

附录A 词汇表

抽象类 (abstract class) 一种主要用来定义接口的类。抽象类中的部分或全部操作被延迟到其子类中实现。抽象类不能实例化。

抽象耦合 (abstract coupling) 若类A维护一个指向抽象类B的引用,则称类A抽象耦合于B。我们之所以称之为抽象耦合乃是因为 A指向的是一个对象的类型,而不是一个具体对象。

抽象操作 (abstract operation) 一种声明了型构 (signature) 而没有实现的操作。在 C++中,抽象操作对应于纯虚成员函数。

相识关系 (acquaintance relationship) 如果一个类指向另一个类,则这两个类之间有相识关系。

聚合对象 (aggregate object) 一种包含子对象的对象。这些子对象称为聚合对象的部分,而聚合对象对它们负责。

聚合关系 (aggregation relationship) 聚合对象与其部分之间的关系。类为其对象 (例如,聚合对象) 定义这种关系。

黑箱复用 (black-box reuse) 一种基于对象组合的复用方式。这些被组合的对象之间并不开放各自的内部细节,因此被比作“黑箱”。

类 (class) 类定义对象的接口和实现。它规定对象的内部表示,定义对象可实施的操作。

类图 (class diagram) 类图描述类及其内部结构和操作,以及类间的静态关系。

类操作 (class operation) 以类而不是单独的对象为目标的操作。在 C++中,类操作称为静态成员函数。

具体类 (concrete class) 不含抽象操作的类。它可以实例化。

构造器 (constructor) 在C++中,一种系统自动调用的用来初始化新对象实例的操作。

耦合 (coupling) 软件构件之间相互依赖的程度。

委托 (delegation) 一种实现机制,即一个对象把发给它的请求转发/委托给另一个对象。而受托对象将代表原对象执行请求的操作。

设计模式 (design pattern) 设计模式针对面相对象系统中重复出现的设计问题,提出一个通用的设计方案,并予以系统化的命名和动机解释。它描述了问题、解决方案、在什么条件下使用该解决方案及其效果。它还给出了实现要点和实例。该解决方案是解决该问题的一组精心安排的通用的类和对象,再经定制和实现就可用来解决特定上下文中的问题。

析构器 (destructor) 在C++中,一种系统自动调用的用来清理 (finalize) 即将被删除的对象的对象的操作。

动态绑定 (dynamic binding) 在运行时刻才将一个请求与一个对象及其一个操作关联起来。在C++中,只有虚函数可动态绑定。

封装 (encapsulation) 其结果是将对象的表示和实现隐藏起来。在对象之外,看不到其

内部表示，也不能直接对其进行访问。操作（operation）是访问和修改对象表示的唯一途径。

框架（framework） 一组相互协作的类，形成某类软件的一个可复用设计。框架将设计划分为一组抽象类，并定义它们各自的责任及相互之间的合作，以此来指导体系结构级的设计。开发者通过继承框架中的类和组合其实例来定制该框架以生成特定的应用。

友类（friend class） 在C++中，A为B的友类是指A对B中的操作和数据有与B本身一样的访问权限。

继承（inheritance） 两个实体间的一种关系，其中一实体乃是基于另一实体而定义的。类继承以一个或多个父类为基础定义一个新类，这个新类继承了其父类的接口和实现，被称为子类（C++）或派生类。类继承包含了接口继承和实现继承。接口继承以一个或多个已有接口为基础定义新的接口；实现继承以一个或多个已有实现为基础定义新的实现。

实例变量（instance variable） 定义部分对象表示的数据。C++中使用的术语是数据成员。

交互图（interaction diagram） 展示对象间请求流程的一种示意图。

接口（interface） 一个对象所有操作定义的类型集合。接口刻画了一个对象可响应的请求的集合。

元类（metaclass） 在Smalltalk中，类也是对象。元类是类对象的类。

混入类（mixin class） 一种被设计为通过继承与其他类结合的类。混入类通常是抽象类。

对象（object） 一个封装了数据及作用于这些数据的操作的运行实体。

对象组合（object composition） 组装和组合一组对象以获得更复杂的行为。

对象图（object diagram） 描述运行时刻特定对象结构的示意图。

对象引用（object reference） 用于标识另一对象的一个值。

操作（operation） 对象的数据仅能由其自身的操作来存取。对象受到请求时执行操作。在C++中，操作称为成员函数，而Smalltalk使用术语“方法”。

重定义（overriding） 在一个子类中重定义（从父类继承下来的）操作。

参数化类型（parameterized type） 一种含有未确定成分类型的类型。在使用时，将未确定类型处理成参数。在C++中，参数化类型称为模板（template）。

父类（parent class） 被其他类继承的类。Smalltalk又称之为超类（superclass），C++中又称之为基类（base class），有时又称为祖先类（ancestor class）。

多态（polymorphism） 在运行时刻接口匹配的对象能互相替换的能力。

私有继承（private inheritance） 在C++中，一种仅出于实现目的的继承。

协议（protocol） 接口概念的扩展，包含指明可允许的请求序列。

接收者（receiver） 一个请求的目标对象。

请求（request） 一个对象当受到其他对象的请求时执行相应的操作。通常请求又称为消息。

型构（signature） 一个操作的型构定义了它的名称、参数和返回值。

子类（subclass） 继承了另一个类的类。在C++中，子类又称为派生类（derived class）。

子系统（subsystem） 一组相互协作的类形成的一个相对独立的部分，完成一定的功能。

子类型（subtype） 如果一个类型的接口包含另一类型的接口，则前一类型称为后一类

型的子类型。

超类型（supertype） 为其他类型继承的父类型。

工具箱（toolkit） 一组提供实用功能的类，但它们并不包含任何具体应用的设计。

类型（type） 一个特定接口的名称。

白箱复用（white-box reuse） 一种基于类继承的复用。子类复用父类的接口和实现，但它也可能存取其父类的其他私有部分。