何处理。

5、根据权要求 4 所述的直接访问构件对象所有功能的智能指针的封装方法，其特征在于：所述的成员变量至少为一个，且该成员变量的数量与构件对象实现的接口个数相等。

6、根据权要求 4 或 5 所述的直接访问构件对象所有功能的智能指针的封装方法，其特征在于：所述的类智能指针实现方式具体为：类智能指针通过继承接口智能指针实现。

7、根据权要求 4 所述的直接访问构件对象所有功能的智能指针的封装方法，其特征在于：当构造一接口智能指针对象时，且当一个非空的接口指针从一个内存位置拷贝到另一个内存位置时，增加一个引用计数。

8、根据权要求 4 所述的直接访问构件对象所有功能的智能指针的封装方法，其特征在于：当析构一接口智能指针对象，或当两个智能指针对象之间进行赋值时，对于已经包含非空接口指针的内存位置，在重写该内存位置之前，先减少一个引用计数。

9、根据权要求 4 或 5 所述的直接访问构件对象所有功能的智能指针的封装方法，其特征在于：所述的类智能指针实现方式具体为：类智能指针直接在该类中设置与构件对象实现接口一一对应的成员变量，类智能指针直接在该类中定义，并实现该构件对象实现的所有接口功能。

10、一种基于类别的智能指针的封装方法，其特征在于：类别智能指针设有与类别中所有接口相对应的成员变量，该成员变量用于调用继承了该类别的构件对象实现的该类别所有的接口功能；

当构造、析构和赋值被调用方的接口功能时，对所述的接口智能指针中的所述成员变量进行加操作或减操作，以进行引用计数；当向被调用方接口功能传递参数时，对所述接口智能指针的所述成员变量不做任何处理。

11、根据权利要求 10 所述的基于类别的智能指针的封装方法，其特征在于：所述的类别智能指针具有一个或以上的成员变量，该成员变量的数量与构件类中的构件对象实现的接口个数相等；该类别智能指针用于创建构件对象。

12、根据权利要求 10 或 11 所述的基于类别的智能指针的封装方法，其

特征在于：所述的类别智能指针实现方式具体为：类别智能指针通过继承接口智能指针实现。

13、根据权利要求 10 所述的基于类别的智能指针的封装方法，其特征在于：当构造一接口智能指针对象时，且当一个非空的接口指针从一个内存位置拷贝到另一个内存位置时，增加一个引用计数。

14、根据权利要求 10 所述的基于类别的智能指针的封装方法，其特征在于：当析构一接口智能指针对象，或当两个智能指针对象之间进行赋值时，对于已经包含非空接口指针的内存位置，在重写该内存位置之前，先减少一个引用计数。

15、根据权利要求 10 或 11 所述的基于类别的智能指针的封装方法，其特征在于：所述的类别智能指针实现方式具体为：类别智能指针包括与类别中所有接口相对应的成员变量，该成员变量用于调用继承了该类别的构件对象实现的该类别所有的接口功能。

调用构件对象功能的智能指针的封装方法

技术领域

本发明涉及一种调用构件对象功能的智能指针的封装方法，特别是指一种用于在创建构件对象时，用于传递调用方参数或调用构件对象接口功能的智能指针的封装方法，属于计算机技术领域。

背景技术

80年代以来，软件编程技术有了很大的发展，其发展可以大致分为以下几个阶段：

面向对象编程，即通过对软件模块的封装，使其相对独立，从而使复杂的问题简单化。面向对象编程强调的是对象的封装，但模块（对象）之间的关系在编译的时候已被固定，模块之间的关系是静态的，这种关系在程序运行时不能改变；也就是说：在运行时不能换用模块中更小的功能单元。

面向构件编程，即为了使不同软件开发商提供的构件模块（软件对象）可以相互操作使用，构件之间的连接和调用通过标准的协议来实现。构件化编程模型强调协议标准，需要提供各厂商都能遵守的协议体系。就像公制螺丝的标准一样，所有符合标准的螺丝和螺母都可以相互装配。构件化编程模型建立在面向对象技术的基础之上，是完全面向对象的，提供了动态构造部件模块（在运行中可以构造部件）的机制。构件在运行时可以动态装入，是可以更换的。但是，现有的面向构件编程技术要求用户自行定义构件的非自描述接口，使得用户程序的开发依然繁复。

智能指针用于封装指针功能。现有的一些智能指针类用于封装组件对象模型（The Component Object Model，简称 COM）接口指针，这些智能指针自动进行查询接口，和处理增、减引用计数构造或析构构件对象，并可以调用构件对象接口的功能。

例如：以C++程序设计语言为例：在COM编程中，如果一个对象CObject

实现了接口 IA、接口 IB 和接口 IC；接口 IA 中包括功能 FA，接口 IB 中包括功能 FB，接口 IC 中包括有功能 FC。这样，如果用户要调用功能 FA、FB、FC 时，需要写出下面的 C++代码才能调用到功能 FA、FB 和 FC，（假定用户已经获得接口 IA 的接口指针 pIA）：

```
.....
pIA→FA(...);
IB *pIB;
IC *pIC;
pIA→QueryInterface(IID_IB, &pIB);
pIA→QueryInterface(IID_IC, &pIC);
pIB→FB(...);
pIC→FC(...);
pIB→Release();
pIC→Release();
.....
```

为了简单，上述的 C++代码没有考虑功能调用失败的情况。即使如此，上述的调用需要用户编写九行程序，这给编程带来了一定的复杂度。

发明内容

本发明的主要目的在于提供一种构件端接口智能指针的实现方法，用于构件的调用方通过接口智能指针访问该接口所定义的功能，并实现智能管理引用计数和创建构件对象。

本发明的另一目的在于提供一种直接访问构件接口的智能指针的封装方法，通过类智能指针调用构件对象实现的所有接口功能，降低编程的复杂度。

本发明的又一目的在于提供一种基于类别的智能指针的封装方法，类别智能指针类中的成员变量，与具有相同特性的构件所组成的构件类中的所有

接口相对应，用于调用构件类对象实现的所有接口功能，降低编程的复杂度。

本发明的目的是这样实现的：

一种构件端接口智能指针的实现方法：

用一个类实现第一种类型接口智能指针，该第一种类型接口智能指针只做为构件对象的调用方向该构件对象的被调用方接口功能传递参数时，不进行引用计数的增加或者减少；

用一个类实现第二种类型接口智能指针，该第二种类型接口智能指针用于构造、析构和赋值被调用方的构件对象接口功能时，还增加或者减少引用计数；

该第一、二种类型接口智能指针均用于创建构件对象。

当构造一接口智能指针对象时，且当一个非空的接口指针从一个内存位置拷贝到另一个内存位置时，增加一个引用计数。

当析构一接口智能指针对象，或当两个智能指针对象之间进行赋值时，对于已经包含非空接口指针的内存位置，在重写该内存位置之前，先减少一个引用计数。

一种直接访问构件对象所有功能的智能指针的封装方法，类智能指针设有与构件对象实现的接口相对应的成员变量，该成员变量用于调用构件对象实现的接口功能；当构造、析构和赋值被调用方的接口功能时，对所述的接口智能指针中的所述成员变量进行加操作或减操作，以进行引用计数；当向被调用方接口功能传递参数时，对所述接口智能指针的所述成员变量不做任何处理。该成员变量至少为一个，且该成员变量的数量与构件对象实现的接口个数相等。类智能指针实现方式具体为：类智能指针通过继承接口智能指针实现。

所述的类智能指针实现方式具体为：类智能指针直接在该类中设置与构件对象实现接口一一对应的成员变量，类智能指针直接在该类中定义，并实现该构件对象实现的所有接口功能。

一种基于类别的智能指针的封装方法，类别智能指针包括与类别中所有接口相对应的成员变量，该成员变量用于调用继承了该类别的构件对象实现的该类别所有的接口功能；当构造、析构和赋值被调用方的接口功能时，对所述的接口智能指针中的所述成员变量进行加操作或减操作，以进行引用计数；当向被调用方接口功能传递参数时，对所述接口智能指针的所述成员变量不做任何处理。

所述的类别智能指针具有一个或以上的成员变量，该成员变量的数量与构件类中的构件对象实现的接口个数相等；该类别智能指针用于创建构件对象。

所述的类别智能指针实现方式具体为：类别智能指针通过继承接口智能指针实现。

所述的类别智能指针实现方式具体为：类别智能指针包括与类别中所有接口相对应的成员变量，该成员变量用于调用继承了该类别的构件对象实现的该类别所有的接口功能。

本发明用于构件的调用方通过接口智能指针访问该接口所定义的功能，实现了智能管理引用计数；通过类智能指针调用构件对象实现的所有接口功能，降低了编程的复杂度；类别智能指针类中的成员变量与具有相同特性构件所组成的构件类的所有接口相对应，同时方便了对构件对象根据需要进行灵活的配置。

附图说明

图 1 为本发明接口智能指针中成员变量与构件对象实现的接口之间的关系示意图；

图 2 为本发明接口智能指针通过成员变量调用构件对象实现对应功能的示意图；

图 3 为本发明不同类型接口智能指针相互之间继承关系的示意图；

图 4 为本发明类智能指针继承接口智能指针的继承关系示意图;

图 5 为本发明类智能指针中的各成员变量与构件对象实现的各接口之间的关系示意图。

具体实施方式

以下结合附图和具体的实施例对本发明作进一步的详细说明:

在基于构件对象的平台中，用户可通过接口智能指针来访问该这个接口所定义的方法，因此，接口智能指针是对接口的封装。本发明对接口智能指针的实现以 C++程序设计语言为例，具体如下:

将接口智能指针定义为类，这个类中只设有一个成员变量，该成员变量就是实际的指向对象接口的接口指针。

参见图 1 和图 2，其中接口智能指针 IARef 与接口 IA 对应。接口智能指针类实现该接口的所有方法 这些接口方法都通过成员变量调用构件对象实现的方法来实现。

本实施例中，当一个接口智能指针对象被构造时，当一个非空的接口指针从一个内存位置拷贝到另一个内存位置时，则应调用引用增加（AddRef）功能，以便通知对象又有附加的引用发生了;

当一个接口智能指针对象被析构或两个智能指针对象进行赋值时，则对于已经包含非空接口指针的内存位置来说，在重写该内存位置之前，必须先调用释放（Release）功能，以便通知构件对象“该引用已经被销毁”;

当接口智能指针做为入口（“[in]”）接口参数时，则上述两种情况下的 AddRef 功能和 Release 功能的调用可以被优化掉。

具体而言，在本实施例中实现了两种类型的智能指针，其一只能做为[in]参数使用，而另外一种则继承前一种智能指针，对于构造，析构和赋值操作则进行相应的增加和减少引用计数。

以 C++程序设计语言为例，如果接口智能指针 IXXXRefArg 实现为[in]

接口参数，而作为接口智能指针 IXXXRefArg 的继承，接口智能指针 IXXXRef 中的构造、析构和赋值则进行相应的增加和减少引用计数。

对于 IUnknown 接口（ Iunknown 为美国微软公司 COM 规范中的一种接口）的两个接口智能指针类，接口智能指针类 InterfaceRefArg 和接口智能指针类 InterfaceRef，其中，接口智能指针类 InterfaceRef 继承了接口智能指针类 InterfaceRefArg。其中，接口智能指针类 InterfaceRefArg 中定义了一个成员变量 IUnknown * m_pIface。其它所有接口智能指针追根溯源都继承于该类，因此所有接口智能指针都具有该成员变量 IUnknown * m_pIface。

正如所有的接口都继承于 IUnknown 一样，所有的接口智能指针都继承于 InterfaceRefArg。参见图 3，以接口 IButton 为例：接口智能指针类 IbuttonRef 继承于接口智能指针类 IbuttonRefArg，接口智能指针类 IbuttonRefArg 和 InterfaceRef 均继承于接口智能指针类 InterfaceRefArg。

通过接口智能指针可以创建出实现了该接口的构件对象，并使该智能指针的成员变量指向这个新创建出的构件对象。

为了解决在编程中代码复杂度不能降低的问题，本发明采用类智能指针对构件类封装，假定已经有了指向上述的的 CObject 对象的类智能指针变量 m_cObject。则调用上述三个功能 FA、FB 和 FC 的代码为：

```
m_cObject.FA(...);
m_cObject.FB(...);
m_cObject.FC(...);
```

由此可以看出：使用类智能指针，代码简单且程序易懂。

在以 C++语言为例：类智能指针表现为类，这个类具有若干个成员变量。每个成员变量用来指向构件对象的一个接口，成员变量的数目等于构件对象

实现的接口个数。成员变量和构件对象实现的接口一一对应。通过类智能指针，可以调用构件对象实现的所有接口功能。如上所述，通过类智能指针 m_cObject，就可以调用接口 IA 的接口功能 FA，也可以调用接口 IB 的接口功能 FB，还可以调用接口 IC 的接口功能 FC。

本发明中，类智能指针的实现可以有两种方式:

参见图 4，以 C++语言为例，类智能指针的第一种实现方式为：类智能指针直接继承接口智能指针；在这种情况下，当用户调用构件对象实现的功能 CObjectRef::FA(...)时，事实上它调用的就是智能指针 IARef 对应的接口智能指针 IARef::FA(...)。

参见图 5，类智能指针的第二种实现方式为：构件对象的类智能指针不继承构件对象实现的所有接口对应的接口智能指针，而是直接在该类中定义若干个成员变量，每个成员变量对应一个该构件对象实现的接口，成员变量的个数等于构件对象实现的接口数目。类智能指针直接在该类中定义并实现该构件对象实现的所有接口功能。

在这种情况下，当用户调用构件对象实现的功能 CObjectRef::FA(...)时，构件对象实现的功能 CObjectRef::FA(...)的实现将调用成员变量 m_pIAface→FA(...)。

通过类智能指针可以创建出这个类智能指针对应的构件对象，并且使这个类智能指针继承来的成员变量指向该新创建出的对象。

为了解决在编程中代码复杂度不能降低的问题，同时方便对构件对象根据需要进行灵活的配置，本发明采用类别智能指针对构件类封装。

就像接口智能指针是对接口的封装一样，类别智能指针是对类别的封装。本发明将具有相同特点的构件对象接口划分成一类。例如：各种型号的声卡均用于记录和播放声音，各种型号的声卡都具有相应的驱动程序，因此可以把每种型号声卡的驱动程序做成一个声卡构件类；一个构件类中包括若

干个接口，在该声卡构件类中，所有的声卡驱动构件均包含有一个相同的接口集合。一个类别为一个接口的集合，用于被构件类和类别进行继承。所有继承这个类别的构件类，都实现这个类别包括的所有接口。而继承这个类别的类别，它的接口集合就变为两个接口集合的并集。

本发明中，类别和构件类的区别是：类别是一个接口的集合，用于被继承；构件类也是一个接口的集合，构件类却不能被继承；所有继承类别的构件类都必须实现该类别包括的所有接口。

本发明中类别智能指针的实现和类智能指针相同，即类别智能指针采用与类智能指针相同的方式实现，在此不再赘述。

最后应说明的是：以上实施例仅用以说明本发明而并非限制本发明所描述的技术方案；因此，尽管本说明书参照上述的各个实施例对本发明已进行了详细的说明，但是，本领域的普通技术人员应当理解，仍然可以对本发明进行修改或者等同替换；而一切不脱离本发明的精神和范围的技术方案及其改进，其均应涵盖在本发明的权利要求范围当中。

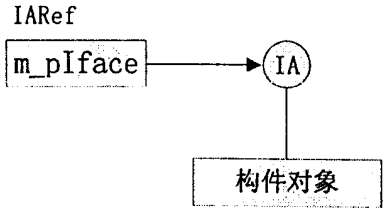

IARef
m_pIface
IA
构件对象

图 1

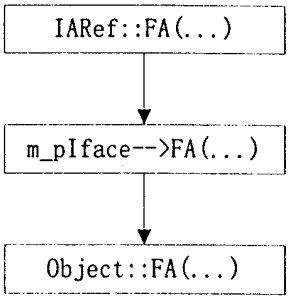

IARef::FA(...)
m_pIface-->FA(...)
Object::FA(...)

图 2

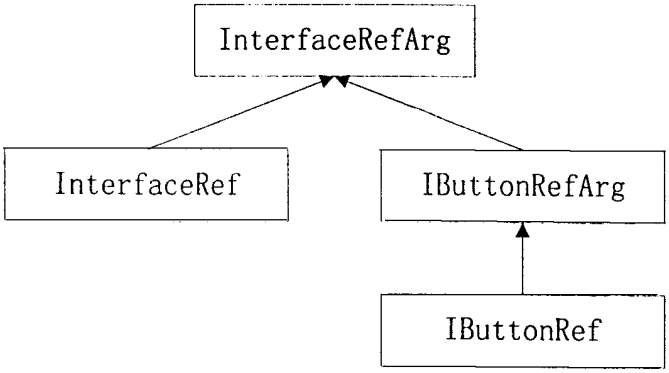

InterfaceRefArg
InterfaceRef
IButtonRefArg
IButtonRef

图 3

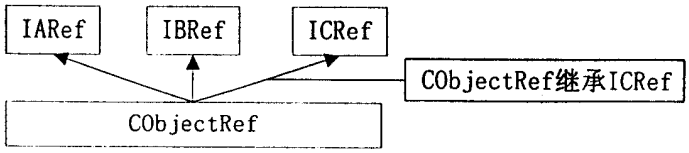

IARef
IBRef
ICRef
CObjectRef继承ICRef
CObjectRef

图 4

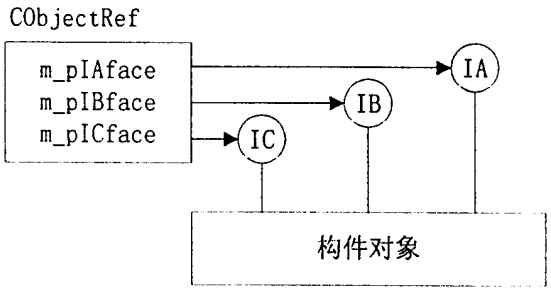

CObjectRef
m_pIAface
m_pIBface
m_pICface
IA
IB
IC
构件对象

图 5