GB

中国国家标准汇编

526

GB 28174

（2011年制定）

中国标准出版社

中 国 国 家 标 准 汇 编

526

GB 28174

（2011 年制定）

中国标准出版社　编

中国标准出版社

北　京

图书在版编目(CIP)数据

中国国家标准汇编:2011 年制定.526:GB 28174/中国标准出版社编.—北京:中国标准出版社,2012
ISBN 978-7-5066-6975-7

Ⅰ.①中… Ⅱ.①中… Ⅲ.①国家标准-汇编-中国-2011 Ⅳ.①T-652.1

中国版本图书馆 CIP 数据核字(2012)第 197845 号

中国标准出版社出版发行
北京市朝阳区和平里西街甲 2 号(100013)
北京市西城区三里河北街 16 号(100045)

网址 www.spc.net.cn
总编室:(010)64275323 发行中心:(010)51780235
读者服务部:(010)68523946

中国标准出版社秦皇岛印刷厂印刷
各地新华书店经销

开本 880×1230 1/16 印张 43.25 字数 1 294 千字
2012 年 10 月第一版 2012 年 10 月第一次印刷

*

定价 220.00 元

出 版 说 明

1.《中国国家标准汇编》是一部大型综合性国家标准全集。自1983年起，按国家标准顺序号以精装本、平装本两种装帧形式陆续分册汇编出版。它在一定程度上反映了我国建国以来标准化事业发展的基本情况和主要成就，是各级标准化管理机构，工矿企事业单位，农林牧副渔系统，科研、设计、教学等部门必不可少的工具书。

2.《中国国家标准汇编》收入我国每年正式发布的全部国家标准，分为“制定”卷和“修订”卷两种编辑版本。

“制定”卷收入上一年度我国发布的、新制定的国家标准，顺延前年度标准编号分成若干分册，封面和书脊上注明“20××年制定”字样及分册号，分册号一直连续。各分册中的标准是按照标准编号顺序连续排列的，如有标准顺序号缺号的，除特殊情况注明外，暂为空号。

“修订”卷收入上一年度我国发布的、被修订的国家标准，视篇幅分设若干分册，但与“制定”卷分册号无关联，仅在封面和书脊上注明“20××年修订-1，-2，-3，……”字样。“修订”卷各分册中的标准，仍按标准编号顺序排列（但不连续）；如有遗漏的，均在当年最后一分册中补齐。需提请读者注意的是，个别非顺延前年度标准编号的新制定的国家标准没有收入在“制定”卷中，而是收入在“修订”卷中。

读者配套购买《中国国家标准汇编》“制定”卷和“修订”卷则可收齐由我社出版的上一年度我国制定和修订的全部国家标准。

3. 由于读者需求的变化，自1996年起，《中国国家标准汇编》仅出版精装本。

4. 2011年我国制修订国家标准共1 989项。本分册为“2011年制定”卷第526分册，收入国家标准GB 28174的最新版本。

中国标准出版社

2012年8月

目　　录

ICS 35.080
L 77

中华人民共和国国家标准

GB/T 28174.1—2011

统一建模语言(UML)
第1部分:基础结构

Unified modeling language(UML)—
Part 1:Infrastructure

2011-12-30 发布　　2012-06-01 实施

中华人民共和国国家质量监督检验检疫总局
中国国家标准化管理委员会　发布

前　言

GB/T 28174《统一建模语言(UML)》分为4个部分：

——第1部分：基础结构；

——第2部分：上层结构；

——第3部分：对象约束语言(OCL)；

——第4部分：图交换。

本部分为GB/T 28174的第1部分。

本部分按照GB/T 1.1—2009给出的规则起草。

本部分参考面向对象工作组(OMG)的《统一建模语言：基础结构》2.0版。

请注意本文件的某些内容可能涉及专利。本文件的发布机构不承担识别这些专利的责任。

本部分由全国信息技术标准化技术委员会(SAC/TC 28)提出并归口。

本部分起草单位：北京大学、广东省广业信息产业集团有限公司、广东万维博通信息技术有限公司、中国电子技术标准化研究所。

本部分主要起草人：麻志毅、许立勇、周伟强、唐泽欢、江善东、高健。

引　言

统一建模语言(UML)是一种可视化规约语言，用于定义和构造计算机信息系统的制品，并将其文档化。它是一种通用建模语言，可以和所有主流的面向对象和面向构件的方法一起使用，并适用于所有的应用领域和实现平台(如：CORBA、J2EE、.NET 等)。

0.1　统一建模语言不同版本之间的关系

由于 UML 的技术较新，所以该国际标准历经多次的版本演化，下面是 UML 在 OMG 的演化过程：

1997　　UML1.1
1998　　UML1.2
1999　　UML1.3
2001　　UML1.4
2003　　UML2.0

GB/T 28174 的本部分正文中的 UML 均指 UML2.0 统一建模语言和 GB/T 28174。

0.2　关于对读者的建议

需要了解语言中的元模型构造物，利用这些构造物进行元模型扩展或者是构造新的建模语言的用户可阅读基础结构部分(GB/T 28174.1)。

应用系统建模用户和建模工具制造方都需阅读上层结构(GB/T 28174.2)。但要注意，该部分的内容是交叉引用的，可不按目次顺序阅读。

对于要精确地对模型进行约束的应用系统建模用户或要支持对象约束语言的建模工具制造方，需阅读对象约束语言部分(GB/T 28174.3)。

支持在不同的软件工具间平滑且无缝地交换文档的建模工具制造方，需阅读图交互部分。

0.3　关于本部分

本部分的第 4 章和第 5 章描述了定义 UML 语言体系结构和规格说明的方法。

本部分的第 6 章至第 10 章描述了元模型的基础结构库(Infrastructure Library)的结构和内容，这些元模型包括 UML 元模型和相关元模型，如元对象设施(MOF)和公共仓库元模型(CWM)。基础结构库定义了 UML 的可重用元语言核心与元模型扩展机制。元语言核心能够用于制定各种元模型，包括 UML、MOF 和 CWM。另外，基础结构库还定义了一种外廓扩展机制，当某些平台和建模领域不具备元模型建模能力时，利用这种扩展机制可以为这些平台对 UML 进行定制。基础结构库的最顶层包如图 1 所示。

核心包是基础结构库可重用部分的主体，而且被进一步细分，如图 3 所示。

原子类型(PrimitiveTypes)包比较简单，它包含若干预定义类型，预定义类型通常用于元模型(metamodeling)建模，因此它们不但用于基础结构库本身，而且用于 MOF 和 UML 等元模型(metamodels)。抽象包包括若干只含有少量元模型且粒度适当的包，它们中的大部分是抽象的。这个包的目的是提供高可用的元类集，在定义元模型时被特化。构造包也包含若干粒度适当的包，且把抽象包的多个方面集中在一起。构造包中的元类趋向于具体而不是抽象，并且适用于面向对象建模范式。来看一下元模型，如 MOF 和 UML，它们通常因为要自动输入核心中其他包的内容而引入构造包。基本包(Basic)包括一个构造包的子集，它主要是为了使用 XMI。

外廓包(profiles)包含创建特定元模型外廓的机制，尤其是对 UML 的扩展。这种扩展机制是 MOF 提供的通用扩展功能的子集。

统一建模语言(UML)
第1部分:基础结构

1 范围

GB/T 28174的本部分规定了用于对各类软件系统进行可视化、详述、构造和文档化的统一建模语言。本语言也可用于对其他领域进行建模。

本部分适用于统一建模语言(UML)的基础语言构造物,包括讲述UML的体系结构、UML的设计原理以及如何应用这些原理来组织UML的方法。

2 规范性引用文件

下列文件对于本文件的应用是必不可少的。凡是注日期的引用文件,仅注日期的版本适用于本文件。凡是不注日期的引用文件,其最新版本(包括所有的修改单)适用于本文件。

GB/T 28174.2 统一建模语言(UML) 第2部分:上层结构

GB/T 28174.3 统一建模语言(UML) 第3部分:对象约束语言(OCL)

GB/T 28174.4 统一建模语言(UML) 第4部分:图交换

3 术语和定义、缩略语

3.1 术语和定义

下列术语和定义适用于本文件,也适用于GB/T 28174.2、GB/T 28174.3和GB/T 28174.4。

3.1.1

抽象类 abstract class

不能直接被实例化的类。

相对语:**具体类**(3.1.49)

3.1.2

抽象 abstractio

强调事物的一定特征而忽视无关的其他特征的结果。所定义的抽象与观察者的视角有关。

3.1.3

动作 action

行为规约的基础单元,用以描述所建模系统(计算机系统或现实世界系统)中的转换或处理。动作包含在活动中,活动提供动作的语境。

见:**活动**(3.1.9)。

3.1.4

动作序列 action sequence

解析为一系列动作的表达式。

3.1.5

动作状态 action state

表示原子动作执行的状态,通常为操作调用。

3.1.6

激活 activation

动作执行的启动。

3.1.7

主动类 active class

其实例为主动对象的类。

见：**主动对象**(3.1.8)。

3.1.8

主动对象 active object

可以执行其自己的行为而不要求方法调用的对象。有时把它称为“拥有控制线程的对象”。主动对象响应来自其他对象的通信点，由主动对象的行为单独决定，而不是由调用对象决定。这暗示着主动对象在一定程度上是自治的和交互式的。

见：**主动类**(3.1.7)**和**(**控制**)**线程**(3.1.206)。

3.1.9

活动 activity

通过顺序化的从属单元(其基本元素是单个的动作)，表示为执行流的参数化行为的规约。

见：**动作**(3.1.3)。

3.1.10

活动图 activity diagram

使用控制和数据流模型描绘行为的图。

3.1.11

活动者 actor

参与者

在用况中使用的构造物，它定义了当一个用户或任何其他系统与所考虑中的本系统交互时所扮演的一个角色。它是相互作用的实体的类型，但它本身是外部的事物。活动者可表示为人员用户、外部硬件或其他事物。一个活动者不必表示为一个特定的物理实体。例如，单个物理实体可以扮演几个不同的活动者，反过来，单个给定的活动者可以由多个物理实体扮演。

3.1.12

聚集 aggregate

在聚合关系(整体—部分)中作为“整体”的类。

见：**聚合**(3.1.13)。

3.1.13

聚合 aggregation

关联的一种特殊形式，它描述聚集(整体)和部件(部分)间的整体—部分关系。

见：**组合**(3.1.48)。

3.1.14

分析 analysis

系统开发过程的一个阶段，其主要目的是形成独立于实现考虑的问题域模型。分析注重于做什么，设计注重于如何做。

见：**设计**(3.1.63)。

3.1.15

分析时[期] analysis time

涉及在软件开发过程的分析阶段期间发生的事情。

见：**设计时**(3.1.64)，**建模时**(3.1.120)。

3.1.16

实参　argument

实际参数　actual parameter

对随后要解决的参数的绑定。一个独立的变量。

3.1.17

制品　artifact

产品　product

开发过程所使用或产生的一个物理信息片段。例如，模型、源文件、脚本、二进制可执行文件。可以用制品构成可部署的构件的实现。

相对语：**构件**(3.1.43)。

3.1.18

关联　association

在类目实例间可出现的关系。

3.1.19

关联类　association class

兼有关联和类的性质的模型元素。可以把关联类看作是具有类的性质的关联，或看作具有关联的性质的类。

3.1.20

关联端　association end

关联的端点，它把关联连接到类目。

3.1.21

属性　attribute

类目的结构性特征，它刻画类目的实例。通过命名关系，一个属性把类目的一个实例与一个值或多值联系起来。

3.1.22

辅助类　auxiliary class

一种衍型化的类，通常通过实现从属的逻辑或控制流，它支持另外的更核心的或更基础的类。通常将辅助类与焦点类一起使用，对于在设计阶段规约构件的辅助逻辑和控制流也有一定的作用。

3.1.23

行为　behavior

操作或事件的可观察的效果(包括结果)。它规约产生行为特征的效果的计算。可以采取若干形式来描述行为：交互、状态机、活动或过程(一组动作)。

3.1.24

行为图　behavior diagram

描绘行为特征的一种图形式。

3.1.25

行为特征　behavioral feature

模型元素的动态特征，例如操作或方法。

3.1.26

行为模型侧面　behavioral model aspect

强调系统中的实例行为的模型侧面，包括实例的方法、协作和状态历史。

3.1.27

二元关联 binary association

两个类之间的关联。是 n 元关联的特例。

3.1.28

绑定 binding

通过为模板参数提供实参，从模板创建模型元素。

3.1.29

布尔型 boolean

取值范围为真或假的枚举类型。

3.1.30

布尔表达式 boolean expression

求值为布尔值的表达式。

3.1.31

势 cardinality

集合中的元素个数。

相对语：**势域**(3.1.123)。

3.1.32

子 child

在泛化关系中，另一个元素(父)的特化。

见：**子类**(3.1.193)**和子类型**(3.1.198)。

相对语：**父**(3.1.138)。

3.1.33

调用 call

调用类目上的一个操作的一种动作状态。

3.1.34

类 class

描述一组对象的类目，这些对象共享关于特征、约束和语义的同一规约。

3.1.35

类目 classifier

一组某些方面相同的实例的集合。类目可以具有刻画其实例的特征。类目包含接口、类、数据类型和构件。

3.1.36

分类 classification

实例到类目的指派。

见：**动态分类**(3.1.70)、**多重分类**(3.1.121)、**静态分类**(3.1.185)。

3.1.37

类图 class diagram

显示一组说明性的(静态)模型元素的图，例如，这样的元素可为类、类型以及它们的内容及关系。

3.1.38

客户 client

请求其他类目服务的类目。

相对语：**供方**(3.1.201)。

3.1.39

协作　collaboration

如何实现操作或像用况这样的类目的规约，这样的实现是由用特定的方法扮演特定的角色的一组类目和关联实施的。

见：**交互**(3.1.100)。

3.1.40

协作发生　collaboration occurrence

协作的特殊使用，用以解释一个类目的各部件间或一个操作的各性质间的关系。它也可用以指示协作如何表示类目或操作。一个协作发生指明了一组角色或连接件，按照给定的协作(由协作发生的类型指定)，它们在特定的类目或操作中进行合作。在一个类目或操作中，一个给定的协作可以有多个发生，每一个都涉及一组不同的角色和连接件。一个给定的角色或操作可以出现在同一个或不同协作的多个发生中。

见：**协作**(3.1.39)。

3.1.41

通信图　communication diagram

注重于在生命线间交互的图，在图中描述的核心是内部结构的体系结构以及如何响应传递过来的消息。通过用顺序号的模式给出消息的顺序。顺序图和通信图表达类似的信息，但表示的方式不同。

见：**顺序图**(3.1.175)。

3.1.42

编译时(期)　compile time

涉及到在编译软件模块期间发生的事情。

见：**建模时**(3.1.120)、**运行时**(3.1.170)。

3.1.43

构件　component

系统的模块化部分，它封装自己的内容，且它的声明在其环境中是可以替换的。构件利用提供和请求接口定义自身的行为。这样，构件起类型的作用，其一致性由提供和请求接口来定义(包含静态和动态语义)。

3.1.44

构件图　component diagram

显示构件间的组织和依赖的图。

3.1.45

组合类　composite

一个通过组合关系与一个或多个类发生关系的类。

见：**组合**(3.1.48)。

3.1.46

组合状态　composite state

由并发(正交)子状态或顺序(不相交)子状态组成的状态。

见：**子状态**(3.1.195)。

3.1.47

组合结构图　composite structure diagram

描述类目内部结构，包括该类目与系统其他部分的交互点的图。它图示了共同地执行容器类目的行为的部件的配置。这种体系结构图规约了在特定语境中一组扮演部件(角色)的实例，以及它们所需要的关系。

3.1.48

组合 composition

组成聚合 composite aggregation

聚合的一种形式，它要求部分实例一次最多包含在一个组成类中，组成对象负责创建和销毁其部分。组合可以是递归的。

3.1.49

具体类 concrete class

能直接被实例化的类。

相对语：**抽象类**(3.1.1)。

3.1.50

并发 concurrency

在同一时间段内两个或多个活动的发生。通过交错或同时执行两个或多个线程，实现并发。

见：**线程**(3.1.206)。

3.1.51

并发子状态 concurrent substate

与包含在同一组合状态中的其他子状态同时存在的子状态。

见：**组合状态**(3.1.46)。

相对语：**不相交子状态**(3.1.67)。

3.1.52

可连接元素 connectable element

抽象元类，用以表示可以通过连接件链接的模型元素。

见：**连接件**(3.1.53)。

3.1.53

连接件 connector

使得能够在两个或多个实例间进行通信的链接。可用像指针这样简单的事物或像网络连接这样的复杂事物实现链接。

3.1.54

约束 constraint

语义条件或限制。为了阐述一些模型元素的语义，约束可以用自然语言文本、数学形式化表示法或机器可读的语言来表达。

3.1.55

容器 container

a) 包含其他实例的实例，它提供访问或遍历其内容的操作。如数组、表或集合。

b) 包含其他构件的构件。

3.1.56

包容层次 containment hierarchy

由模型元素以及其间的包容关系组成的命名空间层次。一个包容层次形成一张图。

3.1.57

语境 context

用于特定目的(如规约操作)的一组相关建模元素的视图。

3.1.58

数据类型 data type

其值没有标识的类型，即这样的值是纯值。数据类型包括内建的基本类型(如整型和串)和枚举类型。

3.1.59

委托 delegation

一个对象把消息发给另一个对象让其响应的能力。委托是继承的一种替代方案。

相对语：**继承**(3.1.97)。

3.1.60

依赖 dependency

两个建模元素之间的关系，其中一个建模元素(独立元素)的改变会影响另一建模元素(依赖元素)。

3.1.61

部署图 deployment diagram

描述系统执行的体系结构的图。它把系统制品表示为结点(通过通信路径连接结点能创建具有任意复杂性的网络)。结点通常以嵌套的方式定义，并表示硬件设备或软件执行环境。

见：**构件图**(3.1.44)。

3.1.62

派生元素 derived element

能从其他元素计算出的模型元素，说明它是为了清晰可见，或者说为了设计的目的包含了它，即使它没有添加什么语义信息。

3.1.63

设计 design

系统开发过程的一个阶段，其主要目的是决定怎样实现系统。在设计期间所做的策略和技术决策，用于满足所要求的系统功能需求和质量需求。

3.1.64

设计时(期) design time

涉及在系统开发过程的设计阶段发生的事情。

见：**建模时**(3.1.120)。

相对语：**分析时**(3.1.15)。

3.1.65

开发过程 development process

在系统开发期间，为特定目的而进行的一组部分有序的步骤，如构造模型或实现模型。

3.1.66

图 diagram

一组模型元素的图形表示，大多数情况下绘制为由弧(关系)和顶点(其他模型元素)组成的连通图。GB/T 28174.2 的附录 A 列出了 UML 所支持的各种图。

3.1.67

不相交子状态 disjoint substate

不能与包容在同一组合状态内的其他子状态同时存在的子状态。

见：**组合状态**(3.1.46)。

3.1.68

分布单元 distribution unit

一组被分配到一个进程或一个处理器的对象或构件集合。可以用运行时组成类或聚集表示分布单元。

3.1.69

域 domain

用一组概念和术语刻画的知识领域或活动领域，由该领域的实践者理解。

3.1.70

动态分类　dynamic classification

一个实例从一个类目到另一个类目的指派。

相对语：**多重分类**(3.1.121)，**静态分类**(3.1.185)。

3.1.71

元素　element

模型的成分。

3.1.72

进入动作　entry action

在一个状态机中，当一个对象进入一个状态时有一个方法执行的动作，不考虑达到该状态所采取的转换。

3.1.73

枚举　enumeration

是一种数据类型，其实例是命名值的列表。例如，RGBColor＝{red，green，blue}。布尔是一种预定义的枚举，其值取自集合{false，true}。

3.1.74

事件　event

对有意义的发生的规约，该发生在时间和空间上有特定位置，并引起相关行为的执行。在状态图的语境中，事件是能触发转换的发生。

3.1.75

异常　exception

一种特殊的信号，通常用以表示故障情形。异常的发送方使执行终止，并且执行由异常的接收者继续，异常的接收者也可能是发送方本身。异常的接收者隐式地由执行期间的交互顺序决定，不显式地指定它。

3.1.76

执行发生　execution occurence

在交互图上表示的生命线中的一种行为单元。

3.1.77

退出动作　exit action

当对象退出状态机的某一状态时由方法执行的动作，而不管退出时所采取的转移。

3.1.78

引出　export

在包的语境中，使某一元素在所处命名空间之外可见。

见：**可见性**(3.1.228)。

相对语：**引入**(3.1.95)。

3.1.79

表达式　expression

计算某一特定类型的值的字符串。例如，表达式“(7＋5＊3)”计算“数”类型的值。

3.1.80

扩展　extend

从扩展用况到基用况的一种关系，它详述了为扩展用况定义的行为如何拓广(遵守在扩展中定义的条件)为基用况定义的行为。该扩展的行为被插入到基用况的扩展点处。基用况不依赖扩展用况的行为的执行。