



[返回总目录](#)

目 录

| | |
|-----------------|---|
| 附录1 词 汇 表 | 2 |
|-----------------|---|

附录1 词汇表

Abstract class 抽象类。没有实例的类，抽象类只能用来继承。

Action 动作。可执行语句的过程。动作可能附加在状态转移时或用于指定活动。

Action expression 动作表达式。结果为一个或多个动作的表达式。

Action state 动作状态。当处于某种状态时执行内部动作时所处的状态。当执行动作时，动作状态会典型地从一个状态自动转移到另一个状态。

Activation 激活。执行一个动作。当一个对象正在被执行时称为被激活。

Active class 活动类。一个类（class），其对象是并发执行的，并且具有自己的控制线程。

Active object 活动对象。具有自己的控制线程的对象。

Activity diagram 活动图。以正在完成的工作为焦点显示系统的交互信息。活动图展示了在完成该工作而进行的一系列动作，包括交替的执行和对象。

Actor 角色。外部与系统进行交互的人或其它系统。

aggregate 聚合。一种关系，一个类由另一个类组成。该关系表明了一种“整体和部分”的连接，是关联的一种特殊情况。

Analysis 分析。用于分析问题，并对解决办法进行假设或分析。在面向对象分析中，会为现实世界中的所有实体、包括它们之间的关系、协作等创建一个模型，而无需考虑技术解决方案。

And-state 与状态。可能会同时（与其它状态）存在的子状态。

Application 版类化的组件《application》，表示一个可执行的程序。

Architecture 架构。“……软件系统的子系统和组件及其之间关系的一个描述。典型地，子系统和组件是通过不同的视图描述的，显示了软件系统与功能和非功能的性质。系统的软件架构是一个人工的产物（artifact），它是软件设计活动的结果”（Bushchmann, 1996）。

Argument 参数。在操作的说明（signature）中，值被解析成参数（parameter）。

Association 关联。描述类（及它们的对象）之间一组链接的关系。用来描述所涉及的类的对象之间具有某些种类的联系。

Association class 关联类。附属于某个关联的类，提供有关该连接的其它信息。关联类与普通的类相似，可以有属性、操作以及其它的关联。

Association role 关联角色。在一个关联中，它所涉及的每个类都可以连到一个或多个角色。角色名表明从该关联的角度来看，该类（及其对象）所扮演的角色。角色名是关联的一部分，不是类的一部分。

Asynchronous message 异步消息。当没有明确返回到调用方时发生异步控制流，发送方在发送完该消息后不等待它被处理就继续执行。

Attribute 属性。类中存储数据的成员。属性有名字、类型以及一个可能是默认的值。

Becomes 变成。一个版类化的相关（dependency）关系，它的源和目标是同一个实例（instance）。版类《becomes》的意思是，一个实例会从一个状态（源实例状态）变化到另一个状态（目标实例状态）。典型地，它是用来表明在物理架构中，如何把一个对象移动到其它地方（从一个节点移到另一个节点上）。

Behavior 行为。一个可见的效果，包括它的结果。

Bind 绑定。一个版类化的相关关系。相关关系连接类型或者协作，并且把参数与源和目的的属性进行匹配。绑定相关性说明，源是目标的一个绑定，其中的参数是为了使源定义更完整。典型地，它是用来表明一个参数化的类（模板）的实例化。

Boundary object 边界对象。一种版类化的类，它的对象是用来与角色（actor）进行通信的。

Broadcast 广播。应用到一组消息的一个约束（constraint）。广播指定不能以某种顺序启用该组消息。

Call 调用。一种版类化的相关关系，用来连接类中的操作。如果类 A 对类 B 有一个调用相关关系，那么 A 中的操作可能会调用任何 B 中的操作。

Class 类。一种用来为知识性和物理性事物建模的模型元素。类是一个类型（type），可以被实例化成对象，这些对象是同一个类的不同实体（对象）。类是用成员描述的，成员具有属性、操作和与其它类的关系。类可以被通用化和专有化，

Class diagram 类图。用类和它们的定义以及相互之间的关系展示静态结构。

Collaboration 协作。描述了一组对象如何进行交互来完成某个特定的功能，一个协作有一个上下文和一个交互。上下文显示了在该协作中涉及的对象以及它们之间的链接，交互显示了在该协作中对象所进行的通信。

Collaboration diagram 协作图。描述了对对象是如何交互，侧重于空间的协作，意即明确地给出对象之间的关系（链接）。

Complete inheritance 完全继承。完全的通用化约束，表明在一个继承关系中的所有子类都被指定，不能再添加其它子类。

Component 组件。使用类图或交互图中定义的逻辑模型元素描述的一个物理实现。可以从开发的不同阶段如编译时、链接时或运行时来观察组件。

Component diagram 组件图。描述了软件的（源，链接和可执行）组件以及它们相互之间的相关关系，表示了代码的结构。在展开图中组件是被划分到节点（node）中去的。

Composite state 复合状态。一个由子状态（与子状态或或子状态）组成的状态。

Composition 复合。复合聚合是一个聚合，其中整体拥有它的部分（强拥有：strong ownership）。部分位于整体内，并且和整体同时消亡，即如果整体不存在了，则部分也不存在。

Concrete class 具体类。一种可以有实例（对象）的类。与它相反的是抽象类。

Concurrency 并发。当两个或多个活动同时（并行）执行时。

Concurrent substate 并发子状态。见 Add-state。

Constraint 约束。一种语法条件或限制。UML 预定义了一些约束；其它的可以是用户自定义的。例如，一个约束可以限制哪些对象可以是某个特定关联的一部分。

Context 上下文。对协作所涉及的一组对象包括它们之间的链接的描述。

Control object 控制对象。一种版类化的类，它的对象典型地是负责实体对象的处理和操作序列。控制对象通常控制用例的实现。

Copy 复制。复制相关关系把两个实例连接起来，意思是源实例是目标实例的一个精确复制（但它们仍然有唯一和不同的标识符，因此它们是不同的实例）。

Deadlock 死锁。当多个线程在相互等待对方时发生。

Dependency 相关性。在两个模型元素之间的一种关系，对其中独立模型元素的更改将会对相关元素产生影响。

Deployment diagram 展开图。显示了过程、设备以及架构中所执行的软件组件的运行时架构。它是系统拓扑结构，包括硬件单元的结构以及每个单元上所执行软件的最终物理描述。

Design 设计。用来描述如何在一个技术环境中实现一个分析模型。设计模型应该提供足够的细节信息以便在一种程序设计语言中实现系统。

Destroyed 注销。影响对象的生命周期的一种约束。受到约束的对象在执行一次交互时被终止。

Diagram 图。一种视图元素，显示（映射）一组模型元素。UML 有九种图：用例图、组件图、类图、展开图、状态图、活动图、序列图、协作图和对象图。

Discriminator 鉴别子。见 Powertype。

Disjoint inheritance 不相交继承。从同一个超类继承的子类不能被专有化为同一个子类（用多继承）。不相交继承是默认值，与覆盖继承相反。

Document 文档。版类化组件《document》代表一个文档（其中是文献而不是能够被编译的源代码）。

Element 元素。是 UML 中多种要素的抽象基类。元素类是一个基点，其上可以附加许多机制。元素被专有化为模型元素、视图元素、系统和模型。模型元素是来自被建模的系统的一个抽象（如一个类、消息、节点、事件等）。视图元素是单个模型元素或一组模型元素（当视图元素是一个图时）的一个映射（文字的或图形的）。视图元素是文字或图形符号（如一个矩形代表一个类）。

Entity object 实体对象。实体版类是用于为持有信息的对象建模的，典型的是一些商业对象如债务、发票、保险合同等。通常，它们是持续地存储在系统中的。

Enumeration 枚举。一种版类化的原始类型，指定了枚举原始类型所有允许的值（一个定义域）。

Event 事件。在时间或空间上一个明显的发生。

Event-signature 事件说明。包括一个事件名和参数。

Export 导出。一个包可以把它的内容（元素）导出，以便其它包引用或使用。

Extends 扩展。在用例之间的一种通用化关系，其中一个用例通过增加动作把另一个用例扩展成一个更通用化的用例。扩展用例可能包含被扩展的用例（依扩展的条件而定）。由版类化的通用化《extends》描述。

Facade 外观。一种版类化的包只引用来自其它包的元素（导入，通过友好相关等）。外观没有自己的元素，但却典型地表示了包所提供服务的接口。

File 文件。版类化的组件《file》是一个包含源代码的文件。

Fire 起动。当附着的事件规范发生时称启动了一个状态转移。

Friend 友元。包或类之间版类化的友好相关表明友元素可以访问到被保护和私有可见的元素（通常不能访问）。

Generalizable element 可通用化元素。并不是对所有的元素都能进行通用化和专有化，只有可通用化元素才可以。可通用化元素是版类、包和类型（包括类型的子类型（subtype）和类）。

Generalization 通用化。在一个通用元素和一个更专有元素之间的一种关系。专有元素与通用元素完全一致并且还有更多的信息或行为。在使用通用元素的所有地方都可以使用专有元素的实例。

Global 全局的。应用于一个链接角色（role）的一种约束，指明相应的实例是可见的，因为它的辖域是全局的（可通过一个整个系统都知道的名字访问该实例）。

Guard-condition 守卫条件。状态转移的一个布尔表达式。如果守卫条件与某个事件说明结合，则该事件必然发生，并且该守卫条件可以附加到一个状态转移上，意思是指当条件成真时，将改变状态。

History indicator 历史指示符。用来记录内部状态。

Import 导入。包之间的版类化的导入相关关系是指一个包导入另一包的具有公共可见性的元素并能够访问它们。

Incomplete inheritance 不完全继承。一个不完全通用化表明可以在未来加入新的子类。与不完全通用化相反的是默认的完全通用化。

Inheritance 继承。见通用化。

Instance 实例。通过类型描述的单个成员。类（类型的子类）的实例是一个对象。

Interaction 交互。显示了类 / 对象是如何交互来完成某个特定功能。

Interaction diagram 交互图。序列图、协作图和活动图的通称。

Interface 接口。接口描述了类、包和组件与外部可见和可访问行为。

Interface inheritance 接口继承。接口之间的通用化和专有化（因为接口不具有实现，所以只继承

接口)。

Invariant 不变性。应用于类型，指定在该类型实例的整个生命周期中，必须满足某种性质（如条件）。

Label 标签。用在序列图的周边指定交互、循环、时间约束等。标签也可用在协作图和序列图的消息中，用来指定通信的细节（消息标签）。

Library 图书馆。版类化组件《library》说明了组件是动态还是静态图书馆。

Link 链接。实例之间的一种语义连接，是相关的实例化，是两个（或多个）对象之间的实际连接。

Link role 链接角色。相关角色的实例。

Local 局部的。对链接角色的一种约束，指定相应的实例是可见的，因为它在一个操作中是一个局部变量。

Member 成员。类型或类的一部分，一个属性或操作。

Message 消息。对象之间的通信，传递活动将会导致发生的事件信息，通常把接收消息也看成一个事件。

Metaclass 元类。可以被实例化为其它类的类（是类的类）。

Metamodel 元模型。用元语言来描述其它模型的模型。元模型是用来描述 UML 的，UML 既是一种建模语言也是一种元语言，因为它用它自己表示自己。

Metatype 元类型。实例是类型的类型。

Method 方法。操作的实现。

Model 模型。用图表达的、系统的一个抽象描述。

Model coordination 模型协调。必须能把同一事物的不同模型集成到一起，并找出相互之间的关系。应该能在各个不同的抽象层次上（如结构和行为）以及不同的层次之间（如，系统与子系统，分析与设计）协调模型。

Model element 模型元素。UML 的概念，如类、对象、节点和状态等。大多数模型元素都有相对应的视图元素，是它们的图形形象，从而在图中可以映射到它们。

Model integration 模型集成。如果一组模型具有相同的意图，并且表示同一事物，则应该能够不产生任何二义性地把它们集成到一起。

Model quality 模型质量。在模型质量高的模型中，所有模型都有明确和清楚的意图，并且都表达了被研究和建模事物的本质。所有模型都应该易于通信、验证、确认和维护。

Modeling language 建模语言。用语法和语义定义的、用来表示模型的语言。一些建模语言还有一些实用的规则。

Multiple inheritance 多继承。一种类型是由一个以上超类型专有化而来的。

Multiplicity 重数。允许的链接数量以及它们如何链接两端的对象。一个关联或聚合的每个角色都有一个非负整数的取值范围，指定了每个角色所允许的对象的数量。

Name 名。一个用来标识模型元素的字符串。

New 新建。一种影响对象整个生命周期的约束，描述了对象是在该交互的执行过程中被创建的。

Node 节点。具有某种计算性资源的物理对象（设备），包括有进程的计算机，以及打印机、读卡机和通信等设备。

Note 笔记。附加在一个或一组元素上的一个注释，笔记没有语义。

Object 类的一个实例。

Object diagram 对象图。系统执行的一个快照，显示了对象以及它们之间的链接。

Object lifeline 对象生命线。在序列图中的一条线，表示某个对象的存在。

Operation 操作。对象的成员。操作被定义成有一个说明及实现的函数。

Or-constraint 或约束。应用于一组链接受到限制的关联。当关联把某单个类和一组其它的类连接起来时可应用或约束，该约束指定该单个类的一个对象只能被连接到相关类中的一个对象（在该相关的对面）。

Or-state 或状态。不能同时存在的子状态。

Overlapping inheritance 覆盖继承。用多继承可以把从同一超类继承而来的子类专有化为同一子类，与不相交继承相反。

Package 包。链接元素的一种组合机制；例如，创建图、对象、类和节点等的包。

Page 页面。版类化的组件《page》表示一个 Web 页面。

Parameter 参数。可以被更改、传递或返回的变量的规格说明。参数可包括名、类型和指针。操作、消息和事件中用到参数。

Parameterized class 参数化的类。一种不完整的类，需要与一个参数绑定（典型地是一个类型）从而变成完整的。用来表示用类型（如类和原始类型）填补的通用类来实例化其它更为专有化的类。通常，参数化的类是使用继承的另一选择。在 C++ 中，参数化的类叫做模板（template）。

Pattern 模式。用来设计面向对象系统的灵活、通用、经过证明的、简单而可重用的解决方案。在 UML 中用协作为模式建档。

Persistence 持续性。应用于类型，把一个类定义成持续性的，意味着该类的对象可以被存放在一个数据库或文件中，并且在程序不同的执行中，该对象还可以保留它自己的值（状态）。

Postcondition 后置条件。在操作完成后必须为真的一个条件。

Powertype 动力类型。当对通用化进行专有化时，必须使用一个鉴别子来表明该继承的基础，鉴别子用于区分实例，它的类型叫做动力类型。动力类型属于类型层，而鉴别子属于实例层。

Precondition 前置条件。在操作被执行前必须为真的条件。

Primitive type 原始类型。非类的类型，如整数和枚举类型。UML 中没有预定义的原始类型。

Process (activities) 进程（活动）。对一组如果能够正确完成则将满足某个明确目标的相关活动的描述。

Process (program execution) 进程（程序执行）。一种“重”的控制线程，与线程（一种“轻”的控制线程）相反。进程和线程之间的重要区别是进程通常把它所有的内部结构和执行封装在它自己的内存空间中，而线程在一个与其它线程共享空间内执行。

Process user 进程用户。进程的用户。在软件工程的进程概念下，通常指的是开发人员。

Projection 映射。模型元素通过视图元素映射在图中。例如，类是一种模型元素，它的视图元素是一个矩形。

Property 性质。对固有性质的一种描述，如元素的名字。性质可以是附加到模型元素的预定义或用户定义的有标记的值。性质表指定了属性的取值范围。

Qualifier 量词。量化的关联是与一到多或多到多关联一起用的。量词区分关联多端的对象集合（如，在关联的对象导航中是关键）。

Recursion 递归。当操作调用它自身时（直到一个条件为真）。

Refinement 精化。对同一事物的两个不同抽象层次的描述之间的一种关系。精化是一种版类化的相关关系，可用来把同一事物的分析描述和设计描述连接起来。

Relationship 关系。模型元素的一种语义连接。关系被专有化为通用化、相关、关联、转移和链接。

Role 角色。关联（聚合）所涉及的每个类可以附加角色，表明从该关联的角度该类所扮演的角色。角色是一种为类及其对象指定上下文的非常有用的技术。角色名是关联的一部分，而不是类的一部分。

Scheduling 调度。活动对象之间的部分同步是通过对活动对象的调度完成的。当有一组线程正在等待执行时，由调度确定下一步该执行哪个线程。

Self 自我。对链接角色的一种约束，指明对象可以向它自己发送消息（典型地被调度程序使用）。

Semantics 语义。用来描述事物的意思，可把语义看作概念及其符号之间的一个链接，并且当进行有关该概念的交流时，必须知道这种链接，否则会产生误解。

Send-clause 发送子句。动作发生的变化；在状态图中，在两个状态之间转移时发送消息的语法。

Sequence diagram 序列图。描述对象之间如何相互交互。序列图侧重时间，即它体现出何时发送和接收消息。

Signal 信号。一种版类化的类，其对象被作为消息发送。

Signature 说明。通常是操作的名，包括参数表及返回类型。

Simple message 简单消息。表示一条控制流，不涉及任何有关通信的细节而显示了控制是如何从一个对象传递给另一个对象。当有关通信的细节是未知或认为与图中无关时，采用这种消息类型。

Starvation 饥饿。某个线程（活动对象）不能再得到运行，这种情况发生在线程优先级的定义使得其它线程无法或很难得到控制权时。

State 状态。对象的状态是由它的属性值及其与其它对象的链接决定的。状态是对象内以前活动的结果。

State diagram 状态图。显示了对对象的生命周期（也是子系统和系统的生命周期）。状态图显示了事件（消息、时间、差错以及状态改变）是如何随着时间影响对象的状态的。

State vertex 状态顶点。一个状态转移的源或目标。

Stereotype 版类。建模元素的一种类型，扩展了 UML 的语义。版类必须以 UML 中已经定义的元素为基础，可以扩展语义但不能扩展已经存在元素的结构。某些版类是 UML 预定义好的，用户可以自定义版类。

Stub 残段。版类化的包，代表一个不完整实现的包。残段表示另一个系统的一小部分。

Subclass 子类。其它类的专有化的一个类。

Substate 子状态。其它状态内的一个状态。

Subsystem 子系统。系统的一部分。在 UML 中用包表示子系统。

Subtype 子类型。是其它类型的专有化的一个类型。

Superclass 超类。其它某个类的通用化类。

Superstate 超状态。包含其它状态和子状态的状态。

Supertype 超类型。其它某种类型的通用化的类型。

Swimlane 泳道。在活动图中，根据它们的职责划分的一组活动。泳道可用于几种目的，例如，用来明确地说明在何处（由哪个对象）完成某个动作或在机构的哪个部分完成该工作。

Synchronization 同步。同步机制是用来控制并发线程执行的对象，确保不出现共享资源使用冲突或整体上对资源无效的使用。

Synchronous message 同步消息。嵌套的控制流，典型地作为一个操作调用来实现。在调用方恢复执行前，处理该消息的操作（包括为处理该消息而发出的其它嵌套消息）必须已经完成。

Syntax 语法。限制概念（元素）相互结合的规则。

System 系统。以某种方式组织起来的项，例如信息系统，商业系统或嵌入式系统。

Table 表。版类化组件《table》表示一个数据库表。

Tagged value 加标签值。把性质明确定义成一个名-值对。在加标签值中，名被称为标签，UML 预定义了一些特定的标签。在 UML 中，性质一般地用来与元素有关的值中，包括类的属性、关联和加标签值。

Template 模板。见参数化类。

Thread 线程。进程是一个“重”控制流，而线程是一个“轻”控制流。进程和线程之间最重要的

区别在于，进程通常把它所有的内部结构和执行封装在自己的内存空间，而线程在与其它线程共享的内存空间中执行。

Time event 时间事件。在一个指定事件（通常是一个状态的入口）之后的一段指定时间的消息。

Tiem expression 时间事件的表达式。

Trace 跟踪。从一个模型元素到另一个模型元素之间的一种版类化的相关关系。互相被跟踪的元素可以在同一个图中或者在不同的图中。跟踪指定源必须能够在概念上跟踪回目标。

Transient 过渡。影响对象的生命周期的一种约束。过渡对象的创建和注销是在一个协作中的同一执行中进行的，也就是说，过渡约束是创建和注销约束的结合。

Transition 转移。两个状态之间的一种关系，表明当某个指定的事件发生并且/或满足某些指定条件时，处在第一个状态的对象将完成某种指定的动作并进入第二个状态。

Type 类型。对一组共享相同的操作、属性、关系和语义的实例的一种描述，原始类型、类和用例都是类型。

Use case 用例。（从一个外部角色的角度）描述如何使用系统。用例说明了系统的功能，并且是用外部角色、用例和被建模的系统的角度来描述。用例应该对某个特定角色产生一个可见的结果。

Use-case diagram 用例图。用例模型是用用例图描述的，其中包含了系统、角色和用例等元素，显示了这些元素之间不同的关系。

Use-case model 用例模型。用用例来描述系统的功能需求。

Uses 使用。一种通用化关系，一个用例使用另一个用例，指定专有化的用例包括了通用用例的行为。

Utility 公用类型。一种版类化的类型，只包含类辖域范围的操作和属性，不能对公用类型进行实例化。

Value 值。类型定义域的一个元素。类型定义域指定了某个特定类型，例如数 42 属于整数类型。

View 视图。显示了被建模系统的不同侧面。视图不是一个图表（graph），而是一个抽象，由一组图（diagram，是实际的图表）组成。

View element 视图元素。一个或多个模型元素的映射（也叫做表示）。

Visibility 可见性。所有公共的、保护的、私有的和实现的允许值的枚举。可见性指定在类型和包内所能允许对元素的访问。

Vote 投票。应用于消息的一种约束。投票包含了一组返回的消息，投票约束指定必须在返回的值中通过多数投票才能选出返回值。