面向对象 锁定

本词条由"科普中国"百科科学词条编写与应用工作项目 审核 。

面向对象(Object Oriented,OO)是软件开发方法。面向对象的概念和应用已超越了程序设计和软件开发，扩展到如数据库系统、交互式界面、应用结构、应用平台、分布式系统、网络管理结构、CAD技术、人工智能等领域。面向对象是一种对现实世界理解和抽象的方法，是计算机编程技术[1]发展到一定阶段后的产物。

中文名	面向对象	作　用	软件开发
外文名	Object Oriented,OO	使用领域	CAD技术、人工智能等

目录

早期发展

早期的计算机编程是基于面向过程的方法，例如实现算术运算1+1+2 = 4，通过设计一个算法就可以解决当时的问题。随着计算机技术的不断提高，计算机被用于解决越来越复杂的问题。一切事物皆对象，通过面向对象的方式，将现实世界的事物抽象成对象，现实世界中的关系抽象成类、继承，帮助人们实现对现实世界的抽象与数字建模。通过面向对象的方法，更利于用人理解的方式对复杂系统进行分析、设计与编程。同时，面向对象能有效提高编程的效率，通过封装技术，消息机制可以像搭积木的一样快速开发出一个全新的系统。面向对象是指一种程序设计范型，同时也是一种程序开发的方法。对象指的是类的集合。它将对象作为程序的基本单元，将程序和数据封装其中，以提高软件的重用性、灵活性和扩展性。[2]

起初，"面向对象"是专指在程序设计中采用封装、继承、多态等设计方法。[2]

面向对象的思想已经涉及到软件开发的各个方面。如，面向对象的分析（OOA，Object Oriented Analysis），面向对象的设计（OOD，Object Oriented Design）、以及我们经常说的面向对象的编程实现（OOP，Object Oriented Programming）。[2]

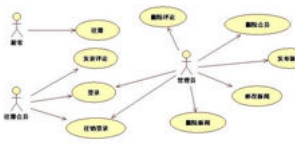
面向对象的分析根据抽象关键的问题域来分解系统。面向对象的设计是一种提供符号设计系统的面向对象的实现过程，它用非常接近实际领域术语的方法把系统构造成"现实世界"的对象。面向对象程序设计可以看作一种在程序中包含各种独立而又互相调用的对象的思想，这与传统的思想刚好相反：传统的程序设计主张将程序看作一系列函数的集合，或者直接就是一系列对电脑下达的指令。面向对象程序设计中的每一个对象都应该能够接受数据、处理数据并将数据传达给其它对象，因此它们都可以被看作一个小型的"机器"，即对象。[2]

背景

面向对象是在结构化设计方法出现很多问题的情况下应运而生的。结构化设计方法求解问题的基本策略是从功能的角度审视问题域。它将应用程序看成实现某些特定任务的功能模块，其中子过程是实现某项具体操作的底层功能模块。在每个功能模块中，用数据结构描述待处理数据的组织形式，用算法描述具体的操作过程。面对日趋复杂的应用系统，这种开发思路在下面几个方面逐渐暴露了一些弱点。[2]

1.审视问题域的视角

在现实世界中存在的客体是问题域中的主角，所谓客体是指客观存在的对象实体和主观抽象的概念，他是人类观察问题和解决问题的主要目标。例如，对于一个学校学生管理系统来说，无论是简单还是复杂，始终是围绕学生和老师这两个客体实施。在自然界，每个客体都具有一些属性和行为，例如学生有学号、姓名、性别等属性，以及上课、考试、做实验等行为。因此，每个个体都可以用属性和行为来描述。[2]



面向对象图册

之间相互驱动、相互作用，最终使每个客体按照设计者的意愿改变其属性状态。[2]

结构化设计方法所采用的设计思路不是将客体作为一个整体，而是将依附于客体之上的行为抽取出来，以功能为目标来设计构造应用系统。这种做法导致在进行程序设计的时候，不得不将客体所构成的现实世界映射到由功能模块组成的解空间中，这种变换过程，不仅增加了程序设计的复杂程度，而且背离了人们观察问题和解决问题的基本思路。另外，再仔细思考会发现，在任何一个问题域中，客体是稳定的，而行为是不稳定的。例如，不管是国家图书馆，还是学校图书馆，还是国际图书馆，都会含有图书这个客体，但管理图书的方法可能是截然不同的。结构化设计方法将审视问题的视角定位于不稳定的操作之上，并将描述客体的属性和行为分开，使得应用程序的日后维护和扩展相当困难，甚至一个微小的变动，都会波及到整个系统。面对问题规模的日趋扩大、环境的日趋复杂、需求变化的日趋加快，将利用计算机解决问题的基本方法统一到人类解决问题的习惯方法之上，彻底改变软件设计方法与人类解决问题的常规方式扭曲的现象迫在眉睫，这是提出面向对象的首要原因。[2]

2.抽象级别

抽象是人类解决问题的基本法宝。良好的抽象策略可以控制问题的复杂程度，增强系统的通用性和可扩展性。抽象主要包括过程抽象和数据抽象。结构化设计方法应用的是过程抽象。所谓过程抽象是将问题域中具有明确功能定义的操作抽取出来，并将其作为一个实体看待。这种抽象级别对于软件系统结构的设计显得有些武断，并且稳定性差，导致很难准确无误地设计出系统的每一个操作环节。一旦某个客体属性的表示方式发生了变化，就有可能牵扯到已有系统的很多部分。而数据抽象是较过程抽象更高级别的抽象方式，将描述客体的属性和行为绑定在一起，实现统一的抽象，从而达到对现实世界客体的真正模拟。[2]

3.封装体

封装是指将现实世界中存在的某个客体的属性与行为绑定在一起，并放置在一个逻辑单元内。该逻辑单元负责将所描述的属性隐藏起来，外界对客体内部属性的所有访问只能通过提供的用户接口实现。这样做既可以实现对客体属性的保护作用，又可以提高软件系统的可维护性。只要用户接口不改变，任何封装体内部的改变都不会对软件系统的其他部分造成影响。结构化设计方法没有做到客体的整体封装，只是封装了各个功能模块，而每个功能模块可以随意地对没有保护能力客体属性实施操作，并且由于描述属性的数据与行为被分割开来，所以一旦某个客体属性的表达方式发生了变化，或某个行为效果发生了改变，就有可能对整个系统产生影响。[2]

4.可重用性

可重用性标识着软件产品的可复用能力，是衡量一个软件产品成功与否的重要标志。当今的软件开发行业，人们越来越追求开发更多的、更有通用性的可重用构件，从而使软件开发过程彻底改善，即从过去的语句级编写发展到现在的构件组装，从而提高软件开发效率，推动应用领域迅速扩展。然而，结构化程序设计方法的基本单位是模块，每个模块只是实现特定功能的过程描述，因此，它的可重用单位只能是模块。例如，在C语言编写程序时使用大量的标准函数。但对于今天的软件开发来说，这样的重用力度显得微不足道，而且当参与操作的某些数据类型发生变化时，就不能够再使用那些函数了。因此，渴望更大力度的可重用构件是如今应用领域对软件开发提出的新需求。[2]

上述弱点驱使人们寻求一种新的程序设计方法，以适应现代社会对软件开发的更高要求，面向对象由此产生。[2]

概念

⑴对象。

对象是人们要进行研究的任何事物，从最简单的整数到复杂的飞机等均可看作对象，它不仅能表示具体的事物，还能表示抽象的规则、计划或事件。[2]

⑵对象的状态和行为。

对象具有状态，一个对象用数据值来描述它的状态。

对象还有操作，用于改变对象的状态，对象及其操作就是对象的行为。

对象实现了数据和操作的结合，使数据和操作封装于对象的统一体中[2]

⑶类。

具有相同特性（数据元素）和行为（功能）的对象的抽象就是类。因此，对象的抽象是类，类的具体化就是对象，也可以说类的实例是对象，类实际上就是一种数据类型。

类具有属性，它是对象的状态的抽象，用数据结构来描述类的属性。

类具有操作，它是对象的行为的抽象，用操作名和实现该操作的方法来描述。[2]

⑷类的结构。

在客观世界中有若干类，这些类之间有一定的结构关系。通常有两种主要的结构关系，即一般--具体结构关系，整体--部分结构关系。

①一般--具体结构称为分类结构，也可以说是“或”关系，或者是“is a”关系。

②整体--部分结构称为组装结构，它们之间的关系是一种“与”关系，或者是“has a”关系。[2]

信息。发送一条消息至少要包括说明接受消息的对象名、发送给该对象的消息名（即对象名、方法名）。一般还要对参数加以说明，参数可以是认识该消息的对象所知道的变量名，或者是所有对象都知道的全局变量名。 [2]

类中操作的实现过程叫做方法，一个方法有方法名、返回值、参数、方法体。

特征

(1)对象唯一性。

每个对象都有自身唯一的标识，通过这种标识，可找到相应的对象。在对象的整个生命期中，它的标识都不改变，不同的对象不能有相同的标识。 [2]

(2)抽象性。

抽象性是指将具有一致的数据结构（属性）和行为（操作）的对象抽象成类。一个类就是这样一种抽象，它反映了与应用有关的重要性质，而忽略其他一些无关内容。任何类的划分都是主观的，但必须与具体的应用有关。 [2]

(3)继承性。

继承性是子类自动共享父类数据结构和方法的机制，这是类之间的一种关系。在定义和实现一个类的时候，可以在一个已经存在的类的基础之上来进行，把这个已经存在的类所定义的内容作为自己的内容，并加入若干新的内容。 [2]

继承性是面向对象程序设计语言不同于其它语言的最重要的特点，是其他语言所没有的。

在类层次中，子类只继承一个父类的数据结构和方法，则称为单重继承。

在类层次中，子类继承了多个父类的数据结构和方法，则称为多重继承。

多重继承，JAVA、VB、NET、Objective-C均仅支持单继承，注意在C++多重继承时，需小心二义性。

在软件开发中，类的继承性使所建立的软件具有开放性、可扩充性，这是信息组织与分类的行之有效的方法，它简化了对象、类的创建工作量，增加了代码的可重用性。

采用继承性，提供了类的规范的等级结构。通过类的继承关系，使公共的特性能够共享，提高了软件的重用性。 [2]

(4)多态性（多形性）

多态性是指相同的操作或函数、过程可作用于多种类型的对象上并获得不同的结果。不同的对象，收到同一消息可以产生不同的结果，这种现象称为多态性。

多态性允许每个对象以适合自身的方式去响应共同的消息。

多态性增强了软件的灵活性和重用性。 [2]

要素

(1)抽象。

抽象是指强调实体的本质、内在的属性。在系统开发中，抽象指的是在决定如何实现对象之前的对象的意义和行为。使用抽象可以尽可能避免过早考虑一些细节。

类实现了对象的数据（即状态）和行为的抽象。 [2]

(2)封装性（信息隐藏）。

封装性是保证软件部件具有优良的模块性的基础。

面向对象的类是封装良好的模块，类定义将其说明（用户可见的外部接口）与实现（用户不可见的内部实现）显式地分开，其内部实现按其具体定义的作用域提供保护。

对象是封装的最基本单位。封装防止了程序相互依赖性而带来的变动影响。面向对象的封装比传统语言的封装更为清晰、更为有力。 [2]

(3)共享性

面向对象技术在不同级别上促进了共享

同一类中的共享。同一类中的对象有着相同数据结构。这些对象之间是结构、行为特征的共享关系。

在同一应用中共享。在同一应用的类层次结构中，存在继承关系的各相似子类中，存在数据结构和行为的继承，使各相似子类共享共同的结构和行为。使用继承来实现代码的共享，这也是面向对象的主要优点之一。

4.强调对象结构而不是程序结构

开发方法

面向对象开发方法的研究已日趋成熟，国际上已有不少面向对象产品出现。面向对象开发方法有Coad方法、Booch方法和OMT方法等。[2]

1.Booch方法

Booch最先描述了面向对象的软件开发方法的基础问题，指出面向对象开发是一种根本不同于传统的功能分解的设计方法。面向对象的软件分解更接近人对客观事务的理解，而功能分解只通过问题空间的转换来获得。[2]

2.Coad方法

Coad方法是1989年Coad和Yourdon提出的面向对象开发方法。该方法的主要优点是通过多年来大系统开发的经验与面向对象概念的有机结合，在对象、结构、属性和操作的认定方面，提出了一套系统的原则。该方法完成了从需求角度进一步进行类和类层次结构的认定。尽管Coad方法没有引入类和类层次结构的术语，但事实上已经在分类结构、属性、操作、消息关联等概念中体现了类和类层次结构的特征。[2]

3.OMT方法

OMT方法是1991年由James Rumbaugh等5人提出来的，其经典著作为“面向对象的建模与设计”。

该方法是一种新兴的面向对象的开发方法，开发工作的基础是对真实世界的对象建模，然后围绕这些对象使用分析模型来进行独立于语言的设计，面向对象的建模和设计促进了对需求的理解，有利于开发得更清晰、更容易维护的软件系统。该方法为大多数应用领域的软件开发提供了一种实际的、高效的保证，努力寻求一种问题求解的实际方法。[2]

4.UML(Unified Modeling Language）语言

软件工程领域在1995年～1997年取得了前所未有的进展，其成果超过软件工程领域过去15年的成就总和，其中最重要的成果之一就是统一建模语言（UML）的出现。UML将是面向对象技术领域内占主导地位的标准建模语言。[2]

UML不仅统一了Booch方法、OMT方法、OOSE方法的表示方法，而且对其作了进一步的发展，最终统一为大众接受的标准建模语言。UML是一种定义良好、易于表达、功能强大且普遍适用的建模语言。它融入了软件工程领域的新思想、新方法和新技术。它的作用域不限于支持面向对象的分析与设计，还支持从需求分析开始的软件开发全过程。[2]

模型

对象模型

对象模型表示了静态的、结构化的系统数据性质，描述了系统的静态结构，它是从客观世界实体的对象关系角度来描述，表现了对象的相互关系。该模型主要关心系统中对象的结构、属性和操作，它是分析阶段三个模型的核心，是其他两个模型的框架。[2]

1.对象和类

(1)对象。

对象建模的目的就是描述对象。

(2) 类。

通过将对象抽象成类，我们可以使问题抽象化，抽象增强了模型的归纳能力。

(3) 属性。

属性指的是类中对象所具有的性质（数据值）。

(4) 操作和方法。

操作是类中对象所使用的一种功能或变换。类中的各对象可以共享操作，每个操作都有一个目标对象作为其隐含参数。

方法是类的操作的实现步骤。[2]

2.关联和链

关联是建立类之间关系的一种手段，而链则是建立对象之间关系的一种手段。

(1) 关联和链的含义。

链表示对象间的物理与概念联结，关联表示类之间的一种关系，链是关联的实例，关联是链的抽象。

(3) 受限关联。

受限关联由两个类及一个限定词组成，限定词是一种特定的属性，用来有效的减少关联的重数，限定词在关联的终端对象集中说明。

限定提高了语义的精确性，增强了查询能力，在现实世界中，常常出现限定词。

(4) 关联的多重性。

关联的多重性是指类中有多少个对象与关联的类的一个对象相关。重数常描述为“一”或“多”。 [2]

3.类的层次结构

(1) 聚集关系。

聚集是一种“整体－部分”关系。在这种关系中，有整体类和部分类之分。聚集最重要的性质是传递性，也具有逆对称性。

聚集可以有不同层次，可以把不同分类聚集起来得到一颗简单的聚集树，聚集树是一种简单表示，比画很多线来将部分类联系起来简单得多，对象模型应该容易地反映各级层次。

(2)一般化关系。

一般化关系是在保留对象差异的同时共享对象相似性的一种高度抽象方式。它是“一般---具体”的关系。一般化类称为你类，具体类又能称为子类，各子类继承了父类的性质，而各子类的一些共同性质和操作又归纳到你类中。因此，一般化关系和继承是同时存在的。一般化关系的符号表示是在类关联的连线上加一个小三角形。

4.对象模型

(1)模板。模板是类、关联、一般化结构的逻辑组成。

(2)对象模型。

对象模型是由一个或若干个模板组成。模板将模型分为若干个便于管理的子块，在整个对象模型和类及关联的构造块之间，模板提供了一种集成的中间单元，模板中的类名及关联名是唯一的。 [2]

动态模型

动态模型是与时间和变化有关的系统性质。该模型描述了系统的控制结构，它表示了瞬间的、行为化的系统控制 [2]

性质，它关心的是系统的控制，操作的执行顺序，它表示从对象的事件和状态的角度出发，表现了对象的相互行为。 [2]

该模型描述的系统属性是触发事件、事件序列、状态、事件与状态的组织。使用状态图作为描述工具。它涉及到事件、状态、操作等重要概念。 [2]

1.事件

事件是指定时刻发生的某件事。 [2]

2.状态

状态是对象属性值的抽象。对象的属性值按照影响对象显著行为的性质将其归并到一个状态中去。状态指明了对象对输入事件的响应。 [2]

3.状态图

状态图是一个标准的计算机概念，他是有限自动机的图形表示，这里把状态图作为建立动态模型的图形工具。 [2]

状态图反映了状态与事件的关系。当接收一事件时，下一状态就取决于当前状态和所接收的该事件，由该事件引起的状态变化称为转换。 [2]

状态图是一种图，用结点表示状态，结点用圆圈表示；圆圈内有状态名，用箭头连线表示状态的转换，上面标记事件名，箭头方向表示转换的方向。 [2]

功能模型

功能模型描述了系统的所有计算。功能模型指出发生了什么，动态模型确定什么时候发生，而对象模型确定发生的客体。功能模型表明一个计算如何从输入值得到输出值，它不考虑计算的次序。功能模型由多张数据流图组成。数据流图用来表示从源对象到目标对象的数据值的流向，它不包含控制信息，控制信息在动态模型中表示，同时数据流图也不表示对象中值的组织，值的组织在对象模型中表示。 [2]

数据流图中包含有处理、数据流、动作对象和数据存储对象。

1.处理

数据流图中的数据流将对象的输出与处理、处理与对象的输入、处理与处理联系起来。在一个计算机中，用数据流来表示一中间数据值，数据流不能改变数据值。[2]

3.动作对象

动作对象是一种主动对象，它通过生成或者使用数据值来驱动数据流图。[2]

4.数据存储对象

数据流图中的数据存储是被动对象，它用来存储数据。它与动作对象不一样，数据存储本身不产生任何操作，它只响应存储和访问的要求。[2]

实例分析

面向对象分析的目的是对客观世界的系统进行建模。本节以上面介绍的模型概念为基础，结合“银行网络系统”的具体实例来构造客观世界问题的准确、严密的分析模型。[2]

分析模型有三种用途：用来明确问题需求；为用户和开发人员提供明确需求；为用户和开发人员提供一个协商的基础，作为后继的设计和实现的框架。[2]

（一） 面向对象的分析

系统分析的第一步是：陈述需求。分析者必须同用户一块工作来提炼需求，因为这样才表示了用户的真实意图，其中涉及对需求的分析及查找丢失的信息。下面以“银行网络系统”为例，用面向对象方法进行分析。

银行网络系统问题陈述：设计支持银行网络的软件，银行网络包括人工出纳站和分行共享的自动出纳机。每个分理处用分理处计算机来保存各自的帐户，处理各自的事务；各自分理处的出纳站与分理处计算机通信，出纳站录入帐户和事务数据；自动出纳机与分行计算机通信，分行计算机与拨款分理处结帐，自动出纳机与用户接口接受现金卡，与分行计算机通信完成事务，发放现金，打印收据；系统需要记录保管和安全措施；系统必须正确处理同一帐户的并发访问；每个分处理为自己的计算机准备软件，银行网络费用根据顾客和现金卡的数目分摊给各分理处。[2]

（二）建立对象模型

首先标识和关联，因为它们影响了整体结构和解决问题的方法，其次是增加属性，进一步描述类和关联的基本网络，使用继承合并和组织类，最后操作增加到类中去作为构造动态模型和功能模型的副产品。[2]

1.确定类

构造对象模型的第一步是标出来自问题域的相关的对象类，对象包括物理实体和概念。所有类在应用中都必须有意义，在问题陈述中，并非所有类都是明显给出的。有些是隐含在问题域或一般知识中的。[2]

按图10-19所示的过程确定类

查找问题陈述中的所有名词，产生如下的暂定类。

软件 银行网络 出纳员 自动出纳机 分行

分处理 分处理计算机 帐户事务出纳站

事务数据 分行计算机 现金卡 用户 现金

收据 系统 顾客 费用 帐户数据

访问 安全措施 记录保管

根据下列标准，去掉不必要的类和不正确的类。[2]

⑴ 冗余类：若两个类表述了同一个信息，保留最富有描述能力的类。如"用户"和"顾客"就是重复的描述，因为"顾客"最富有描述性，因此保留它。

⑵ 不相干的类：除掉与问题没有关系或根本无关的类。例如，摊派费用超出了银行网络的范围。

⑶ 模糊类：类必须是确定的，有些暂定类边界定义模糊或范围太广，如"记录保管"就模糊类，它是"事务"中的一部分。

⑷ 属性：某些名词描述的是其他对象的属性，则从暂定类中删除。如果某一性质的独立性很重要，就应该把他归属到类，而不把它作为属性。[2]

⑸ 操作：如果问题陈述中的名词有动作含义，则描述的操作就不是类。但是具有自身性质而且需要独立存在的操作应该描述成类。如我们只构造电话模型，"拨号"就是动态模型的一部分而不是类，但在电话拨号系统中，"拨号"是一个重要的类，它日期、时间、受话地点等属性。[2]

2.准备数据字典

为所有建模实体准备一个数据字典。准确描述各个类的精确含义，描述当前问题中的类的范围，包括对类的成员、用法方面的假设或限制。[2]

3.确定关联

两个或多个类之间的相互依赖就是关联。一种依赖表示一种关联，可用各种方式来实现关联，但在分析模型中应删除实现的考虑，以便设计时更为灵活。关联常用描述性动词或动词词组来表示，其中有物理位置的表示、传导的动作、通信、所有者关系、条件的满足等。从问题陈述中抽取所有可能的关联表述，把它们记下来，但不要过早去细化这些表述。[2]

下面是银行网络系统中所有可能的关联，大多数是直接抽取问题中的动词词组而得到的。在陈述中，有些动词词组表述的关联是不明显的。最后，还有一些关联与客观世界或人的假设有关，必须同用户一起核实这种关联，因为这种关联在问题陈述中找不到。[2]

银行网络问题陈述中的关联：

·银行网络包括出纳站和自动出纳机；

·分行共享自动出纳机；

·分理处提供分理处计算机；

·分理处计算机保存帐户；

·分理处计算机处理帐户支付事务；

·分理处拥有出纳站；

·出纳站与分理处计算机通信；

·出纳员为帐户录入事务；

·自动出纳机接受现金卡；

·自动出纳机与用户接口；

·自动出纳机发放现金；

·自动出纳机打印收据；

·系统处理并发访问；

·分理处提供软件；

·费用分摊给分理处。

隐含的动词词组：

·分行由分理处组成；

·分理处拥有帐户；

·分行拥有分行计算机；

·系统提供记录保管；

·系统提供安全；

·顾客有现金卡。

基于问题域知识的关联：

·分理处雇佣出纳员；

·现金卡访问帐户。[2]

使用下列标准去掉不必要和不正确的关联：

(1) 若某个类已被删除，那么与它有关的关联也必须删除或者用其它类来重新表述。在例中，我们删除了"银行网络"，相关的关联也要删除。

(2) 不相干的关联或实现阶段的关联：删除所有问题域之外的关联或涉及实现结构中的关联。如"系统处理并发访问"就是一种实现的概念。

(4) 派生关联：省略那些可以用其他关联来定义的关联。因为这种关联是冗余的。银行网络系统的初步对象图如图10-20所示。其中含有关联。 [2]

4.确定属性

属性是个体对象的性质，属性通常用修饰性的名词词组来表示.形容词常常表示具体的可枚举的属性值，属性不可能在问题陈述中完全表述出来，必须借助于应用域的知识及对客观世界的知识才可以找到它们。只考虑与具体应用直接相关的属性，不要考虑那些超出问题范围的属性。首先找出重要属性，避免那些只用于实现的属性，要为各个属性取有意义的名字。按下列标准删除不必要的和不正确的属性： [2]

(1)对象：若实体的独立存在比它的值重要，那么这个实体不是属性而是对象。如在邮政目录中，"城市"是一个属性，然而在人口普查中，"城市"则被看作是对象。在具体应用中，具有自身性质的实体一定是对象。

(2) 定词：若属性值取决于某种具体上下文，则可考虑把该属性重新表述为一个限定词。

(3) 名称：名称常常作为限定词而不是对象的属性，当名称不依赖于上下文关系时，名称即为一个对象属性，尤其是它不惟一时。

(4)标识符：在考虑对象模糊性时，引入对象标识符表示，在对象模型中不列出这些对象标识符，它是隐含在对象模型中，只列出存在于应用域的属性。

(5) 内部值：若属性描述了对外不透明的对象的内部状态，则应从对象模型中删除该属性。

(6) 细化：忽略那些不可能对大多数操作有影响的属性。 [2]

5.使用继承来细化类

使用继承来共享公共机构，以次来组织类，可以用两种方式来进行。

(1) 自底向上通过把现有类的共同性质一般化为父类，寻找具有相似的属性，关系或操作的类来发现继承。例如"远程事务"和"出纳事务"是类似的，可以一般化为"事务"。有些一般化结构常常是基于客观世界边界的现有分类，只要可能，尽量使用现有概念。对称性常有助于发现某些丢失的类。 [2]

(2) 自顶向下将现有的类细化为更具体的子类。具体化常常可以从应用域中明显看出来。应用域中各枚举字情况是最常见的具体化的来源。例如：菜单，可以有固定菜单，顶部菜单，弹出菜单，下拉菜单等，这就可以把菜单类具体细化为各种具体菜单的子类。当同一关联名出现多次且意义也相同时，应尽量具体化为相关联的类，例如"事务"从"出纳站"和"自动出纳机"进入，则"录入站"就是"出纳站"和"自动出纳站"的一般化。在类层次中，可以为具体的类分配属性和关联。各属性和都应分配给最一般的适合的类，有时也加上一些修正。 [2]

应用域中各枚举情况是最常见的具体化的来源。

6.完善对象模型

对象建模不可能一次就能保证模型是完全正确的，软件开发的整个过程就是一个不断完善的过程。模型的不同组成部分多半是在不同的阶段完成的，如果发现模型的缺陷，就必须返回到前期阶段去修改，有些细化工作是在动态模型和功能模型完成之后才开始进行的。 [2]

(1) 几种可能丢失对象的情况及解决办法：

·同一类中存在毫无关系的属性和操作，则分解这个类，使各部分相互关联；

·一般化体系不清楚，则可能分离扮演两种角色的类

·存在无目标类的操作，则找出并加上失去目标的类；

·存在名称及目的相同的冗余关联，则通过一般化创建丢失的父类，把关联组织在一起。 [2]

(2) 查找多余的类。

类中缺少属性，操作和关联，则可删除这个类。

(3)查找丢失的关联。

丢失了操作的访问路径，则加入新的关联以回答查询。 [2]

(4) 网络系统的具体情况作如下的修改：

①现金卡有多个独立的特性。把它分解为两个对象：卡片权限和现金卡。

a.卡片权限：它是银行用来鉴别用户访问权限的卡片，表示一个或多个用户帐户的访问权限；各个卡片权限对象中可能具有好几个现金卡，每张都带有安全码，卡片码，它们附在现金卡上，表现银行的卡片权限。 [2]

b.现金卡：它是自动出纳机得到表示码的数据卡片，它也是银行代码和现金卡代码的数据载体。

③"分理处"和"分离处理机"之间，"分行"和"分行处理机"之间的区别似乎并不影响分析，计算机的通信处理实际上是实现的概念，将"分理处计算机"并入到"分理处"，将"分行计算机"并入到"分行"。[2]

（三）建立动态模型

1.准备脚本

动态分析从寻找事件开始，然后确定各对象的可能事件顺序。在分析阶段不考虑算法的执行，算法是实现模型的一部分。[2]

2.确定事件

确定所有外部事件。事件包括所有来自或发往用户的信息、外部设备的信号、输入、转换和动作，可以发现正常事件，但不能遗漏条件和异常事件。[2]

3.准备事件跟踪表

把脚本表示成一个事件跟踪表，即不同对象之间的事件排序表，对象为表中的列，给每个对象分配一个独立的列。[2]

4.构造状态图

对各对象类建立状态图，反映对象接收和发送的事件，每个事件跟踪都对应于状态图中一条路径。[2]

（四）建立功能建模

功能模型用来说明值是如何计算的，表明值之间的依赖关系及相关的功能，数据流图有助于表示功能依赖关系，其中的处理应于状态图的活动和动作，其中的数据流对应于对象图中的对象或属性。[2]

1.确定输入值、输出值

先列出输入、输出值，输入、输出值是系统与外界之间的事件的参数。[2]

2.建立数据流图

数据流图说明输出值是怎样从输入值得来的，数据流图通常按层次组织。[2]

（五）确定操作

在建立对象模型时，确定了类、关联、结构和属性，还没有确定操作。只有建立了动态模型和功能模型之后，才可能最后确定类的操作。[2]

设计

面向对象设计是把分析阶段得到的需求转变成符合成本和质量要求的、抽象的系统实现方案的过程。从面向对象分析到面向对象设计，是一个逐渐扩充模型的过程。[2]

瀑布模型把设计进一步划分成概要设计和详细设计两个阶段，类似地，也可以把面向对象设计再细分为系统设计和对象设计。系统设计确定实现系统的策略和目标系统的高层结构。对象设计确定解空间中的类、关联、接口形式及实现操作的算法。[2]

（一）面向对象设计的准则

1.模块化

面向对象开发方法很自然地支持了把系统分解成模块的设计原则：对象就是模块。它是把数据结构和操作这些数据的方法紧密地结合在一起所构成的模块。分解系统为一组具有高内聚和松耦合的模块是模块化的属性。[2]

2.抽象

面向对象方法不仅支持过程抽象，而且支持数据抽象。

3.信息隐藏

在面向对象方法中，信息隐藏通过对象的封装性来实现。

4.低耦合

在面向对象方法中，对象是最基本的模块，因此，耦合主要指不同对象之间相互关联的紧密程度。低耦合是设计的一个重要标准，因为这有助于使得系统中某一部分的变化对其它部分的影响降到最低程度。[2]

5.高内聚

⑴操作内聚。

⑵类内聚。

1.设计结果应该清晰易懂

使设计结果清晰、易懂、易读是提高软件可维护性和可重用性的重要措施。显然，人们不会重用那些他们不理解的设计。[2]

要做到：

⑴用词一致。

⑵使用已有的协议。

⑶减少消息模式的数量。

⑷避免模糊的定义。[2]

2.一般——具体结构的深度应适当

3.设计简单类

应该尽量设计小而简单的类，这样便以开发和管理。为了保持简单，应注意以下几点：

⑴避免包含过多的属性。

⑵有明确的定义。

⑶尽量简化对象之间的合作关系。

⑷不要提供太多的操作。[2]

4.使用简单的协议

一般来说，消息中参数不要超过3个。

5.使用简单的操作

面向对象设计出来的类中的操作通常都很小，一般只有3至5行源程序语句，可以用仅含一个动词和一个宾语的简单句子描述它的功能[2]

6.把设计变动减至最小

通常，设计的质量越高，设计结果保持不变的时间也越长。即使出现必须修改设计的情况，也应该使修改的范围尽可能小。[2]

（三）系统设计

系统设计是问题求解及建立解答的高级策略。必须制定解决问题的基该方法，系统的高层结构形式包括子系统的分解、它的固有并发性、子系统分配给硬软件、数据存储管理、资源协调、软件控制实现、人机交互接口。

1.系统设计概述

设计阶段先从高层入手，然后细化。系统设计要决定整个结构及风格，这种结构为后面设计阶段的更详细策略的设计提供了基础。[2]

⑴系统分解。

系统中主要的组成部分称为子系统，子系统既不是一个对象也不是一个功能，而是类、关联、操作、事件和约束的集合。[2]

⑵确定并发性。

分析模型、现实世界及硬件中不少对象均是并发的。[2]

⑶处理器及任务分配。

各并发子系统必须分配给单个硬件单元，要么是一个一般的处理器，要么是一个具体的功能单元。[2]

⑷数据存储管理。

系统中的内部数据和外部数据的存储管理是一项重要的任务。通常各数据存储可以将数据结构、文件、数据库组合在一起，不同数据存储要在费用、访问时间、容量及可靠性之间做出折衷考虑。[2]

⑸全局资源的处理。

必须确定全局资源，并且制定访问全局资源的策略。

⑹选择软件控制机制。

设计中的大部分工作都与稳定的状态行为有关，但必须考虑用户使用系统的交互接口。[2]

2.系统结构的一般框架

3.系统分解——建立系统的体系结构

可用的软件库以及程序员的编程经验。

通过面向对象分析得到的问题域精确模型，为设计体系结构奠定了良好的基础，建立了完整的框架。[2]

4.选择软件控制机制

软件系统中存在两种控制流，外部控制流和内部控制流。

5.数据存储管理

数据存储管理是系统存储或检索对象的基本设施，它建立在某种数据存储管理系统之上，并且隔离了数据存储管理模式的影响。[2]

6.设计人机交互接口

在面向对象分析过程中，已经对用户界面需求作了初步分析，在面向对象设计过程中，则应该对系统的人机交互接口进行详细设计，以确定人机交互的细节，其中包括指定窗口和报表的形式、设计命令层次等项内容。[2]

（四）对象设计

1.对象设计概述

2.三种模型的结合

⑴获得操作。

⑵确定操作的目标对象。

3.算法设计

4.优化设计

5.控制的实现

6.调整继承

7.关联的设计[2]

面向对象程序设计

面向对象程序设计（英语：Object-oriented programming，缩写：OOP）是一种程序设计范型，同时也是一种程序开发的方法。对象指的是类的实例。[2]

已经被证实的是，面向对象程序设计推广了程序的灵活性和可维护性，并且在大型项目设计中广为应用。 此外，支持者声称面向对象程序设计要比以往的做法更加便于学习，因为它能够让人们更简单地设计并维护程序，使得程序更加便于分析、设计、理解。反对者在某些领域对此予以否认。[2]

当我们提到面向对象的时候，它不仅指一种程序设计方法。它更多意义上是一种程序开发方式。在这一方面，我们必须了解更多关于面向对象系统分析和面向对象设计（Object Oriented Design，简称OOD）方面的知识。[2]

面向对象的系统分析

面向对象的分析方法是利用面向对象的信息建模概念，如实体、关系、属性等，同时运用封装、继承、多态等机制来构造模拟现实系统的方法。[2]

传统的结构化设计方法的基本点是面向过程，系统被分解成若干个过程。而面向对象的方法是采用构造模型的观点，在系统的开发过程中，各个步骤的共同的目标是建造一个问题域的模型。在面向对象的设计中，初始元素是对象，然后将具有共同特征的对象归纳成类，组织类之间的等级关系，构造类库。在应用时，在类库中选择相应的类。[2]

实现

（一）程序设计语言

1.选择面向对象语言

在选择编程语言时，应该考虑的其他因素还有：对用户学习面向对象分析、设计和编码技术所能提供的培训操作；在使用这个面向对象语言期间能提供的技术支持；能提供给开发人员使用的开发工具、开发平台，对机器性能和内存的需求，集成已有软件的容易程度。[2]

2.程序设计风格

⑴提高重用性。

⑵提高可扩充性。

⑶提高健壮性。[2]

（二）类的实现

在开发过程中，类的实现是核心问题。在用面向对象风格所写的系统中，所有的数据都被封装在类的实例中。而整个程序则被封装在一个更高级的类中。在使用既存部件的面向对象系统中，可以只花费少量时间和工作量来实现软件。只要增加类的实例，开发少量的新类和实现各个对象之间互相通信的操作，就能建立需要的软件。[2]

一种方案是先开发一个比较小、比较简单的来，作为开发比较大、比较复杂的类的基础。

⑴“原封不动”重用。

⑵进化性重用。

一个能够完全符合要求特性的类可能并不存在。[2]

⑶“废弃性”开发。

不用任何重用来开发一个新类。

⑷错误处理。

一个类应是自主的，有责任定位和报告错误。[2]

（三）应用系统的实现

应用系统的实现是在所有的类都被实现之后的事。实现一个系统是一个比用过程性方法更简单、更简短的过程。有些实例将在其他类的初始化过程中使用。而其余的则必须用某种主过程显式地加以说明，或者当作系统最高层的类的表示的一部分。[2]

在C++和C中有一个main（）函数，可以使用这个过程来说明构成系统主要对象的那些类的实例。

（四）面向对象测试

⑴算法层。

⑵类层。

测试封装在同一个类中的所有方法和属性之间的相互作用。

⑶模板层。

测试一组协同工作的类之间的相互作用。

⑷系统层。

把各个子系统组装成完整的面向对象软件系统，在组装过程中同时进行测试。[2]

易混概念

“面向对象”和“基于对象”的区别

面向对象的三大特点（封装，继承，多态）缺一不可。通常“基于对象”是使用对象，但是无法利用现有的对象模板产生新的对象类型，继而产生新的对象，也就是说“基于对象”没有继承的特点。而“多态”表示为父类类型的子类对象实例，没有了继承的概念也就无从谈论“多态”。很多流行技术都是基于对象的，它们使用一些封装好的对象，调用对象的方法，设置对象的属性。但是它们无法让程序员派生新对象类型。他们只能使用现有对象的方法和属性。所以当你判断一个新的技术是否是面向对象的时候，通常可以使用后两个特性来加以判断。“面向对象”和“基于对象”都实现了“封装”的概念，但是面向对象实现了“继承和多态”，而“基于对象”没有实现这些，的确很饶口。[2]

“类库的创建者”和“类库的使用者”的区别

从事面向对象编程的人按照分工来说，可以分为“类库的创建者”和“类库的使用者”。使用类库的人并不都是具备了面向对象思想的人，通常知道如何继承和派生新对象就可以使用类库了，然而我们的思维并没有真正的转过来，使用类库只是在形式上是面向对象，而实质上只是库函数的一种扩展。[2]

了：面向对象是把构成问题事务分解成各个对象，建立对象的目的不是为了完成一个步骤，而是为了描叙某个事物在整个解决问题的步骤中的行为。[2]

可以拿生活中的实例来理解面向过程与面向对象，例如五子棋，面向过程的设计思路就是首先分析问题的步骤：1、开始游戏，2、黑子先走，3、绘制画面，4、判断输赢，5、轮到白子，6、绘制画面，7、判断输赢，8、返回步骤2，9、输出最后结果。把上面每个步骤用不同的方法来实现。[2]

如果是面向对象的设计思想来解决问题。面向对象的设计则是从另外的思路来解决问题。整个五子棋可以分为1、黑白双方，这两方的行为是一模一样的，2、棋盘系统，负责绘制画面，3、规则系统，负责判定诸如犯规、输赢等。第一类对象（玩家对象）负责接受用户输入，并告知第二类对象（棋盘对象）棋子布局的变化，棋盘对象接收到了棋子的变化就要负责在屏幕上面显示出这种变化，同时利用第三类对象（规则系统）来对棋局进行判定。[2]

可以明显地看出，面向对象是以功能来划分问题，而不是步骤。同样是绘制棋局，这样的行为在面向过程的设计中分散在了多个步骤中，很可能出现不同的绘制版本，因为通常设计人员会考虑到实际情况进行各种各样的简化。而面向对象的设计中，绘图只可能在棋盘对象中出现，从而保证了绘图的统一。[2]

例子

在面向对象方法中，对象可看成是属性（数据）以及这些属性上的专用操作的封装体。封装是一种信息屏蔽技术，封装的目的是使对象的定义和实现分离。[2]

Step1:

1.1新建一个工程命名为VBOOP；

1.2单击工程菜单，选择添加类模块后并单击确定按钮；

1.3在其属性窗口中将类的名称改为TScore。[2]

Step2：编辑TScore类模块代码

2.1.这里为TScore类定义四个私有（private）变量，它们只能在本模块

中是可见的，即类的一些成员被隐藏起来，用户只能通过属性过程

或函数等方法来访问，从而对对象进行封装。[2]

定义变量的基本语法：

private/public <；变量名> As <；变量类型>

代码部分：

private FName As String\'学生的姓名

private FMath As Single\'数学成绩

private FEnglish As Single \'英语成绩[2]

2.2.为TScore类定义六个公用(public）的属性（Property）过程和一

个计算总分的方法函数。

定义方法的基本语法：

private/public Property Get <；读属性过程名> As <；属性返回值类型>;

private/public Property Let <写属性过程名>(ByVal变量名As 返回值类型）；

private/public Function <；函数名> As <；函数返回值类型>;

get: 将模块中的私有变量的值赋给属性过程，通常称为读;

Let: 通过属性过程给模块中的私有变量值赋，通常称为写。[2]

代码部分：

public Property Get GetName() As String

GetName = FName

End Property

End Property

public Property Get GetMath() As Single

GetMath = FMath

End Property

public Property Let SetMath(ByVal Math As Single)

FMath = Math

End Property

public Property Get GetEnglish() As Single

GetEnglish = FEnglish

End Property

public Property Let SetEnglish(ByVal English As Single)

FEnglish = English

End Property

public Function Total() As Single\'计算总成绩函数

Total = GetMath + GetEnglish

End Function

Step3：回到Form1窗口，在窗口上添加12个控件：

3.1添加5个文本框txtName、txtMath、txtEN、txtTotal;

3.2添加5个标签labName、labMath、labEN、labTotal

其Caption属性分别为姓名、数学、英语、总成绩；

3.3添加2个命令按钮ComSetValues、ComSearch

其Caption属性分别为赋值、查询。

Step4：编辑窗口事件

4.1.构造Score对象及查询关键字SearchKey。在面向对象方法

中，我们可以这样说定义类就是定义数据类型，而声明对

象就是声明变量。也就是说，对象实际上就是变量。

Dim Score As New TScore

Dim SearchKey As String

4.2.给模块中四个私有变量赋值的单击事件

private Sub ComSetValues_Click()

If Val(txtMath.Text) >= 0 And Val(txtMath.Text) <= 100

And Val(txtEN.Text) >= 0 And Val(txtEN.Text) <= 100

Then

With Score

.SetName = txtName.Text

.SetMath = Val(txtMath.Text)

.SetEnglish = Val(txtEN.Text)

End With

```
txtEN.Text = \"\"
Print \"姓名：\" & Score.GetName & \"数学：\" & Score.GetMath & \"英语：\" & Score.GetEnglish
Else
MsgBox \"成绩的取值范围：[0,100]\",64,\"提示\"
End If
End Sub [2]
```

参考资料

1. 编程技术 . 百度百科[引用日期2012-12-12]
2. Stanley B.Lippman / Josee Lajoie / Barbara E.Moo . 《C++ Primer》：人民邮电出版社，2010

词条标签： 中国电子学会， 通信技术

猜你喜欢

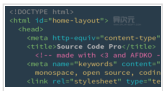
什么是面向对象	面向对象	怎样学习编程	怎么样学编程	java能做什么
怎么样学习编程	什么是编程	软件测试工程师	象棋游戏	学java哪个学校好


什么是单反相机

c语言实例教程


电脑编程入门自学


单片机培训机构


编程


单片机培训


怎么挽留女朋友