ICS 25.040.40

中华人民共和国国家标准

GB/T XXXXX—XXXX/ISO/PAS 15450:2015

自动化系统与集成 对象过程方法

Automation systems and integration —Object-Process Methodology

(ISO/PAS 19450:2015,IDT)

（征求意见稿）

××××－××－××发布 ××××－××－××实施

国家市场监督管理总局
中国国家标准化管理委员会 发布

目 次

前　言

本标准按照GB/T 1.1—2009给出的规则起草。

本标准使用翻译法等同采用ISO/PAS 19450:2015《自动化系统和集成 对象过程方法》（英文版）。

本标准的技术内容和组成结构与ISO/PAS 19450:2015《自动化系统和集成 对象过程方法》（英文版）相一致。

本标准由中国机械工业联合会提出。

本标准由全国自动化系统与集成标准化技术委员会（SAC/TC159）归口。

本标准主要起草单位：　　。

本部分主要起草人：

引　言

对象过程方法（OPM）是一种用于自动化系统建模和知识表达的紧凑型概念性方法、语言和方法论。OPM的应用范围可以从基本元素的简单组合系统到复杂、多学科和动态系统。OPM适用于借助信息技术和计算机技术工具来实施和提供支持。本标准明确地指定了OPM的语言和方法论以便为系统架构师、设计师和OPM兼容工具开发商建立一个能支持所有类型系统建模的共同基础。

OPM为同一模型提供了两种语义上同等模式的表达方式：图形式和文本式。一组层次化和相互关联的对象过程图（OPDs）构成了图形模型，而一组采用英语语言子集的自动生成的句子则构成了对象过程语言（OPL）所表达的文本模式。在一个图形化可视模型中，每个OPD都包含有以图形符号被描绘出来的OPM要素，有时还带有标签注释。OPD语法指定了管理这些图形要素之间约定的一致性和正确性方法。通过使用OPL，OPM以保留图形模型约束的方式为每一个OPD生成了相应的文本模式。鉴于OPL的语法和语义是英文自然语言的一个子集，域专家可以很容易地理解文本模型。

OPM符号支持具有形式化语法和语义系统的概念建模。这种形式通常作为基于模型的系统工程的基础，包括系统架构规划、工程设计、开发、生命周期支持、通信和演化。此外，OPM的这种独立于领域的性质使系统建模适用于整个科学、商业和工业团体以用于其特殊应用领域中制造以及其它工业和商业系统的开发、调查和分析，从而使公司能够将不同技能和能力融入到一个通用直观且形式化的框架中，并且提供互操作性。

OPM为系统施工、测试、集成和日常维护提供了一个公共视图，确保了多学科环境下工作的实施。此外，公司通过使用OPM可以改善其对系统功能的纵览、人员任务分配的灵活性以及管理异常和错误恢复。系统规范可为任何必要细节进行扩展，包括系统的功能、结构和行为方面。

OPM的一个特殊应用体现在技术标准的起草和编写。OPM有助于勾画一项标准的实施情况及识别和减少标准中的不足，从而显著提高后续草案的质量。使用OPM，即使作为一个系统基于模型的文本也可进行扩展以包含更多细节，这样基础模型就会一直保持其高度的形式化和一致性。

本标准为系统构造师和设计师提供了一个能够精确而有效地将其用于系统建模的基准。OPM工具供应商可将PAS作为一种形式化标准规范来创建软件工具以增强概念性建模。

本标准提供了符合扩展巴科斯范式（EBNF）语言语法规范的规范性文本表达式。所有要素都呈现在第5 条到第12条中，仅很少地涉及方法论。第13条呈现了与放大和展开相关的上下文管理机制。

本标准为展现OPM而使用了若干惯例。具体来说，文本中的宋体加粗字体和图表标题、表格标题以及文本标题中的斜体加粗字体、都对用于OPM的对象、过程、状态和关联标签的标签名称进行了区分。OPL所保留的单词是带有宋体加粗字体的逗号和句号的宋体常规字体。大多数图形同时包含有一个图形图像、OPD部分和一个等效文本，即OPL部分。鉴于这是一种语言规范，精确使用术语定义是非常重要的，且通常用法中的一些术语在使用OPM时具有特殊的意义。第B.6条解释了使用OPM的其它惯例。

附录A介绍了以EBNF形式表示的OPL形式化语法。

附录B介绍了OPM应用程序中通常使用的约定和模式。

附录C介绍了作为OPM模型的OPM的各个方面。

附录D总结了OPM的动态性和仿真功能。

自动化系统与集成 对象过程方法

1 范围

本标准对对象过程方法（OPM）进行详细说明，使从业者可以将对象过程方法（OPM）的概念、语义和语法概念模型作为一种建模范例和语言来建立不同细节程度的概念模型,并可使工具商能够提供应用建模产品以帮助那些从业者们。

尽管本标准介绍了对象过程方法所使用的一些例子以提高清晰度,但并未尝试要为对象过程方法的所有可能性应用提供一个完整性的参考。

2 术语与定义

下列提供的术语与定义适用于本文件。

2.1

抽象化 abstraction

降低细节和系统模型完整性（2.8）的程度以为能更好地理解。

2.2

受影响物 affectee

受到一个过程（2.58）事件影响的转换物（2.78），比如其状态（2.69）发生改变。

注：一个受影响物只能是一个有状态的对象（2.66）。一个无状态对象（2.67）只能被创建或被消耗，但不会受到影响。

2.3

代理 agent

一个人或一组人的使能者 (2.17)。

2.4

属性 attribute

表征一件非自身事物(2.76)的对象 (2.39)。

2.5

行为 behaviour

对象 (2.39) **转换** (2.77)，其产生于对一个对象过程方法 (2.43)模型的执行，包含了模型中的事物(2.76) 集合体和对象的关联 (2.36) 。

2.6

受益者 beneficiary

从系统运行 (2.46) 所获得功能值(2.82) 的<系统>利益相关者(2.65) 。

2.7

类 class

具有同等持久性 (2.50) 、重要性、归属价值及相同特征(2.21)和状态集 (2.69)的事物集合体(2.76)。

2.8

完整性 completeness

在模型中说明一个系统全部细节的<系统模型>范围。

2.9

条件关联 condition link

从一个对象(2.39) 或对象状态 (2.69) 到一个过程 (2.58) 的程序关联 (2.56) ，意味着一个程序约束。

2.10

被消耗物 consumee

一个过程 (2.58) 事件所消耗或消除的被转换物 (2.78) 。

2.11

上下文 context

由一个对象过程图 (2.41) 和相应对象过程语言 (2.42)文本所代表的一个对象过程方法(2.43)模型的<模型>部分。

2.12

控制关联 control link

带有附加控制语义的程序关联 (2.56) 。

2.13

控制修饰符 control modifier

修饰一个关联 (2.36) 符号以对其添加控制语义，使其成为一个控制关联 (2.12) 。

注：事件的控制修饰符符号为“e”（3.18），条件的控制修饰符符号为'c'。

2.14

区分属性 discriminating attribute

不同值 (2.81)辨别相应的特化关系的属性 (2.4)。

2.15

效果 effect

一个对象 (2.39) 状态(2.69)或一个属性 (2.4) 值 (2.81)的变化 。

注：一个效果只适用于一个有状态对象 (2.66).

2.16

要素 element

事物 (2.76) 或关联 (2.36)。

2.17

使能器 enabler

能使能一个过程 (2.58) 但过程不会进行转换的<过程>对象 (2.39)。

2.18

事件 event

在一个对象的创建（或出现）或一个对象 (2.39) 进入到一个特定状态(2.69) 时的<OPM>点，其中任何一个可能会启动对过程 (2.58) 前置条件(2.53) 的评估 。

2.19

事件关联 condition link

代表一个事件(2.18)从对象 (2.39) 或对象状态 (2.69)到一个过程 (2.58) 的控制关联 (2.12)。

2.20

展示物 exhibitor

通过展示-表征关系而展现（表现为）一个特性 (2.21) 的事物 (2.76)。

2.21

特性 feature

属性 (2.4) 或操作 (2.46) 。

2.22

折叠 folding

通过隐藏一个展开的细化物 (2.62) 的可细化内容 (2.61)而实现的的抽象 (2.1) 机制。

注1：四种折叠的可细化内容是部件（部件折叠）、特征（2.21）（特征折叠）、特化（特化折叠）和实例（2.28）（实例折叠）。

注2：折叠主要应用于对象（2.39）。当应用于一个过程时，其子过程是无序的，适用于建模异步系统，其中，过程 的时间顺序未被定义。

注3：折叠的反义是展开（2.80）。

2.23

功能 function

将功能值（2.82）提供给一个受益者（2.6）的过程（2.58）。

2.24

一般类 general

具有特化的 <OPM> 可细化物 (2.61) 。

2.25

信息化的 informatical

信息的或与信息学有关的，例如数据、信息和知识。

2.26

继承性 inheritance

将一个一般类 (2.24) 的对象过程方法 (2.43) 要素 (2.16) 分派到其特化中去 。

2.27

输入关联 input link

从对象(2.39) 源（输入）状态 (2.69) 到转换过程 (2.58)的关联 (2.36)。

2.28

实例 instance

<模型>对象 (2.39) 实例或过程 (2.58)实例，它在类化-实例化关系中是一个细化物(2.62)。

2.29

实例 instance

<操作>对象 (2.39) 实例或过程 (2.58) 实例，它是一个存在于模型运行 期间 (2.46) ，如模拟或运行实施期间实际意义上的唯一可识别事物 (2.76)。

注：一个过程实例是可被过程事件中所涉及对象集 (2.32) 的操作实例以及过程时间的过程开始和结束时间戳进行辨别的。

2.30

仪器 instrument

非人类使能器 (2.17)。

2.31

调用 invocation

一个过程的启动一个过程((2.58) 的<过程>。

2.32

参与对象集 involved object set

前置过程对象集 (2.54) 和后继过程对象集 (2.54) 的结合。

2.33

放大场景 in-zoom context

置于被放大事物边界内的事物 (2.36)和关联（2.36）(2.76) 。

2.34

放大 in-zooming

<过程>对象(2.39)部分展开(2.80) ，表明构成对象的空间排序 。

2.35

放大 in-zooming

<过程> 过程 (2.80) 部分展开(2.58)，表明构成过程的时间偏序 。

2.36

关联 link

两个对象过程方法 (2.43)事物 (2.76) 之间的结构关系 (2.73) 或程序关系(2.57) 的图形表达 。

2.37

元模型 metamodel

一种建模语言或一种建模语言某部分的模型。

2.38

模型事实 model fact

对象过程方法模型中两个对象过程方法 (2.43)事物 (2.76)或状态 (2.69)之间的关系。

2.39

对象 object

<OPM>模型要素 (2.16) ，表示一个事物 (2.76)物理上或信息上(2.25)存在或也许存在。

2.40

对象类 object class

具有相同结构 (2.74)和转换 (2.77)模式的对象 (2.39) 模式 。

2.41

对象过程图（OPD） Object-Process Diagram OPD

一个对象过程方法模型或一个对象过程方法 (2.43) 模型某部分的图形表达,其中置身于利益世界中的对象 (2.39)和过程 (2.58) 与结构关联 (2.72)和程序关联 (2.56)共同出现。

2.42

对象过程语言（OPL ） Object-Process Language OPL

英语自然语言子集，以文本方式表示对象过程图(2.42)的图形形式所代表的对象过程方法(2.43)）模型。

2.43

对象过程方法 OPM Object-Process Methodology OPM

用于描述复杂和多学科系统的形式化语言和方法，采用单一功能-结构-行为统一的模型，使用双模态图形-文本方式表示系统和其转换(2.77) 或过程 (2.58) 所使用的对象 (2.39)。

2.44

OPD 对象树 OPD object tree

树形图，其根部为一个对象 (2.39)，通过细化 (2.63)详尽描述了对象 。

2.45

OPD 过程树 OPD process tree

树型图的根是系统图 (2.75) ，每个节点是一个对象过程图 (2.42)，它获自于其祖先的对象过程图中（或系统图）的一个放大 (2.35) 过程 (2.58) ，且每个有向边从父对象过程图中的细化过程指向子对象过程图中的相同过程。

注：对象过程方法 (2.43) 模型的阐释通常是通过过程分解经放大而发生，因此OPD过程树是对一个对象过程方法模型进行导航的主要方式。

2.46

操作 operation

一个事物 (2.76) 所执行的过程 (2.58)，其展现了该事物而非其本身的特性。

2.47

输出关联 output link

从一个对象 (2.39) 的转换过程 (2.58) 到输出（目的地）状态 (2.69) 的关联 (2.36)。

2.48

缩小 out-zooming

对象 (2.39) 放大 (2.34) 的反向<对象>。

2.49

缩小 out-zooming

过程 (2.58) 放大 (2.35)的反向 <过程>。

2.50

持久性 perseverance

事物 (2.76) 的属性 (2.60) ，可以是定义了一个对象 (2.39) 的静态（3.76）或定义了一个过程 (2.58) 的动态。

2.51

后置条件 postcondition

过程条件，是成功完成过程(2.58) 的结果。

2.52

后过程对象集 postprocess object set

过程 (2.58) 完成所保留下来的或产生的对象 (2.39)集合体。

注：后过程对象集可包含有状态对象(2.66)，特定状态（2.69）产生由过程性能而致。

2.53

前置条件 precondition

启动一个过程 (2.58) 的<过程>条件。

2.54

前过程对象集 preprocess object set

启动一个过程 (2.58)之前进行评估的对象 (2.39)集合体。

注：对象集合体可包含有状态对象(2.66) ，具体状态(2.69) 对于过程性能是有必要的。

2.55

首要本质 primary essence

一个系统中大多数事物(2.76)的<系统>本质，可以是信息性的(2.25)或物理性的。

2.56

程序关联 procedural link

对象过程方法 (2.43) 中程序关系(2.57)的图形符号（3.43）。

2.57

程序关系 procedural relation

一个对象 (2.39) 或对象状态 (2.69)与一个过程 (2.58)之间的连接或关联 。

注 1：程序关系明确了系统如何运作以实现其功能 (2.23)，指定了对象转换过程的时间依赖或有条件性的启动。

注 2：一个调用 (2.31)或异常关联 (2.36)意味着两个过程之间的执行控制流中的瞬间对象。

2.58

过程 process

系统中的一个或多个对象(2.39) 的转换 (2.77)。

2.59

过程类 process class

执行相同对象(2.39) 转换 (2.77) 模式的过程 (2.58) 模式。

2.60

特性 property

一个特定类型的全部元素 (2.16) 的通用建模注释，用于区分该元素。

注 1：基数约束、路径标签和结构连接 (2.72)标签是常见的特性附注。

注 2：与属性 (2.4) 不同的是，一个特性值在模型模拟或操作实施过程中可能不会发生变化。每一种要素都拥有一组属于自我的特性。

注 3：特性是对象过程方法（2.43）元模型（2.37））中一个要素的属性。

2.61

可细化物 refineable

可被细化(2.63)的< OPM >事物 (2.76)，可以是一个整体(2.83)、一个展示物(2.20)、一个一般类(2.24)或一个类(2.7)。

2.62

细化物 refinee

细化一个可细化物（2.61）的事物（2.76），可以是一个部件、一种特征（2.21）、一个特化或一个实例(2.29) 。

注： 四种细化物的每一种都有一个相应的可细化物（部分-整体、特征-展示物、特化-泛化、实例-类）。

2.63

细化 refinement

<模型>的详尽阐述，用来增加细节范围和随后模型的完整性 (2.8) 。

2.64

结果物 resultee

一个过程 (2.58) 事件所创建的被转换物 (2.78)。

2.65

利益相关者 stakeholder

对正在考虑、开发或部署的系统感兴趣或可能会受到系统影响的< OPM >的个人、组织或群体。

2.66

有状态对象 stateful object

具有特定状态 (2.69) 的对象 (2.39) 。

2.67

无状态对象 stateless object

缺少特定状态 (2.69)的对象 (2.39)。

2.68

状态 state

一个对象 (2.39) 的<对象>可能情况或位置 。

注：在对象过程方法 (2.43)中并无过程 (2.58)状态的概念，如一个模型中的 “启动”、“运行中” 或“完成”。作为替代，对象过程方法对子过程进行表示和建模，如启动、加工处理或完成。也请参见附件C中有关对象过程方法过程元模型的讨论。

2.69

状态 state

在某一时间点所抓拍的系统模型的 <系统> 快照,显示了在进行抓拍快照的时间点内正在执行的所有现有的对象 (2.39) 实例、每一个有状态对象 (2.66) 实例的当前状态以及过程 (2.58) 实例。

2.70

状态表达 state expression

细化 (2.63) ，包含显示了一个状态(2.69)对象(2.39)集合的任何适当的子集。

2.71

状态抑制 state suppression

涉及到隐藏一个状态（2.69）对象(2.39)集合的任何适当子集的抽象化（2.1）。

2.72

结构关联 structural link

对象过程方法 (2.43)中结构关系(2.73)的图形符号。

2.73

结构关系 structural relation

运行时不变的事物之间的联系或关联 。

注：结构关系将会在系统中持续至少一段时间间隔。他们提供了系统结构，且不伴有取决于时间的条件。

2.74

结构 structure

一个对象过程方法 (2.43) 模型中 <OPM>的对象(2.39) 集合体以及它们之间非短暂性关系或关联。

2.75

系统图 System Diagram SD

拥有一个系统性过程 (2.58) 的对象过程图 (2.41) ，显示系统功能 (2.23) 和与该功能连接的对象 (2.39)，为系统的顶层视图描绘了整个情景 (2.11)。

注：系统图是OPD过程树(2.45)的根，在所描绘的整个上下文之外没有详细范围，即不存在被放大的细化物(2.62)。除系统图之外的任何对象过程图 在细化(2.63) 产生的OPD过程树中都是一个结点。

2.76

事物 thing

<OPM>对象（2.39）或过程（2.58）。

2.77

转换 transformation

一个对象 (2.39) 的创建（生成、构建）或消耗（消除、破坏）或一个对象状态中 (2.69)中的变化。

注：只有一个过程 (2.58)可以执行转换。

2.78

被转换物 transformee

一个过程 (2.58) 转换（创建、消耗或影响）的对象 (2.39)。

2.79

转换关联 transforming link

消耗连接、效果关联或结果关联。

2.80

展开 unfolding

细化(2.63)，详细阐述了一个包含其它事物(2.76)以及它们之间的关联 (2.36)的细化物(2.62) 的额外细节。

注 1：四种展开为部件展开、特征展开、特化展开和实例展开。

注 2：展开主要应用于揭示有关展开对象细节的对象 (2.39) 。

2.81

值 value

一个属性 (2.4) 的<属性>状态 (2.69)。

2.82

值 value

系统功能 (2.23) 提供的<功能性>成本效益。

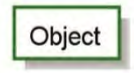
2.83

整体 whole

包含有两个或更多部分的聚合事物 (2.76)，每个部分都具有与聚合相同的持久性 (2.50)。

3 符号 symbols

——对象 object



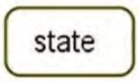
——物理对象 physical object

——环境对象 environmental object

——过程 process

——物理过程 physical process

——环境过程 environmental process



——状态 state

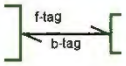

——聚合-分散 aggregation-participation

——展示-表征 exhibition-characterization

——泛化-特化 generalization-specialization

——类化-实例化 classification-instantiation

——单向带标签结构关联 unidirectional tagged structural link



——双向带标签结构关联 bidirectional tagged structural link

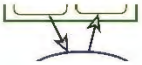
——代理关联 agent link

——仪器关联 instrument link

——效果关联 effect link

——消耗关联 consumption link

——结果关联 result link

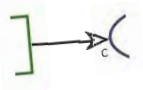


——输入-输出关联对 input-output link pair

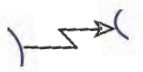
——仪器事件关联 instrument event link

——消耗事件关联 consumption event link

——仪器条件关联 instrumental condition link

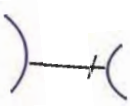


——消费条件关联 consumption condition link

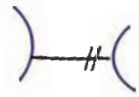


——调用关联 invocation link

——自调用关联 self-invocation link



——超时异常关联 over-time exception link



——时间不足异常关联 under-time exception link

4 一致性

可以预计工具制造商及终端用户很可能随时间推移对本标准的实施和使用呈增长趋势，所以应考虑几种适当的一致性标准：

a) 部分（符号）符合 OPM 应采用 OPM 的语言部分，即 OPM 语义和语法，通过：
   1) 只使用第 3 条中所定义的具有本标准中分派给它们的含义的 OPM 符号；以及，
   2) 只使用第 6 到 11 条所定义的具有本标准中分派给它们的含义的 OPM 要素。
b) 完全符合 OPM 应具备：
   1) 符合 a）； 并且，
   2) 符合 OPM 建模系统的方法和规划，如第 5 条和 13 条 所定义的。
c) 工具制造商达到一致性应具备：
   1) 符合 a）；
   2) 为 b）做好准备：在 a)形式基础上引导和帮助用户遵循 b）；以及，
   3) 根据附件 A 中所明确的 EBNF 定义为 OPL 提供支持。

5 对象过程方法（OPM）原理和概念

5.1 对象过程方法(OPM)建模原理

5.1.1 建模作为一种服务于目的的活动

系统功能和建模目的应引导一个OPM模型细节的范围和程度。一种复杂难于处理的系统可能会涉及到许多利益相关者，包括受益人、所有者、使用者和监管者以及许多硬件及软件组件，所以应将相关的各个方面展现给每个利益相关者。而一般利益相关者，特别是受益者所能发挥的作用或产生的效益预期应该用来辨别和指定建模目的，然后可借此来确定系统模型的范围。

示例：对于一个生产小型部件的制造工厂而言，关心供给率和日期的营销经理的视角并不包含工厂内不会受营销过

程影响的用于制造小部件仪器的机械。然而，从维护经理的角度出发，机械一定会受到影响，因为它们在操作期间将遭受磨损并需要对机器进行维护，这两项工作都能防止它们受损并在其受到损伤时对其进行修理。因此，营销经理使用的OPM 工厂制造模型将与维护经理所用的模型会有很大的不同。

5.1.2 功能、结构和行为的统一

一个系统的OPM结构模型应是由结构关系所连接的物理和信息（逻辑）对象的组合。这些结构关系在系统的生命期间可能会被创建和遭到破坏。

一个系统OPM行为模型，也称为其动态模型，应随时间推移去反映作用于系统上的机制以对系统对象进行转换，即系统内部对象和/或环境对象，即系统外部对象。

系统结构和行为的结合使得该系统能够执行一种功能，其应将系统（功能）值传递给系统受益者中的至少一个利益相关者。一个OPM模型将系统的功能（实用）、结构（静态）及行为（动态）这些方面集成到一个单一的统一模型中去。从关注整个系统功能的角度来看，这一结构-行为的统一为了解利益相关的系统提供了一个可参考的连贯性单一框架，在遵循形式化语法的同时也强化了其直观性的理解。

5.1.3 识 别 功 能 值

提供有一个建模系统过程的功能值应按照系统的主要受益人或受益组织的期望来表达系统功能。对于这种主要过程，即系统功能进行识别和标记在构建一个遵循OPM方法论描述的OPM模型过程中是关键性的第一步。一个恰当的功能标签或名称应澄清并强调系统能够为其主要受益者提供建模系统的中心目标和功能值。使用OPM进行建模应从定义、命名及描绘系统的功能开始来作为其首要过程。

注：这种常常在早期阶段就能引起系统架构团队成员之间的争论是极为有用的，因为它暴露了参与者之间对其将准备架构、建模和设计的系统的分歧，甚至是误解。

当系统功能与其主要受益者的功能值的预期达成一致后，建模者应识别其它主要利益相关者的期望并将其添加到OPM模型上去。

5.1.4 功能与行为

功能值对于受益人来说通常是被隐式地表达在过程当中的，其所强调的是发生了什么的这种行为而非主要过程所发生的功能值的这种目的。建模者应将功能和行为区分开来以创建一个清晰而明确的系统模型。这种区分十分重要，因为在许多情况下，一个系统的功能由不同的概念来实现，各自去执行一个不同的设计，且表现各异。

示例：想象一个系统可使人类使用其车辆来过河，这包含有两个明显的概念，一个是使车辆跨越的静态结构和一个运载车辆的动态移动元素。相应系统的设计是一座桥梁和一个渡口。尽管过河的功能和主要过程对于这两种设计来说是相同的，但它们在其结构和行为中有着明显的不同。

不承认功能与行为之间的差异可能会导致过早地选择一个次优设计。如上面的例子，这也许会导致做出一个建造一座桥梁而根本不考虑更高质量渡口的决定。

5.1.5 系统边界设置

系统环境应是事物的集合体，其置于系统之外，但可与系统进行交互，也有可能改变系统及其环境。建模者应对这些环境事物进行区分，哪些是系统部分，哪些不属于系统部分。即使那些环境事物可能会影响系统或被系统影响，建模者是无法对环境事物的结构和行为进行构造、设计或操纵的。

5.1.6 清晰度和完整性的折中

现实生活中内在存在众多细节和复杂性。要想了解这种系统就牵涉到一个应如何在清晰度和完整性这两种相互矛盾的准则之间达到平衡的一个折中问题。清晰度应是能明确地理解系统结构和行为模型所

传达的程度。完整性应是系统全部细节的规范程度。这两种模型属性是相互矛盾的。一方面，完整性要求完全遵守系统的细节。而另一方面，对清晰度的需求则是给一个单个模型图中的细节程度强加了一个上限，超过此上限后，理解度将因杂乱无章和拥挤而变得每况愈下。

确立一个恰当的平衡需要在模型开发过程中对上下文进行认真的管理。在一个特定模型图中增加完整性的表达常常会导致清晰度的降低。然而，建模者可以利用由整体OPM系统模型所提供的信息集合的优势拥有一个清楚且明确无误但并不完整的图，而另一个图则专注于具有更多细节的系统某些部分的完整性。

5.2 对象过程方法（OPM）基本概念

5.2.1 双模态表示

一个OPM模型应是在语义上具有等效意义的效图形和文字表达式中的双模态。每个OPM模型图形图，如一个OPD图，应拥有一个包含了使用OPL的一个或多个OPM的语言句子的等效OPM文本段落。

注1：OPM 模型的双模态图形-文本表达方式有助于非技术性利益相关者参与到开发阶段对系统需求的献计献策和初始概念的建模中来。这种参与使得这些利益相关者成为积极的参与者并在他们不经意间的介绍后能很快地发现错误。双模态表达式也能帮助新的 OPM 用户在检测文字和相应图形时快速地熟悉 OPM 图形方式的语义。

注2：附录 A 规定了使用 ISO/ IEC 14977：1996 的 OPL 语法：1996。

注3：对于在本标准中所使用的大多数 OPD 数字来说，OPL 句子的相应段落与图形 OPD 相伴随。

5.2.2 对象过程方法（OPM）建模要素

要素，即OPM中任何建模系统的基本组成部分，应分为两种：事物和关联。对象和过程的建模要素应将事物指定在模型的上下文中。关联的建模要素应在模式上下文中的事物之间指定关联。对象应为无状态或有对象状态。关联应是程序性或结构性。图1提供了一个OPM元模型的概观。

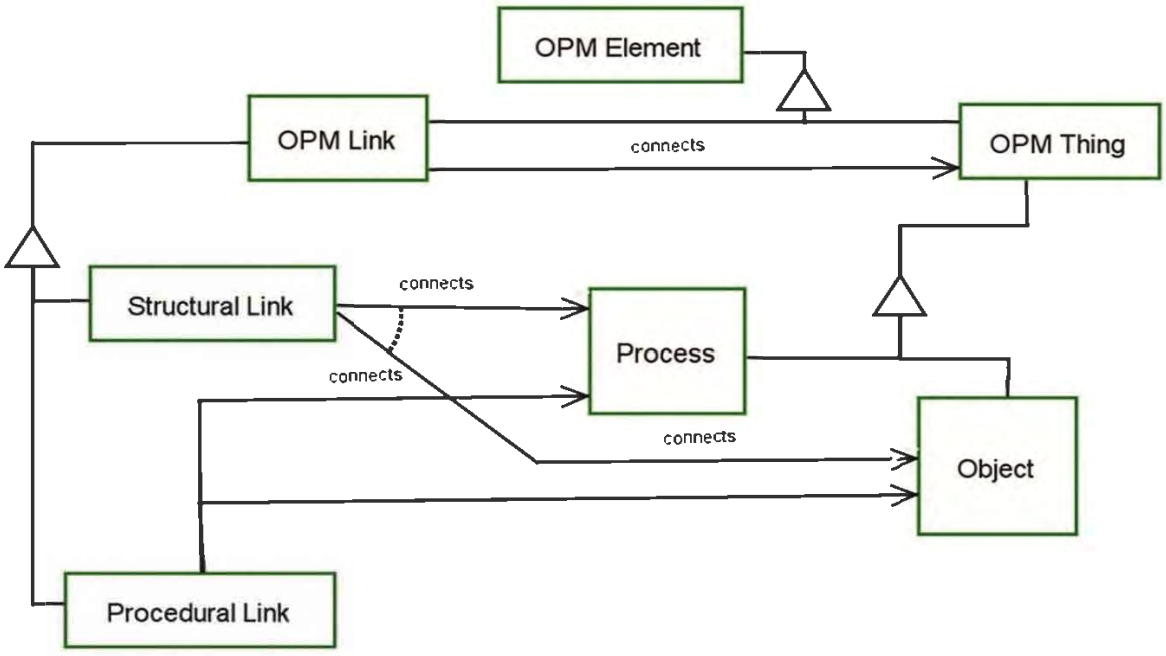


图1 对象过程方法（OPM） 元模型概观

在一个OPM模型内，建模要素应具有独特的符号、文字表达、语法约束和语义阐释。一个OPM模式中的每一个被建模事物都应具有与利益相关者相关的唯一标识名称，并且，独特的源和目标事物应对每个关联或标签关联进行区分。一个建模关联应与其来源和目标事物同为一个具有相应OPL句子的OPM结构。

一旦被识别，一个建模事物就可能会出现在任何与该事物相关的上下文中，且可能会不止一次地出现在上下文中以增强理解。

5.2.3 对象过程方法（OPM） 事物：对象和过程

一个对象应是一个事物，其一旦构成就可以物理形式或信息形式存在。对象之间的关联应构成建模系统的对象结构，即该系统的静态结构方面。一个对象的状态应是一个对象在其生命周期内某个时间点上的一个特定情况的类化。在每个时间点上，一个具有对象状态的对象则处于其某一状态中，或由于一个目前影响该对象的过程结果而处于其状态的其中两种状态之间的过渡。

一个过程应是表达系统中对象转换的事物。一个过程始终与一个或多个对象相关联或发生关系而非孤立存在。一个过程通过创建对象、消耗对象或者改变对象的状态来转换对象。因此，过程通过提供系统的动态和行为方面来补充对象。

注：确认一个检验时间点上哪些子过程正在运行的这种检查过程揭示了一个过程的状态。 OPM没有明确指定一个过程的模型状态。见附录C中的过程元模型。

5.2.4 对象过程方法（OPM）关联：程序和结构

程序关联应表示程序关系。一个程序关系应明确指定系统如何运作以实现其功能，指派转换对象的过程依赖于时间或条件的启动。

结构关联意味着结构关系。一个结构关系应指明在该系统中持续至少一些时间间隔的关联，即系统静态方面，而不应是那些依赖于时间的条件。

5.2.5 对象过程方法（OPM）上下文管理

OPM应为管理模型细节的上下文提供机制以提高理解和清晰度。建模者应从最初功能模型的上下文开始，使用对象结构细化和过程分解来扩展模型细节，每一个递增的细节扩展包含有一个上下文重点。

为实现系统功能，一组重要的过程应包括子过程的一个层次网络。过程层次应诱发过程上的一部分命令，即某些过程结束才可开始其它过程，而其它过程也可同时或作为替补而出现。在过程层次中的任何细节范围上，一个系统中的过程应提供或贡献函数值来作为其祖先过程的一部分。

上下文管理的基本单位是描绘该特定上下文建模要素的对象过程图（OPD）。新图展开和放大提供了上下文之间的结构和程序关联。尽管任何OPD均可包含任何数量的要素，但只有与特定上下文相关的那些要素才应出现在OPD中。

用于事物的名称和标签以及关联的管理上下文应是整个OPM模型，在该模型个，各自分离的模型段将那些产生行为的模型要素中的关系和交互放置于上下文中进行考虑。事物名称在该管理上下文中应是独一无二的。

5.2.6 对象过程方法（OPM） 模型实施

5.2.6.1 概念模型与运行模型

当使用OPM构造模型时，建模者需要了解他们正在创造的概念模型与它们将用来评估系统行为的那个运行模型之间的区别。建模实习生对于这种区别常常拥有一种直观感觉，当创造一个模型时会很急于

将其设想为建模的要素操作实例，即使当这些要素都处在一个非常抽象的状态时。然而，那些并不熟悉OPM所支持的建模类型的人可能会发现这个标准中的规范有些让人摸不着头脑。

一个OPM模型是一种形式框架，其中对象和过程事件皆通过关联手段进行交互。由于一个OPM模型所拥有的这种框架类似于系统结构，且模型要素使用关联进行交互，所以建模者可以通过创建对象和过程操作实例事件对系统行为进行模拟，然后遵循关联中所体现的执行控制流以及OPM语义规则。一个事物事件的发生将抽象的概念模型转换为一个更具体的操作模式。

附件D展示了支持仿真活动的OPM设施。然而，当本标准的用户构建OPM模型时，他们需要记住的是，建模系统行为只在这些事物的操作实例事件存在时才会回发生。两个事物之间出现的关联直到那些事物的操作实例事件存在才意味着行为。“运行”这个词，即运行实例出现确实存在时，在此所提供的每一个规范声明中均为隐式。

注：“实例”这个词也会在类化-实例化关系出现时具有不同的含义。在该用法中，一个实例是类的一个典型的细化物。

5.2.6.2 OPM 模型的实现

OPM概念性框架包括模型的仿真能力。要想成功地利用这种能力，一个建模者需要了解一个作为代表结构和行为的模型与一个执行以该模型为模式功能的运行模型实例之间的区别。模型具有一种架构形式，部分基于结构和程序的排列，建模者随着模型设计的演化使用细节将其进行扩展。一个表达了一致性细节的模型是可以用来作为一个能够实现资源、使用过程来转换对象以及为受益者产生功能值的仿真模型的。

5.2.6.3 OPD 导航和 OPL 构成结构

本标准呈现了创建OPM模型图和相应OPL文本的手段。第13条的放大和展开机制提供了将OPD图形与相应OPL进行相连的方法以表达作为文本的关联性。然而，由于有许多方式将这些关联做上标签，而其中有些方式可能针对于一个工具的实施，所以第13条并没有明确如何分派鉴别连续性层次等级之间、相关OPD图之间或相应OPL段之间的标签。

6 对象过程方法（OPM）事物语法和语义

6.1 对象

6.1.1 描述

一个对象应是一个存在的事物或具有物理或信息化存在潜能的事物。从时间观点来看，一个事物的存在应是持久的。只要没有过程作用于事物，它就应保持其目前的隐式或显式状态。

一个OPM对象对于一个结构模式、属性和特性，即属性和操作来说是一种抽象的适用于该类别的操作实例对象类别标识符。在模型约束范围内，任何对象操作实例的非负数数字都有可能存在。

6.1.2 表达方式

一个包含有标签和对象名称的矩形盒子应意味着一个模型对象以图形形式的存在。图2以图形方式显示了对象车辆乘务组。在OPL文本中，对象名称应以每个单词大写的粗体形式出现。

命名对象常规方式在 B.6.2中进行了讨论。

图2　对象图形符号

6.2　过程

6.2.1　描述

一个过程应是转换一个或多个对象的事物。转换可以是生成（构建，创建）、影响或消耗（销毁、消除）。一个过程应具有正的性能持续时间。

一个OPM过程是用于转换一种模式的抽象类别标识符。为实现具体的操作实例，一个过程实例是该类别所指定的一个过程模式的特定事件。过程操作实例将一个或多个对象操作实例进行转换。

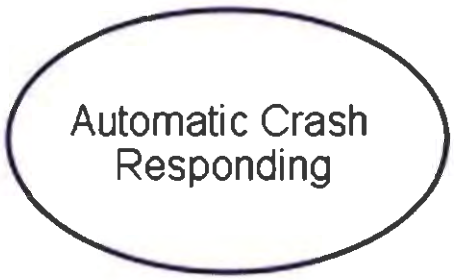
注1：一个过程可通过调用关联（见 8.5.2.5.2）直接调用另一个过程，调用关联会导致调用过程创建一个已调用过程即刻消耗的暂时性对象。

注2：一个过程对一个对象产生的效果通常是该对象的状态发生了变化。然而，也有那种其效果处于状态维护的持久性过程。一个持久性过程的语义是要维持处于其当前状态中的对象，而非去诱发一种变化。

示例：过程存在是最突出的持久性过程；它描述了存在的一种静态（隐式）状态。其他持久性过程的例子还包括持有、维护、保管、停留、等待、延长、扩展、延迟，占有、坚持、继续、支持、扣留和保持。对于生物对象而言，存在牵涉到生活 – 积极维持必要的生命过程。

6.2.2　表达方式

一个含有标签，即过程名称的椭圆形应意味着以图形形式出现的抽象过程类别。图3以图形形式说明了过程的自动碰撞响应。在OPL文本中，过程名称应以每个大写单词的粗体形式出现。



命名过程的常规方式在 B.6.3中进行了谈论。

图3　过程图形符号

6.3　对象过程方法（OPM） 事物

6.3.1 定义对象过程方法（OPM）事物

一个OPM事物应是一个对象或过程。对象和过程在许多方面是对称的，就关系而言具有很多共同点，如聚合、泛化和表征。一个对象在一个过程对一个或多个对象产生行为时就存在了。OPM对象和OPM过程在一个过程有必要转换一个对象这种意义上来说是互相依赖的，而对于一个过程的出现或发生来说，至少有必要对一个对象进行转换。

6.3.2 对象过程测试

为了以一种有效方式应用OPM，建模者需要将对象与过程之间进行一种本质上的区别以作为成功进行系统分析和设计的先决条件。默认情况下，一个名词应识别一个对象。对象过程的测试为建模者提供了区分过程名词和对象名词的准则。为有关一个特定名词是否为一个对象还是一个过程这种问题提供正确答案对于对象过程方法（OPM）来说是最基本也是非常关键的。

要想成为一个过程，一个名词或名词短语应满足以下三个过程标准的每一个准则：

——时间关联，所涉及的名词与时间的推移有关联；

——动词关联，所涉及的动词衍生自一个动词或与一个动词有同根，或拥有与一个动词相关的同义词；以及

——对象转换，所涉及的名词出现、发生、实施、执行、转换，改变或修改至少一个对象或将其保留在其当前状态中。

示例：飞行是一个过程的名词，因为它通过了三个对象过程的测试标准：

1)有一个时间关联；

2) 有与飞行这个动词的关联；以及

3) 通过改变其从源到目的地的位置属性值，它将飞机进行了转换。

6.3.3 对象过程方法（OPM）事物类属属性

OPM的所有事物都应具备以下三种类属属性：

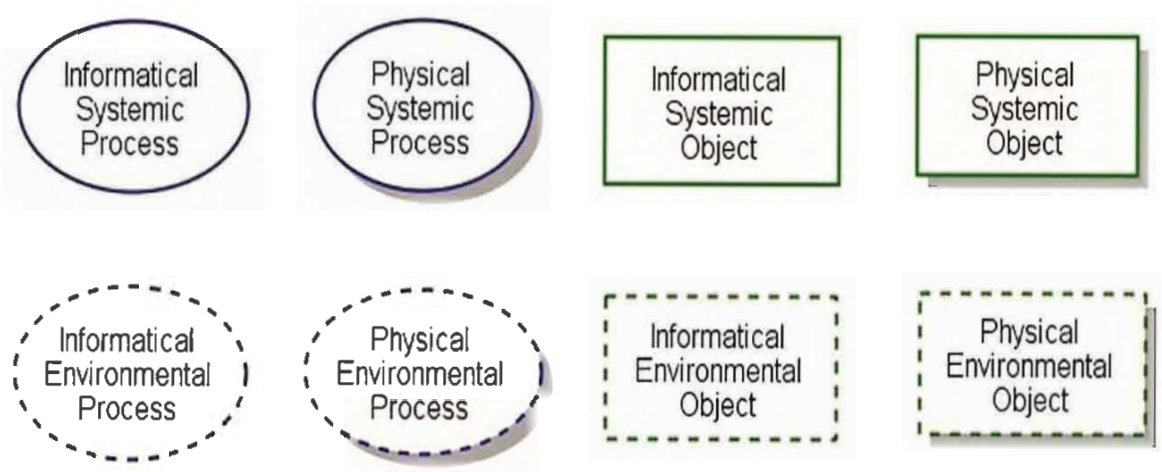
——韧性，涉及到事物的持久性，意味着事物是静态，即一个对象，或是动态，即一个过程。因此，韧性属性的可允许值为静态或动态；

——本质，涉及到事物的性质，表示事情是物理性还是信息性的。因此，本质类属属性的可允许值为物理性的或信息性的；

——隶属关系，涉及到事物范围并表示事物是否是系统性的，即，系统的一部分，或环境性的，即该系统环境的一部分。因此，隶属关系的属性值为系统性的或环境性的。

注：尽管对象是持久性的，也就是说，它们有静态的韧性，而过程是暂时性的，也就是说，它们具有动态的韧性，持久性过程（见6.2.1）和暂时性对象（见8.5.2.5.1）的边界范例是有存在的可能性的。

从图形上看，如图4所示，阴影效果应指的是物理OPM事物，而虚线应指的是OPM环境事物。图4中所示的一个OPM事物的全部八个韧性-本质-隶属关系的类属属性组合均有可能出现。图4下部分从左至右、从上到下表示了对应于图形要素的对象过程语言（OPL）句子。



信息系统过程是一个信息和系统过程；

物理系统过程是一个物理和系统过程；

信息系统对象是一个信息和系统对象；

物理系统对象是一个物理和系统对象；

信息环境过程是一个信息和环境过程；

物理环境过程是一个物理和环境过程；

信息环境对象是一个信息和环境对象；

物理环境对象是一个物理和环境对象。

图4　OPM 事物类属属性组合

6.3.4　事物类属属性默认值

一个事物的隶属关系类属属性的默认值应是系统性的。

任何具有重要意义的系统大部分都趋向于拥有本质上相同事物类属属性值的对象和过程。

示例：数据处理系统是信息性的，虽然他们具有物理组件。一个运输系统，如铁路系统或航空系统，是物理性的，虽然它们具有信息性组件。

一个系统的主要本质应与该系统边界内大多数事物本质的值相同。

系统边界内的一个事物的本质类属属性的默认值应是系统的主要本质。

注：一个辅助工具可为建模者提供一种选择性，即将一个系统的主要本质指定为建立默认事物本质类属属性值的一种手段。

对应一个图表的OPL不应反映事物类属属性的默认值，除非事物还未连接到另外一个事物中去，比如建模过程期间。只要与其它事物的关联一出现，事物类属属性就应适当地并入到描述这些关联的OPL短语中去。

6.3.5　对象状态

6.3.5.1　有状态和无状态对象

对象状态应是一个对象也可在其中存在的可能性状况。一个对象状态只在其所属的对象，即所拥有状态对象的上下文中才具有意义。

一个无状态的对象应是一个无状态特化的对象。

一个有状态对象应是一个具有一组特定可允许状态的对象。在一个运行的模型中、任何时间点上的任何有状态对象的操作实例均处于一个特定的可允许状态，或作为一个目前影响着该对象的过程结果存在于两个可允许状态之间的过渡期间。

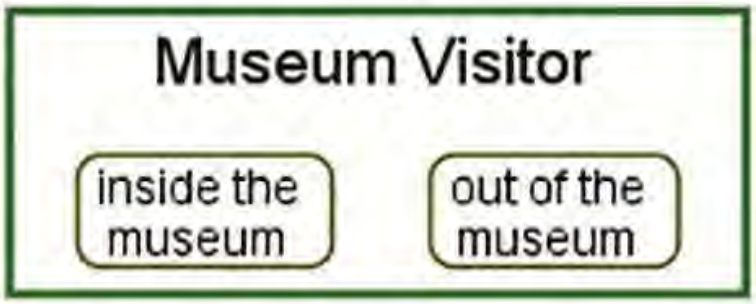
注 1：根据不同模型行为，一个对象的运行实例可处于不同状态。

注：注2：命名对象状态常规在B.6.4中进行了讨论。

6.3.5.2 对象状态表达

就图形来说，一个放置于其所属对象内的标签和一个圆角矩形（a 'rountangle'）意味着一个对象状态。在OPL文本中，对象状态标签应以小写的粗体字出现。

示例：图 5 描述了具有博物馆内部和博物馆外部所标记的两种状态的对象博物馆参观者。下面的图形表达式是相应的 OPL 句子。



博物馆参观者可以在博物馆内或博物馆外。

图5 具有两种状态的有状态对象

6.3.5.3 初始、默认和最终状态

一个对象的初始状态应是其在系统开始运行或系统运行期间所产生的状态。一个对象的最终状态应是其在系统完成操作或在运行期间被系统消耗时的状态。一个对象的默认状态应是其在对象最有可能处于随机检查期间的这种状态 。

注1：一个对象可以拥有零或多个初始状态、零或多个最终状态以及零或一个默认状态。相同状态可以是任何初始、最终和/或默认的组合。

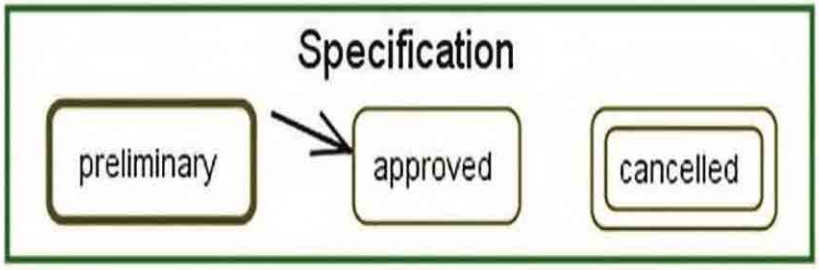
注2：初始和最终状态对于展示一个生命周期模式，例如产品或生物体或系统的对象是特别有用的。

注3：如果一个对象具有一个以上的初始状态，那么就有可能把该状态中正被创建的对象的概率分派至每一个初始状态（见 11.7）。

6.3.5.4 初始、默认和最终状态表达式

从图形上来说，一个厚实的轮廓边界应表示一个初始状态，双边轮廓边界应表示最终状态以及一个左侧斜线所指的开放箭头应表示一个默认状态。相应的 OPL 句子使状态规范变得清晰明了。

示例：图 6 描绘了带有初始、默认和最终状态的对象规范。下面的图形表达式是相应的 OPL 句子。



规范的初步状态是初始状态；

规范的批准状态是默认状态；

规范的取消状态是最终状态。

图6　一个具有初始、默认和最终状态的状态对象

6.3.5.5　属性值

由于一个属性是一个对象，一个属性值就应对应于一个从意义上来说就是一个属性状态的状态。一个对象可具有一个属于不同对象的属性，并且在展示该属性的对象存在期间的一些时间间隔内，该属性的值是具有不同对象的状态。

示例：如果将摄氏度中的温度作为一个发动机属性来考虑的话，那么 75 即是该属性的值。

注1：由于一个属性是一个有状态的对象，一个可允许的属性值是该有状态对象的可允许状态集的一个成员。一个枚举列表或一组的一个或多个数字范围可为该属性定义一组可允许值。

注2：相反，一个属性值是固定的，在模型操作期间不会改变。

测量单位中所表达的具有值的属性应在属性对象名称下面方括号内的OPD中以图形形式表达测量单位，并在相应OPL句子中的属性对象名称之后以文本形式表达测量单位，例如，摄氏度中的温度。

7　对象过程方法（OPM）关联语法和语义概述

7.1　程序关联概述

7.1.1　程序关联类型

一个程序关联应是以下三种类型之一：

——转换关联，其将一个被转换物（一个该过程所转换的对象）或其状态之一与一个过程进行连接以对对象转换进行建模，也就是由于过程性能所产生的结果来对该对象的生成、消耗或状态改变进行建模；

——使能关联，将一个使能器（一个能使过程发生但未被过程转换的对象），即代理或仪器或其状态与一个过程进行连接以为该过程的启用存在而建模；或

——控制关联，它是一个具有执行控制机制附加语义的一个转换关联或使能关联以便对一个启动关联过程的事件建模、对用于过程性能的条件建模或对两个意味着调用或异常的过程关联进行建模。

注：被转换物和使能器是一个对象在其所连接的过程中可以扮演的一种角色。因此，一个对象可拥有一个用于过程使能器的作用也可拥有另一个用于过程被转换物的作用。

7.1.2 对象过程方法（OPM）原理程序关联独特性

一个过程应将一个转换关联连接到至少一个对象或对象状态中去。在任何一个抽象化的特定范围内，一个对象或其状态的任何之一应仅拥有一个作为对应于其所连接过程的模型要素的角色：对象可以是一个被转换物、一个使能器、一个初始器或一个条件对象。在一个抽象化的特定范围内，一个对象或对象状态应只由一个程序关联连接到一个过程上去。

7.1.3 状态-指定程序关联

每个程序关联均有资格作为一个状态-指定程序关联。一个状态-指定程序关联应是一个将过程连接到对象指定状态的程序关联。

7.2 操作语义和执行控制流

7.2.1 事件- 条件 - 行动控制机制

事件- 条件 - 行动范式应提供OPM操作语义和执行控制流。在对象创建的时刻，或从系统角度来说对象的出现或一个对象进入到一个特定状态时，一个事件就会发生。在运行期间，对于一个带有过程关联源的对象来说，比如一个过程的使能器，一个事件的发生应启动对于作为关联源对象关联来说的每一个过程前置条件的评估。

当对一个过程的前置条件评估开始时，事件应为该过程停止存在。只有当评估表示对前提条件满意时，该过程才能启动操作且进行活动。

启动一个过程的操作具有两个前置条件：

a) 一个初始事件；及

b) 一个前置条件得到满足。

因此，事件和前置条件一同为过程运行指定OPM执行控制流。

注：调用和异常是仅在过程之间所发生的事件-条件-行动。

执行控制流应是以事件 - 条件 - 行动为排序的连续性结果，由外部事件启动系统功能开始，当系统功能完成时结束。

7.2.2 前过程对象集和后过程对象集

一个过程的前过程对象集应在该过程开始运行前确认需要满足的前置条件。前过程对象集可能是错综复杂的，包括有复合逻辑表达式，或可能只包含有一个或多个可能处于指定状态中的对象的存在。一个前过程对象集中的典型对象是被消耗物，即过程所消耗的对象，以及受影响物，即过程所影响到的对象以及过程使能器。这些对象的其中一些对象可能对执行控制流具有更进一步的规定，即一个条件关联。每个过程都应有拥有具有至少一个可能处于一个指定状态中的对象的前过程对象集。

后过程对象集应确定该过程需要满足的后置条件。后过程对象集可能是错综复杂的，包括复合逻辑表达式，或可能只包含有多个也许处于指定状态中的对象的其中之一对象的存在。一个后过程对象集中的典型对象是结果物，即过程所生成的对象以及受影响物，即过程影响到的对象。每个过程都应拥有具有至少一个可能处于一个指定状态中的对象的后过程对象集。

注1：相同过程的前过程对象集与后过程对象集的交集点包括过程使能器和受影响物。 被消耗物是前过程对象集的唯一成员，而结果物则是后过程对象集的唯一成员。

对于所涉及的对象集中的对象运行实例语义在 13.2.2.4.4 中进行了介绍。

7.2.3 跳过条件语义与等待非条件关联语义

一个过程的前过程对象集可包含条件关联（参见8.5.3）和非条件关联，即不含条件控制修饰符的程序关联。条件关联的独特方面在于它们 “跳过语义”，它规定如果在条件关联的源对象操作实例不存在的情况下可以跳过或绕过一个过程。如果没有条件关联限制，一个源对象操作实例的非存在性将会导致等待另一个事件和所有源对象的操作实例，或许在一种指定状态中，以此来满足前置条件。

如果有一个或多个非条件关联和一个或多个条件关联，它们的全部存在对满足前置条件和启动过程后都是有必要的。然而，如果有一个或多个未满足的非条件关联和一个或多个未满足的条件关联，那么前者的等待语义与后者的跳过语义之间将会产生一种冲突。为解决这种冲突，条件关联的跳过语义必须比其非条件同行的等待语义更强，并且执行控制流要跳过这个不启动其性能或生成异常的过程。

即使只有一个连接了过程的与条件关联相关的条件不存在，前置条件满意评估就将失败，执行控制会跳过该过程，并且一个事件会通过某种调用关联为下一个后续过程（一个或多个）而发生，见8.5.2.5和13.2.2。

注1：不存在结果事件关联或结果条件关联，因为这些涉及到后过程对象集将要离开的程序关联。当一个过程完成时，它创建的后过程对象集没有更进一步的条件，所以在结果物的创建或受影响物的改变上是没有条件的。创建一个很可能处于一个指定状态中的对象可在后过程对象集中用于一个后续过程的一个事件或条件。

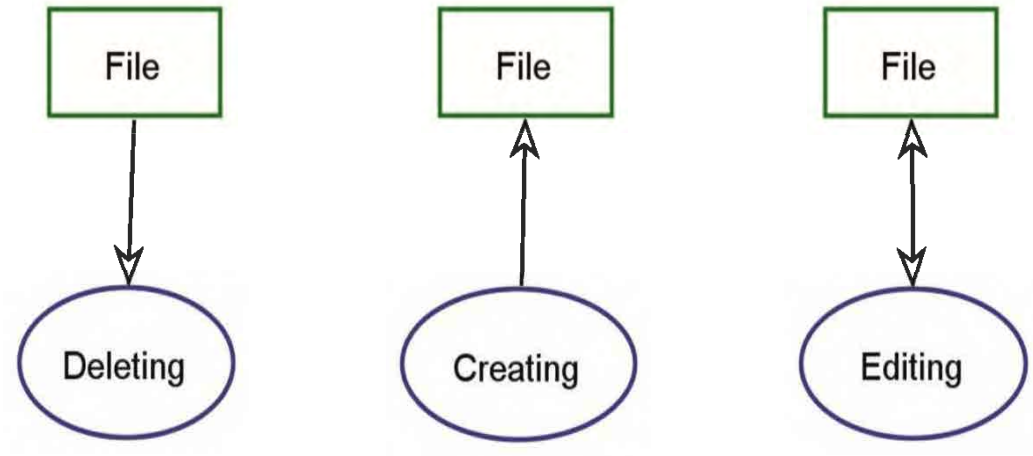
注2：为能实现在所有情况下更强大的执行控制流，建模人员可将未完成的提早结束的过程作为异常处理(见 8.5.4）来建模。

8 程序关联

8.1 转换关联

8.1.1 转换关联类型

一个转换关联应在一个过程与被转换物（它所消耗、创建或改变状态的对象）之间指定一个关联。转换关联的三种类型应是消耗关联、结果关联和效果关联。图7借助位于图形表达式下面的相应OPL句子说明了转换关联的三种类型。



删除毁灭文档。　　创建产生文档。　　编辑影响文档。

图7 转换关联：左边：销毁， 中间：结果 右边：效果

一个被转换物应是一个有关一个特定过程的对象的角色，相同对象对另一过程可具有不同的作用。

8.1.2 消耗关联

一个消耗关联应是一个转换关联，其指定了关联过程消耗（破坏，消除）的关联对象，即被消耗物。

从图形上来说，如图7所示，一个从被消耗物指向消耗过程的带有封闭箭头的箭状符号应表示消耗关联。

一个消耗关联OPL句子的语法应为：过程消耗被消耗物。

被消耗物的存在应是一个过程激活的前置条件或部分前置条件。如果被消耗物不存在，即没有被消耗物的操作实例存在的话，过程激活必须等到被消耗物存在。

一旦过程激活则消耗应立即开始，除非建模者需要一段时间将对象的消耗进行建模。在这种情况下，消耗关联应具有一个显示被消耗物的消耗率属性，被消耗物应具有一个能显示可用数量的属性。

注1：如果对象数量小于所预期的过程持续时间的比率时间，建模者则可以创建一个异常。

注2：见 10.1 有关关联属性的定位。

示例1：钢条对于过程机械加工是一个被消耗物，其产生了结果物轴。一旦机械加工开始，它就会消耗钢条。

示例2：水对于灌溉这个过程是一个被消耗物。被消耗物具有值为 1000 升的数量属性，消耗关联具有每秒值为 50 升流速的属性。在这种情况下，如果灌溉不间断的化，它会持续 20 秒，并以特定流速的值消耗水。

8.1.3 结果关联

一个结果关联应是一个转换关联，其指定关联过程创建（生成，产生）关联的对象，也就是结果物。

从图形上来说，如图7所示，一个从创建过程指向结果物的带有封闭箭头的箭状符号应表示其一个结果关联。

一个结果关联OPL句子的语法应为：过程加工产生结果物。

结果物应在过程完成后立即生成，除非建模者需要一定的时间对象进行建模。在这种情况下，结果关联应具有一个显示其结果物生成速率的属性，而结果物应具有一个显示可用数量的属性。

注：见10.1 用于关联属性的定位。

示例1：钢条是过程机械加工的一个被消耗物，其生成结果物轴。当机械加工完成后，就生成了轴。

示例2：汽油和柴油是过程提炼的结果物，其消耗原油。结果物汽油和柴油各有一个属性数量（立方米）。提炼汽油的结果关联具有值为 1000 的汽油产出率（立方米/小时），提炼柴油的结果关联具有值为 800 的柴油产出率（立方米/小时）。假设拥有足够的原油，如果提炼激活并运行 10 小时的话，它讲会产生 10000 立方米的汽油和 8000 立方米的原油。

8.1.4 效果关联

一个效果关联应是一个转换关联，其指定关联过程影响关联对象，也就是受影响物，即过程导致受影响物状态中的一些未指定的变化。

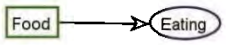
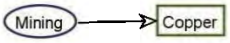
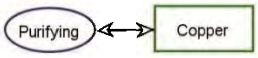
从图形上来说，如图7所示，一个带有两个封闭箭头的双向箭状符号，双方在影响过程之间互指向对方的方向，受影响的对象应表示效果关联。

一个效果关联OPL句子的语法应是：过程处理影响受影响物。

8.1.5 基本转换关联小结

表1 总结了基本转换关联。

表1　基本转换关联小结

名称	语义	OPD & OPL 例子	源	目的地
消耗关联	过程消耗对象。	 吃消耗食物。	被消耗的对象。	消耗过程
结果关联	过程生成对象。	 采矿生成铜。	创建过程	创建对象
效果关联	过程通过从一个状态改变到另一个状态来影响对象。	 提纯影响铜。	受影响对象和影响过程两者均为源和目的地	

8.2　使能关联

8.2.1　使能关联类型

一个使能关联应是一个为过程指定了一个使能器的程序关联。一个过程使能器应是一个为该过程发生的对象。一个使能器在过程完成之后的存在和状态应与过程刚开始运行之前相同。

两种使能关联应是代理关联和仪器关联。

使能器应在其启用的整个过程运行中存在。从系统角度来看，如果使能器在其启动过程操作期间停止存在的话，那个过程应立即结束。

注1：一个使能器扮演的是一个相对于一个特定过程的对象所扮演的角色。同样的对象可以是一个过程的使能器和另一个过程的被转换物。

注2：为实现在所有情况下强大的执行控制流，建模者可以将未完成的提早结束的过程作为一个异常处理（见8.5.4）来建模。

8.2.2　代理和代理关联

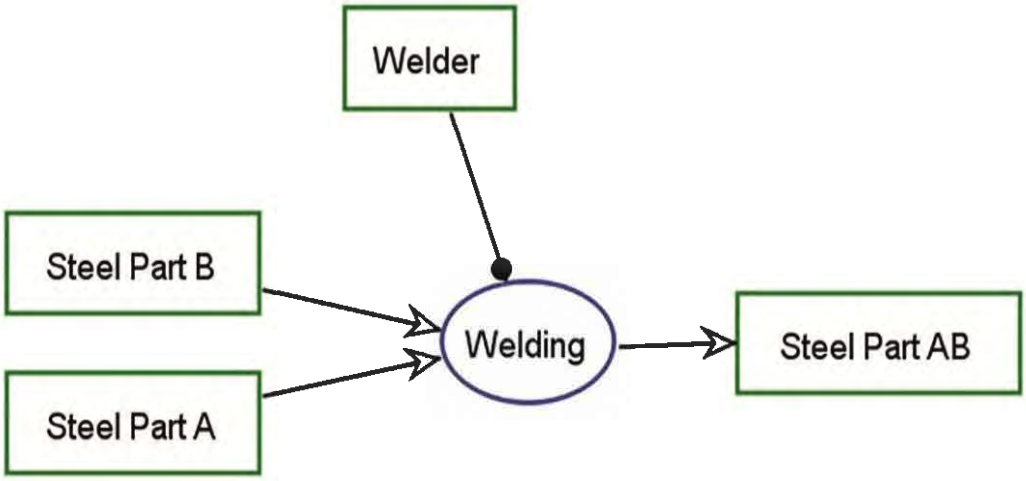
一个代理应是一个或一组能够进行智能决策的人们，其可与系统交互以能在整个过程运行期间启用或控制过程。

一个代理关联应是一个从代理对象到其启用过程的使能关联，指定了代理对象对于关联过程激活和运行的必要性。

从图形上来说，一条位于终端、从代理对象延伸到其启动的过程、类似于一个黑色棒棒糖并带有实心圆的线应表示一个代理关联。

代理关联OPL句子的语法应为：代理处理过程加工。

示例1：图 8 的对象过程图中（OPD），焊接工是焊接的代理。实施将对象钢材 A 部分与对象钢材 B 部分的焊接以创建钢材 AB 部分的这个过程需要一个人类的焊接工。焊接工是焊接的代理。然而，焊接不会将焊接工进行转换，但没有焊接工则焊接将不会发生。



焊接工进行焊接；
焊接消耗钢材A部分和钢材B部分；
焊接产生钢材AB部分。

图8　代理关联例子

示例2：图 8 中的对象过程图中，无论是什么原因，焊接工在焊接完成之前离开的话，焊接将会提前停止，钢材 AB 部分的创建就不会出现，尽管焊接已经消耗了钢材 A 部分和钢材 B 部分。

8.2.3　仪器和仪器关联

一个仪器应是一个过程的无生命或不能进行决策的使能器，而这个过程则是在仪器不存在和不可用的情况下无法启动或发生。

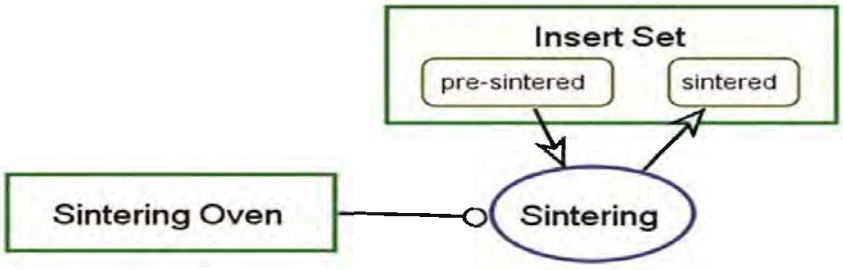
一个仪器关联应是一个从仪器对象到其所启动的过程的使能关联，明确指出了仪器对象对于关联的过程激活和运行是必要的。

从图形上来说，一条位于终端、从仪器对象延伸到其启动的过程、类似于一个白色棒棒糖并带有一个开放圆的线应表示一个仪器关联。

一个仪器关联的OPL句子的语法应为：加工处理需要仪器。

示例1：一个制造过程也许无法消耗或（不考虑损耗）改变一台可使棒料转换到机械零件的机器的状态。在这种背景下，机器是制造过程的一个工具。

示例2：在图 9 的 OPD 中，烧结炉是插入设置的仪器，因为没有它烧结就不会发生。然而，虽然插入设置的对象被转换（它的状态从预烧结变为烧结），在不考虑磨损的情况下，烧结炉由于实施烧结过程的结果而保持了未受到影响的状态。



插入设置可以是预烧结或烧结。

烧结需要有烧结炉。

烧结将插入设置从预烧结变为烧结。

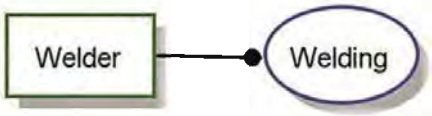
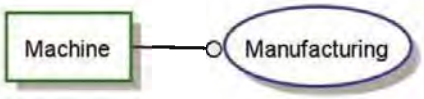
图9　仪器关联例子

示例3：在 图 9 的 OPD 中，如果在烧结过程中烧结炉停止运行的话，如由于严重开裂，烧结将停止并且插入设置不会处于其烧结状态中，虽然它已经离开了其预烧结状态。

8.2.4　基本使能关联小结

表2 总结了基本使能关联。

表2　基本使能关联小结

名称	语义	OPD & OPL 例子	源	目的地
代理关联	代理人是一个或一组人，其使得所连接但未被该过程转换的过程出现。	 焊机处理焊接。	代理人-启用对象	已被启动过程
仪器关联	仪器是一个无生命的对象，其使得所连接但未被该过程转换的过程出现。	 制造需要机械。	仪器 - 启用对象	已被启动过程

8.3　状态-指定转换关联

8.3.1　状态-指定消耗关联

一个状态-指定消耗关联应是一个从被消耗物的特定状态到消耗（破坏、消除）的对象的关联过程的消耗关联。处于一个指定状态中的被消耗物的存在应是一个用于过程激活的前置条件或部分前置条件。如果被消耗物不处于该指定状态中的话，那么过程激活应等待被消耗物在那个特定状态下存在。

从图形上来说，一个从对象的指定状态指向消耗对象的过程的带有一个封闭箭头的箭状符号应表示状态-指定消耗关联。

一个状态-指定消耗关联的OPL句子的语法应为：过程消耗指定-状态对象。

消耗应在过程激活时立即启动，除非建模者需要一定的时间对对象的消耗进行建模。在这种情况下，消耗关联应具有一个显示被消耗物消耗率的属性，并且被消耗物应具有一个显示可用数量的属性。

注1：如果对象数量小于所预期的过程持续时间的速率时间的话，建模者可以创建一个异常。

注2：见 11.1 连接属性的定位。

示例1：钢条在状态预热处理对于过程机械加工是一个被消费物，其产生了一个结果物轴。当机械加工激活时，它就会消耗预热处理钢棒，因为该预热处理钢条只能用于变成高过程的一个结果物轴。如果钢条在之前经历过热处理过程的话，它即处于热处理状态而因此不用再进行机械加工。

示例2：继例 1 之后，钢条处于状态预热处理状态，并具有值为 600 的属性数量[单位]。状态-指定的消费关联具有值为 60 的属性率[单位/小时]。当机械加工操作时，它就会在 10 个工作小时后消耗 600 个钢条。

8.3.2 状态-指定结果关联

一个状态-指定结果关联应是一个从过程到过程所创建（生成、生产）的结果物的一个指定状态的结果关联。处于指定状态的结果物的存在应是一个在生成过程一旦完成时的后置条件或部分后置条件。

从图形上来说，一个从过程指向对象指定状态的具有一个封闭箭头的箭状符号应表示状态-指定结果关联。

一个状态-指定结果关联OPL句子的语法应为：过程产生指定-状态对象。

在特定状态下结果物的生成应在过程完成后立即开始，除非建模者需要时间来对对象的生成进行建模。在这种情况下，结果关联应具有一个显示其结果物生成速率的属性，而结果物应具有显示在该指定状态下可用数量的属性。

注1：见 10.1 连接属性的定位。

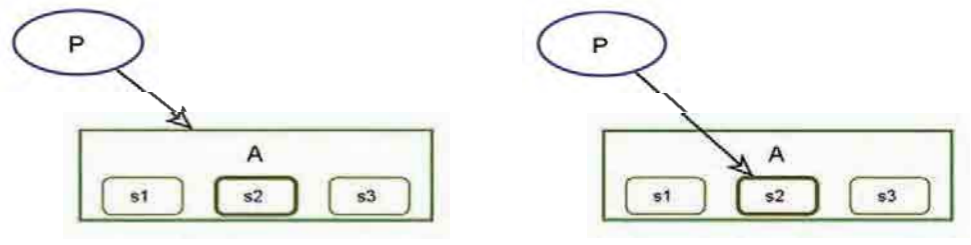
注2：运行期间，一个运行模型可以包含一个对象的多个运行实例，每一个不同状态拥有一个运行实例。

示例1：处在预热处理状态的钢条对于机械过程来说是一个被消耗物，其产生了处于预热处理状态的结果物轴。一个从机械加工到轴的预热处理的状态-指定结果关联表示该模型规范。

一个生成具有初始状态的有状态对象的结果关联应在该对象矩形处附带其初始状态之外的其中状态之一。

注3：建模者也许想将 OPL 放在图 10 中的右侧，但是左侧的 OPL 减少了模糊性。

示例2：



A 可以是 s1，s2 或 s3。
S2 是初始。
P 产生 A。

A 可以是 s1，s2 或 s3。
S2 是初始。
P 产生 s2 A。

图10 结果关联到正确 （左边）和不正确（右边）的具有一个初始状态的对象

8.3.3 状态-指定效果关联

8.3.3.1 输入和输出效果关联

一个输入源关联应是一个来自对象指定状态，即一个输入源，到转换过程的关联，而输出目标关联应是来自转换过程到一个对象的指定状态，即一个输出目标的关联。这些关联提供了三种处于一个单个对象连接到一个单个过程情景下的可能性建模情况：

a)输入-输出-指定效果关联，指定了输入源和输出目的地状态两者；

b)输入-指定效果关联仅指定了输入源状态；

c)输出-指定效果关联仅指定了输出目标的状态。

8.3.3.2 输入-输出-指定效果关联

一个输入 - 输出指定效果关联应是一对效果关联，其中输入源关联连接到一个来自受影响物的指定状态的影响过程，且输出目标关联从相同过程关联连接到一个相同受影响物的不同输出目标状态。在输入源状态存在的受影响物对于影响过程启动应是一个前置条件或前置条件的一部分。在输出目标状态存在的受影响物应是影响过程结束时的一个后置条件或后置条件的一部分。

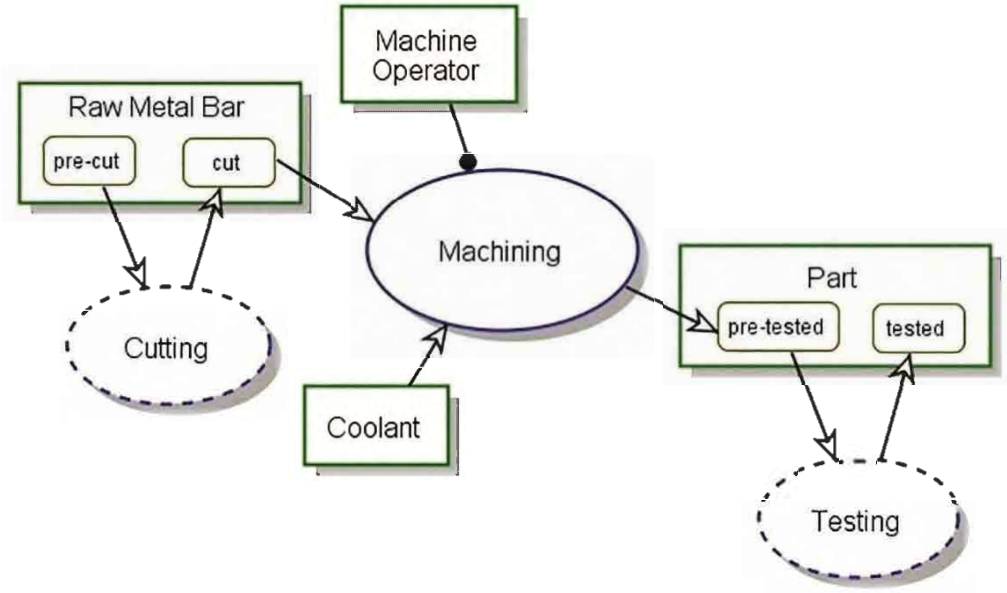
从图形上来说，一对包含有一个从受影响物输入源状态指向输入源关联的影响过程封闭箭头的箭状符号以及一个从该过程指向过程完成时的受影响物输出目标状态，即输出目标连接的相同箭应意味着输入-输出-指定效果关联。

一个输入-输出-指定效果关联的OPL句子的语法应为：过程将对象从输入-状态改变为输出-状态。

示例1：图 11 中的 OPD 描绘了状态-指定消耗关联和结果关联。机械加工只能消耗处于切割状态的金属棒原料并生成预测试状态中的部分。切割和测试是环境过程。切割需要先于机械加工以便将金属棒原料从其预切割状态变为切割状态，而测试从预测试部分变为测试部分。

注1：在一个输入-输出-指定效果关联的情况下，一旦一个影响过程开始，它就会导致对象退出其输入源状态。然而，对象只在过程完成时才可达到其输出目标状态。在过程开始与完成之间，受影响物对象处于两种状态之间的过渡期。

示例2：图 11 的 OPD 中，切断将金属棒原料从其预-切割变为其切割状态。只要切割处于活跃状态，金属棒原料的状态就处于过渡期，势必会过渡到切割状态：切割将其预-切割状态中带出来但尚未将其带入过程完成的切割状态。虽然切割的金属棒原料的状态是不确定的，它仍可被部分切割及可重复使用或被大部分切割而无法使用。在这两种情况的任何一种情况下，它是不能用于机械加工的，因为它不处于其切割状态。



金属棒原料是物理性的；
金属棒原料 可以是预-切割或切割；
机械操作员是物理性的；
冷却液是物理性的；
机械加工是物理性的；
机械加工需要冷却液；
机械操作员处理机械加工；
部分是物理性的；
部分可以是预-测试或被测试的；
测试是环境性和物理性的；
切割是环境性的和物理性的；
切割将金属棒原料从预-切割变为切割；
机械加工消耗切割金属棒原料；
机械加工生成预-测试部分；
测试将部分从预-测试变为测试。

图11　状态-指定消耗关联和结果关联

注2：如果一个活跃的影响过程过早停止，即它没有完成的话，任何受影响物的状态都保持着不确定性，除非异常处理将对象解析至其可允许状之一的状态。

8.3.3.3　输入-指定效果关联

一个输入-指定效果关联应是一对效果关联，其中输入源连接了一个来自受影响物输入源状态的影响过程，而输出目标关联将来自同样的过程连接到同样的受影响物而并没有指定一个特定状态。对象输出目标状态应是其默认状态，或者，如果对象没有默认状态，则对象的状态概率分布应确定该对象的输出目标状态（见11.7）。

在输入源状态存在的受影响物是一个影响过程激活的前置条件或前置条件的一部分。处于其任何一个状态的受影响物的存在应是影响过程完成时的一个后置条件或后置条件的一部分。

从图形上来说，一对包含有一个封闭箭头的箭状符号从受影响物输入源状态，即输入关联和一个来自该过程到受影响物但未到其任何一个状态的相似箭状符号应意指输入-指定效果关联。

一个输入-指定效果关联的OPL句子的语法应为：过程改变了对象的输入状态。

8.3.3.4 输出-指定效果关联

一个输出-指定的效果关联应是一对效果关联，其中输入源关联与一个来自受影响物的影响过程进行连接而没有指定一个特定状态，并且，输出目标关联从相同过程连接到一个相同受影响物的输出目标状态中去。受影响物的存在应是一个用于影响过程激活的前置条件或前置条件的一部分。在输出目标状态存在的受影响物应是影响过程结束时的一个后置条件或后置条件的一部分。

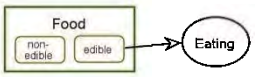
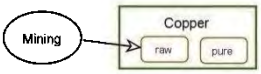
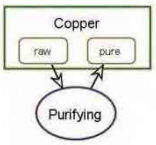
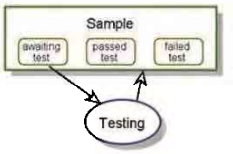
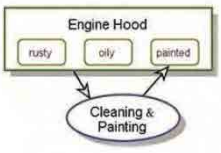
从图形上来说，由来自一个受影响物具有封闭箭头的箭状符号所组成的一对箭没有指定一个特定状态，即输入关联，一个类似的从该过程到该受影响物，即输出关联的输出目的地状态的箭状符号应表示输出-指定效果关联。

一个输入-指定效果关联的OPL句的语法应为：过程将对象变为输出-状态。

8.3.4 状态-指定转换关联小结

表3 总结了状态-指定转换关联。

表3　状态-指定转换关联小结

名称	语义	OPD & OPL 例子	来源	目标
状态-指定消耗关联	过程只在对象处于指定状态中才消耗对象。	 吃消费了可吃食品。	被消耗物状态	过程
状态-指定结果关联	过程生成处于指定状态中的对象。	 采矿生成了粗铜。	过程	结果物状态
输入-输出指定效果关联对 （包含一个状态-指定输入关联和一个状态-指定输出关联）	过程将对象从一个指定输入状态通过输入关联改变为一个通过输出关联的指定输出状态。	 提纯将铜从原料变成纯铜。	受影响物源状态	影响过程
			影响过程	受影响物目标状态
输入-指定效果关联对 （包含一个状态-指定输入关联和一个状态-未指定输出关联）	过程将来自一个指定状态的对象改变为任何输出状态。	 测试从等待测试变为样本。	受影响物源状态	影响过程
			影响过程	受影响物
输出-指定效果关联对 （包含一个状态-未指定输入关联和一个状态-指定输出关联）	过程将对象从任何输入状态改变为一个指定输出状态。	 清洗和喷涂将发动机罩改变成上漆。	受影响物	影响过程
			影响过程	受影响物目的地状态

8.4　状态-指定使能关联

8.4.1　状态-指定代理关联

一个状态-指定代理关联应是一个来自代理指定状态到一个过程的代理关联。处于指定状态中的代理对过程激活和运行是必要的。

以图形来说，一条处于终端、从代理出发的特定状态延伸至其所启动的过程的一个类似黑色棒棒糖的实心圆的线应表示一个状态-指定代理关联。

一个状态-指定代理关联的OPL句子的语法应为：状态-指定代理负责加工。

注：状态名称标签的开头不以大写字母出现，除非它们出现在一个OPL句子的开始。

示例：一个飞行员需要保持清醒状态才有资格作为一个飞机飞行过程的代理。在 OPL 中：清醒的飞行员进行飞行。

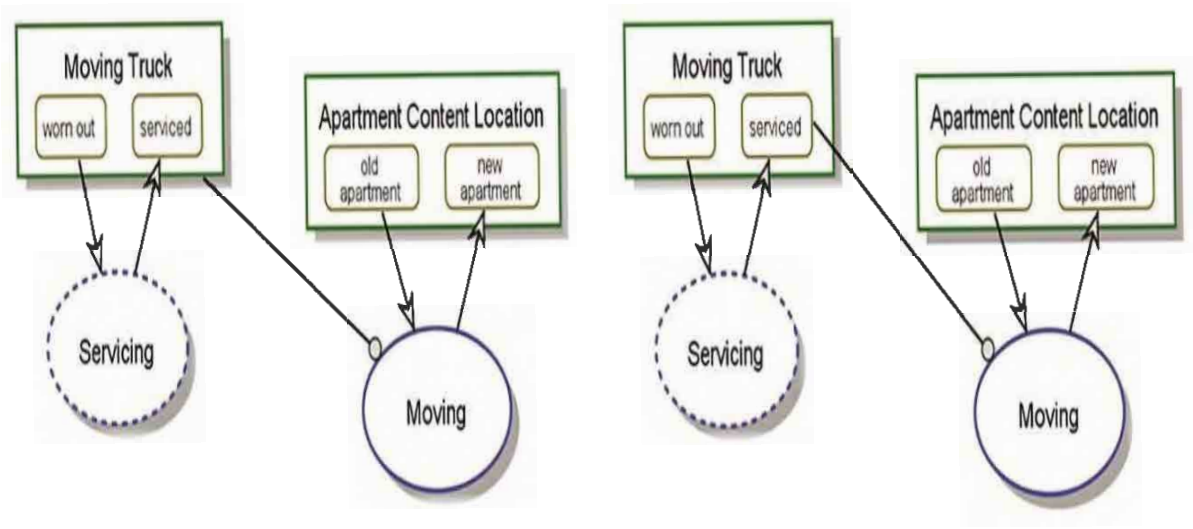
8.4.2 状态-指定仪器关联

一个状态-指定仪器关联应是一个从仪器指定状态到过程的仪器关联。处于指定状态的仪器对于过程激活和性能是必要的。

以图形来说，一条处于终端、从仪器对象的特定状态延伸至其所启动的过程、类似白色棒棒糖的具有实心圆的线应表示一个状态-指定仪器关联。

一个状态-指定仪器关联的OPL句子的语法应为：处理过程需要指定-状态的仪器。

示例：图 12 中的 OPD 描绘了基本与状态仪器关联之间的差别。左侧的移动卡车这个对象是移动的工具，意味着该对象的状态并不重要，而右侧的移动卡车的服务这种合格状态是一个移动的工具，意味着只有在移动卡车被服务时移动才有可能发生。



移动卡车是物理性的。
移动卡车可以是磨损或被检修。
服务是环境性和物理性的。
服务将移动卡车 从磨损变成被维修。
公寓位置是物理性的。
公寓位置可以是旧公寓或新公寓。
移动是物理性的。
移动需要移动卡车。
移动将公寓位置从旧公寓变成新公寓。

.

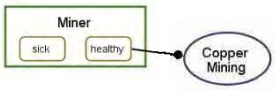
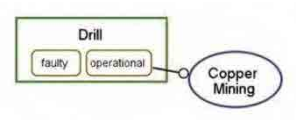
移动卡车是物理性的。
移动卡车可以是磨损或被检修。
服务是环境性和物理性的。
服务将移动卡车 从磨损变成被维修。
公寓位置是物理性的。
公寓位置可以是旧公寓或新公寓。
移动是物理性的。
移动需要被维修过的移动卡车。
移动将公寓位置从旧公寓变成新公寓。

图12 左侧的仪器关联与右侧的状态-指定仪器关联

8.4.3 状态-指定使能关联小结

表4 总结了状态-指定使能关联。

表4　状态-指定使能关联小结

名称	语义	OPD & OPL 样本	来源	目标
状态-指定代理关联	人类代理使使得过程处于指定状态。	 健康的矿工进行铜矿开采。	代理状态	使能过程
状态-指定仪器关联	过程要求仪器处于指定状态。	 铜矿开采需要操作打眼。	仪器状态	使能过程

8.5　控制关联

8.5.1　控制关联类型

作为OPM运行语义基础的事件-条件- 行为范式的一部分（参见8.2.1），一个事件关联、一个条件关联和一个异常关联应分别表达一个事件、一个条件和一个时间异常。这三种关联类型应是控制关联。控制关联应在一个对象与一个过程之间或在两个过程之间发生。

一个事件关联应指定一个源事件和目标过程以在事件发生时进行激活。事件发生会导致对过程前置条件的满意度进行一个评估。

满足前置条件能使过程继续运行，并且过程变得活跃起来。如果过程前置条件未能得到满足，则过程操作将不会发生。无论评估成功与否，事件都会被丢失。

如果过程的前置条件得不到满足的话，过程激活将不会发生，直至另一个事件将该过程激活。控制关联决定过程是否需等待另一个激活事件，或者执行控制流是否该绕行过过程。

注1：后续事件可来自其他源以启动前置条件的评估。

一个条件关联应是一个源对象或对象状态与目标过程之间的程序关联。一个条件关联应提供一个绕行机制，其能使系统执行控制在其前置条件满意度评估失败时跳过或绕行目标过程。

注2：在没有条件关联绕行机制的情况下，未能满足前置条件的这种失败将会约束该过程要等待满足前置条件。

对于事件关联和条件关联这两者来说，每一种进入转化关联和使能关联，即从一个对象或对象状态到一个过程的关联都应具有一个相应类型的事件关联和条件关联。

异常关联应是一个因某种原因无法成功完成的过程或比预期时间更长或更短的过程与一个管理异常情况的过程之间的一种程序关联。

注3：由于未能成功完成所产生的失败常常会导致时间不足或超时运行，异常连接可服务于其它情形。此外，所有与时间无关的异常状态可使用取值范围（每一种用法请参见第第C.6条 ）来建模。

从图形来说，一个紧挨着转换关联或使能关联，即从一个对象或对象状态到一个过程、作为一个注释出现的控制修饰符应表示相应的控制关联。表示事件的注释符号“e” 代表一个事件关联，表示条件的注释符号“c” 应表示一个条件关联。用于一个异常关联的控制修饰符注释则是一个或两个横跨异常管理过程附近关联的短棒。

8.5.2 事件关联

8.5.2.1 转换事件关联

8.5.2.1.1 消耗事件关联

一个消耗事件关联应是一个对象与一个对象运行实例所启动的过程之间的一个带标注消耗关联。满足过程前置条件和后续过程性能会消耗启动对象的实例。

从图形来说，一个带有封闭箭头的箭状符号从对象指向箭头旁带有表示事件的小写字母 “e” 注释的过程应表示消耗事件关联。

消耗事件关联的OPL句子的语法应为：对象启动过程，过程消耗对象。

8.5.2.1.2 效果事件关联

一个效果事件关联应是一个从对象到一个对象操作实例所启动的效果关联的标注部分。满足过程的先决条件和后续过程性能将会以某种方式影响到启动对象。

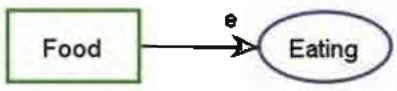
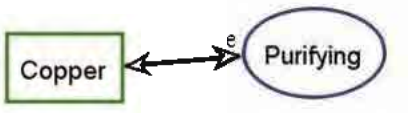
从图形上来看，一个位于对象与表示事件的箭状符号的过程尾部旁、带有小写字母“e”注释之间的每个端部具有封闭箭头的双向箭状符号应表示效果事件关联。

效果事件关联的OPL句子的语法应为：对象启动过程，过程影响对象。

8.5.2.1.3 转换事件关联总结

表5 总结了转换事件关联。

表5 转换事件关联总结

名称	语义	OPD & OPL 样本	来源	目的地
消耗事件关联	对象启动过程，如果进行操作，它就会消耗对象。	 食物诱发吃，吃消耗食物。	启动被消费者	启动过程，其消耗启动被消费物
效果事件关联	对象启动过程，如果进行操作，它就会影响对象。	 铜材引致提炼，提纯影响铜材。	启动受影响者	启动过程，其影响到受影响者
注：事件关联是从对象到过程的关联；从过程到对象的关联不是一个事件关联。				

8.5.2.2 使能事件关联

8.5.2.2.1 代理事件关联

一个代理事件关联应是一个从代理对象到其所启动和启用的过程的一个带标注使能关联。

以图形来说，一条处于终端、类似一个黑色棒棒糖的实心圆的线从一个代理对象延伸至其所启动和启用的带有一个代表事件的小写字母“e”注释的过程应表示一个状态-指定事件关联。

代理事件关联的OPL句子的语法应为：代理启动并处理过程。

8.5.2.2.2 仪器事件关联

一个仪器事件关联应是一个从仪器对象到其启动和启用过程的带标注使能关联。

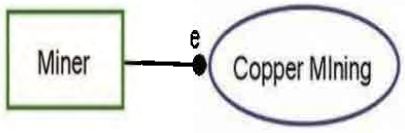
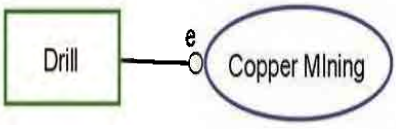
从图形来说，一条处于终端、类似一个白色棒棒糖的空心圆的线从一个仪器对象延伸至其所启动和启用的带有一个代表事件的小写字母“e”注释的过程应表示一个仪器事件关联。

仪器事件关联的OPL句子的语法应为：仪器启动过程，过程需要仪器。

8.5.2.2.3 使能事件关联总结

表6 总结了使能事件关联。

表6 使能事件关联总结

名称	语义	OPD & OPL 例子	来源	目标
代理事件关联	代理 -一个人：两者启动和启用过程。代理需要通过过程持续时间而存在。	 矿工启动并进行铜矿开采。	启动代理	已启动过程
仪器事件关联	对象启动作为一个仪器的过程，因此它不会改变，但需要通过过程持续时间而存在。	 钻探启动铜矿开采，而铜矿开采则需要钻探。	启动仪器	已启动过程

8.5.2.3 状态-指定转换事件关联

8.5.2.3.1 状态-指定事件关联

一个状态-指定消耗事件关联应是一个来自对象的指定状态到一个对象操作实例所启动的过程的带标注消耗关联。对指定状态中包含启动对象过程的先决条件及后续过程性能的满足度将会消耗启动对象。

从图形来说，一个具有封闭箭头的箭状符号从对象的指定状态指向箭头旁带有一个表示事件的小写字母 “e” 的注释过程应意味状态-指定消耗事件关联。

一个状态-指定消费事件关联的OPL句子的语法应为：指定-状态对象启动过程，而过程则消耗对象。

8.5.2.3.2 输入-输出-指定效果事件关联

一个输入-输出-指定的效果事件关联应是一个在对象运行实例进入指定输入源状态时启动影响过程的带标注输入-输出效果关联。

从图形来说，在输入关联的箭状符号头终端旁带有一个表示事件的小写字母 “e” 注释的输入-输出-指定效果关联应表示输入-输出-指定效果事件关联。

一个输入-输出-指定效果事件关联的OPL句子的语法应为：输入-状态对象启动过程，而过程将对象由输入-状态改变为输出-状态。

8.5.2.3.3 状态-指定效果事件关联

一个输入-指定效果事件关联应是一个在对象运行实例进入指定输入源状态时启动影响过程的带标注输入-指定效果关联。

从图形来说，在输入关联箭头端，一个带有代表事件的小写字母 “e” 注释的输入-指定效果关联应表示输入-指定效果事件关联。

一个输入-指定效果事件关联的OPL句子的语法应为：输入-状态对象启动过程，而过程则将改变了对象的输入-状态。

8.5.2.3.4 输出-指定效果事件关联

一个输出-指定效果事件关联应是一个在对象运行实例开始存在时启动影响过程的带标注输出-指定效果关联。

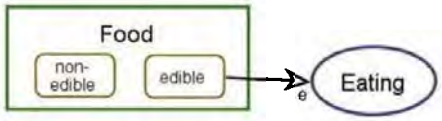
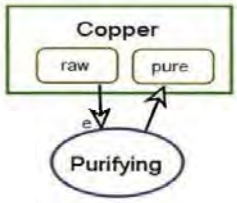
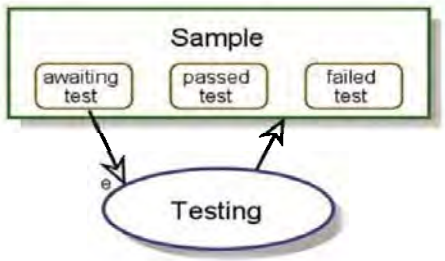
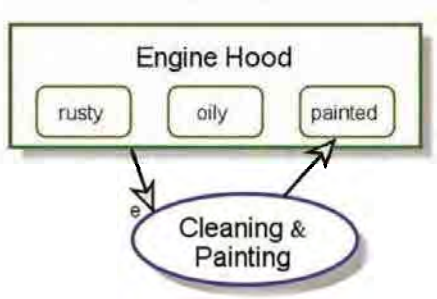
从图形来说，在输入关联箭头端一个带有表示事件的小写字母 “e” 注释的输出-指定效果关联应表示输出-指定效果事件关联。

一个输出-指定效果事件关联的OPL句子的语法应是：处于任何状态下的对象启动过程，而过程则将对象变成目标-状态。

8.5.2.3.5 状态-指定转换事件关联小结

表7 总结了状态-指定转换事件关联。

表7　状态-指定转换事件关联小结

名称	语义	OPD & OPL 例子	来源	目标
状态-指定消耗事件关联	处于指定状态中的对象即启动过程也同时被其消耗。	 可食用启动吃，而吃则消耗食物。	被消耗物状态	已启动的过程
输入-输出指定事件关联对	处于指定状态中的对象即启动过程也同时被其转换至输出状态。	 铜材原料启动了提纯，而提纯则将铜材从原料变成纯铜。	受影响物源状态	启动过程
			启动过程	受影响物目标状态
输入-指定效果关联对	处于指定状态中的对象即启动过程也同时被转换成其状态之一的状态。	 等待测试样本启动了测试，而测试则从等待测试改变了样本。	受影响物源状态	已启动的过程
			启动过程	受影响物
输出-指定事件关联对	（处于任何状态的其中一种状态的）对象即启动过程又同时被其转换至输出状态。	 发动机罩启动清洁和喷漆，而清洁和喷漆又将发动机罩变成被上漆状态。	受影响物	启动过程
			启动过程	受影响物目标状态

8.5.2.4 状态-指定使能事件关联

8.5.2.4.1 状态-指定代理事件关联

一个状态-指定代理事件关联应是在代理进入指定状态时启动过程的带标注状态-指定代理关联。

从图形来说，一个在关联过程端旁带有一个代表事件的小写字母 “e” 注释的状态-指定关联应表示状态-指定代理事件关联。

一个状态-指定代理事件关联的OPL句子的语法应为：指定-状态代理启动并处理过程。

8.5.2.4.2 状态-指定仪器事件关联

一个状态-指定仪器事件关联应是一个在仪器运行实例进入指定状态时启动过程的带标注状态-指定仪器关联。

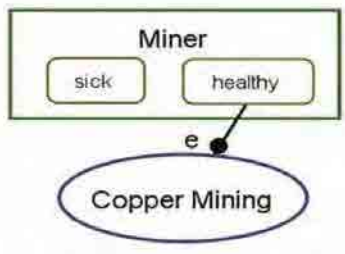
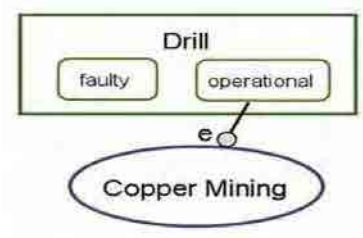
从图形来说，在关联过程端旁一个带有代表事件的小写字母 “e” 注释的状态-指定仪器关联应表示状态-指定仪器事件关联。

一个状态-指定仪器事件关联的OPL句子的语法应为：指定-状态仪器启动过程，而过程则需要指定-状态仪器。

8.5.2.4.3 状态-指定使能事件关联小结

表8 总结了状态-指定使能事件关联。

表8 状态-指定使能事件关联小结

名称	语义	OPD & OPL 范例	来源	目标
状态-指定代理事件关联	处于指定状态的人类代理即启动过程也充当其代理。 代理需要在整个过程持续时间处于指定的状态。	 健康的矿工启动和从事铜矿开采。	代理状态	已启动过程
状态-指定仪器事件关联	处于指定状态的对象即启动过程也是其运行的仪器。 仪器需要在整个过程持续时间处于指定的状态。	 运行钻探启动铜矿开采，而铜矿开采需要运行钻探。	仪器状态	已启动的过程

8.5.2.5 调用关联

8.5.2.5.1 过程调用和调用关联

过程调用应是一个过程通过此调用来启动一个过程的事件。一个调用关联应是一个从源过程到其调用（初始）目标过程的关联，表示当源过程完成时，它会立即启动处于调用关联另一端的目标过程。

注1：如果先前过程没有成功完成的话，则一个正常或预期的执行控制流将不会调用一个新过程。通常由建模者负责其任何被中止的过程。第 C.6 条提供了几种由于一个尤其像 C.6.8 的那种失败而需管理一个终结过程的方法。

注2：鉴于一个 OPM 过程执行一种转换，所以调用关联在语义上所隐喻的是通过调用随后被调用目标过程立刻消耗了的源过程来创建一个临时对象。在一个 OPM 模型中，一个调用关联可取代一个元过程所创建的过渡性暂时物理或信息对象（如查询中的记录 ID）以启动那个会立即消耗一个临时对象的目标过程。

从图形来说，一个来自调用源过程的闪电符号折线到处于调用过程的一个以封闭箭头结尾的被调用目标过程应意味着一个调用关联。

一个调用关联的OPL句子的语法应为：调用-过程调用已调用的过程。

8.5.2.5.2 自调用关联

自调用应是一个过程本身的调用，当过程完成时，该过程就会立即进行自调用。自行-调用关联应指定自调用。

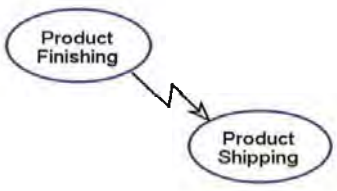
从图形来说，一对源于过程并在回到源过程之前头尾相连的调用关联应表示自调用关联。

一个自调用关联的OPL句子的语法应为：调用过程进行自调用。

8.5.2.5.3 调用关联小结

表9 总结了调用关联。

表9 调用关联小结

名称	语义	OPD & OPL 例子	来源	目标
调用关联	一旦调用过程结束，它就会调用由点用连接所指向的过程。	 产品完成调用了产品运输。	启动过程	另一启动的过程
自调用关联	一旦过程完成，它将立刻自我启动。	 重复性过程加工进行自调用。	启动过程	相同过程

8.5.3 条件关联

8.5.3.1 基本条件转换关联

8.5.3.1.1 条件消耗关联

一个条件消耗关联应是一个从被消耗物到一个过程的带标注消耗关联。如果一个被消耗物运行实例在一个事件启动过程时存在的话，那么该被消耗物运行实例的出现就会满足有关该对象的过程先决条件。如果整个前过程对象集的评估满足了前置条件，那么过程就会开始并消耗该被消耗物实例。然而，如果在一个事件启动过程时一个被消耗物运行实例不存在，则过程前置条件评估失败，而且执行控制流将绕行或 “跳过” 没有过程性能的过程。

从图形来说，一个带有封闭箭头的箭状符号从被消耗物指向箭头旁带有一个代表条件的小写字母“c”注释的过程意味着一个条件消耗关联。

一个条件消耗关联的OPL句子的语法应为：如果对象存在，则过程发生，在此情况下，对象被消耗，否则过程将被跳过去。

条件消耗关联OPL句子的一个替代语法应为：如果对象存在，则过程发生并消费对象，否则将绕行过程。

注：见13.2.2.4.2关于相关“跳过”语义的补充细节和图C.25中的几个例子。

8.5.3.1.2 条件效果关联

一个条件效果关联应是一个从受影响物到过程的带标注效果关联。如果一个受影响物对象运行实例在一个事件启动过程时存在的话，则该受影响物实例的存在将会满足相对于该对象的过程前置条件。如果整个前过程对象集的评估满足了前置条件，则过程开始并影响那个受影响物实例。然而，如果一个受影响物运行实例在一个事件启动过程时不存在的话，则过程前置条件评估失败，并且执行控制流将绕行或 “跳过”没有过程性能的过程。

从图形来看，一个具有两个封闭箭头的双向箭状符号。每一个箭状符号均在受影响物与影响过程之间指向一个方向、位于一个箭头过程端旁带有一个表示条件的小写字母“c” 的注释应表示一个条件效果关联。

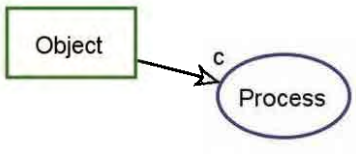
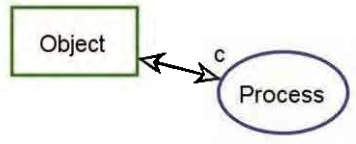
条件效果连接的OPL句子的语法应为：如果对象存在，则过程发生，在此情况下，过程会影响对象，否则过程将被跳过去。

一个条件效果关联的OPL句子的替代语法应为：如果对象存在，则过程发生并且影响对象，否则过程将被绕行。

8.5.3.1.3 条件转换关联小结

表10 总结了条件转换关联。

表10　条件转换关联小结

名称	语义	OPD & OPL 例子	来源	目标
条件消费关联	如果一个对象运行实例存在并且其余过程先决条件得到满足的话，那么这个过程将执行和消耗对象实例，否则执行控制将继续进行以启动下一个过程。	 如果对象存在，则过程就会发生，在这种情况下，过程消耗对象，否则过程将被跳过去。	条件对象	制约过程
条件效果关联	如果一个对象操作实例存在，且其余的进程先决条件得到了满足，那么这个过程将执行并影响对象实例，否则执行控制将继续进行以启动下一个进程。	 如果对象存在，则过程就会发生，在这种情况下，过程会影响对象，否则过程将会被跳过去。	条件对象	制约过程

8.5.3.2　基本条件使能关联

8.5.3.2.1　条件代理关联

一个条件代理关联应是一个从代理到过程的带标注代理关联。如果一个代理运行实例在一个事件启动过程时存在的话，则该代理实例的存在将会满足关于该对象的过程前置条件。如果整个前过程对象集的评估满足了前置条件的话，该过程就会开始，并且该代理会处理其性能。然而，如果一个代理运行实例在一个事件启动过程时不存在的话，则过程前置条件评估失败，执行控制流就会绕行或 “跳过” 没有过程性能的过程。

从图形来看，一条位于末端类似一个黑色棒棒糖的实心线从一个代理对象延伸到其所启用的过程的一个代表条件的带有小写字母“c” 的注释意味着一个条件代理关联。

条件代理关联的OPL句子的语法应为：如果代理存在，则代理处理过程，否则过程将被跳过去。

用于条件代理关联的OPL句子的一个替代语法应为：如果代理存在，则代理处理过程，否则过程将被绕行。

8.5.3.2.2　条件仪器关联

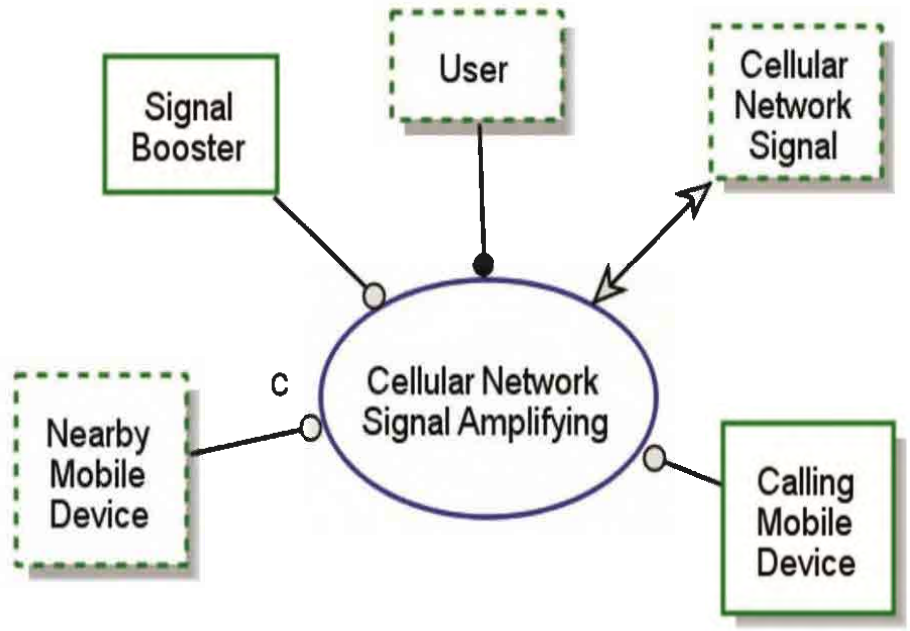
一个条件仪器关联应是从仪器到过程的带标注仪器关联。如果一个仪器运行实例在一个事件启动过程时存在的话，则该仪器运行实例的出现就会满足关于该对象的过程先决条件。如果整个前过程对象集

的评估满足了前置条件，那么该过程就会开始。然而，如果一个仪器运行对象在一个事件启动过程时不存在的话，那么过程前置条件的评估则失败，且执行控制流将会被绕行或“跳过”没有过程性能的过程。

从图形来说，一条处于终点末端类似白色棒棒糖的空心圆线从一个仪器对象延伸到其所启用的过程的一个处于过程终端、带有一个代表条件的小写字母 “c” 的带标注应表示一个条件仪器关联。条件仪器关联OPL句子的语法应为：如果仪器存在，则过程发生，否则过程将被跳过去。

条件仪器关联OPL句子的一个替代语法应为：如果仪器存在，则过程发生，否则过程将被绕行。

示例：图 13 是一个从附近移动设备到蜂窝网络信号放大的带有条件仪器关联的 OPD，其只在一个环境对象附近移动设备存在时才会发生，否则将被跳过，因为如果附近没有设备的话则不需要进行放大。



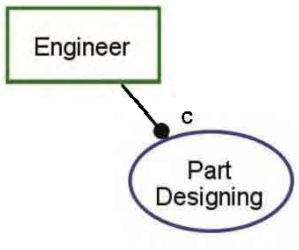
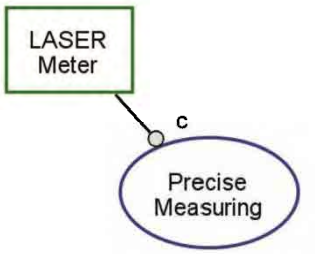
如果附件移动设备存在，则蜂窝网络信号放大发生，否则蜂窝网络信号放大被跳过去。

图13　条件仪器关联（带有部分 OPL ）

8.5.3.2.3　基本条件使能关联小结

表11总结了基本条件使能关联。

表11　基本条件使能关联小结

名称	语义	OPD & OPL 例子	来源	目标
代理条件关联	如果代理出现的话，代理者启用过程，否则过程就会被跳过去。	 如果工程师在的话，他会处理部分设计工程，否则部分设计工程会被跳过。	条件代理	制约过程
仪器条件连接	如果它存在的话，仪器就会启用过程，否则过程就会被跳过去。	 如果激光仪存在，就会发生精确测量，否则精确测量会被跳过去。	条件仪器	制约过程

8.5.3.3　条件状态-指定转换关联

8.5.3.3.1　条件状态-指定消费关联

一个条件状态-指定消耗关联应是一个从一个被消耗物的指定状态到过程的带标注条件消耗关联。如果一个处在指定状态的该被消耗物运行实例在一个事件启动过程时存在，那么该被消耗物实例的出现将会满足有关该对象的过程先决条件。如果对整个前过程对象集的评估满足了前置条件，则过程就会开始并消耗那个被消耗物实例。然而，如果一个处在指定状态的该被消耗物运行实例在一个事件启动过程时不存在的话，则过程前置条件的评估失败，执行控制流将绕行或“跳过”没有过程性能的过程。

从图形来说，一个带有封闭箭头的箭状符号从受消费物的指定状态指向在箭头附件带有一个表示条件的小写字母“c” 注释的过程意味着一个条件状态-指定消耗关联。

条件状态-指定消耗关联的OPL句子的语法应是：如果对象是指定-状态，则过程会发生，在此情况下，对象被消耗，否则过程将被跳过去。

条件状态-指定消耗关联的OPL句子的一个替代语法应为：如果指定-状态对象存在，则过程发生并消费对象，否则过程将被绕行。

8.5.3.3.2　条件输入-输出-指定效果关联

一个条件输入 - 输出-指定效果关联应是一个从源输入状态到过程的带标注输入-输出-指定效果关联。如果一个处于特定状态的受影响物运行实例在一个事件启动过程时存在，则该受影响物实例的存

在满足了关于该对象的过程先决条件。如果整个前过程对象集的评估满足了前置条件，那么过程将会开始并通过将实例状态从特定输入状态变为特定输出状态来影响该对象运行实例。然而，如果一个处于特定状态的受影响物运行实例在一个事件启动过程时不存在，则过程前置条件的评估失败，并且执行控制流将会绕行或 “跳过” 没有过程性能的过程。

以图形来说，在输入连接箭头旁、带有一个代表条件的小写字母 “c” 的条件输入- 输出 - 指定效果关联意味着一个条件输入-输入-指定效果关联。

条件输入- 输出 - 指定效果关联的OPL句子的语法应为：如果对象是输入-状态，则过程发生，在此情况下，过程将对象从输入-状态改变为输出-状态，否则过程将会被跳过去。

条件输入 - 输出-指定效果关联的OPL句子的一个替代语法应为：如果输入状态对象存在，则过程将对象从输入-状态改变为输出-状态，否则过程将被跳过去。

8.5.3.3.3 条件输入-指定效果关联

一个条件输入-指定效果关联应是一个从一个源输入状态到一个过程的带标注输入-指定状态关联。如果处于指定状态的一个受影响物运行实例在一个事件启动过程时存在，则该受影响物的出现满足了相关于该对象的过程前置条件。如果整个前过程对象集的评估满足了前置条件，则该过程开始并通过将实例状态从指定的输入状态改变为一个目标状态来影响对象实例。目标状态应是默认状态，或者，如果该对象不具备默认状态，则对象的状态概率分布应确定该对象的输出目标状态（见12.7）。然而，如果一个处于指定状态的受影响物运行实例在一个事件启动过程时不存在，则过程前置条件的评估失败，执行控制流将绕行或 “跳过” 没有过程性能的过程 。

从图形来说，在输入连接箭头附近、一个带有代表条件的小写字母 “c” 注释的条件输入-指定效果关联意味着条件输入-指定效果关联。

一个条件输入-指定效果关联的OPL句子的语法应为：如果对象是输入-状态，则过程发生，在此情况下，过程改变了对象的输入-状态，否则过程将被被跳。

条件输入-指定效果关联的OPL句子的一个替代语法应为：如果输入-状态对象存在，则过程改变了对象的输入-状态，否则过程将被绕行。

8.5.3.3.4 条件输出-指定效果关联

一个条件输出-指定效果关联应是一个从一个源对象到过程的带标注输出-指定效果关联。如果一个受影响物运行实例在一个事件启动过程时存在，则该受影响物的出现满足关于该对象的过程前置条件。如果整个前过程对象集的评估满足前置条件，则该过程开始并通过将实例状态改变为指定输出-状态来影响该对象实例。然而，如果一个受影响物的运行实例在一个事件启动过程时不存在，则过程前置条件的评估失败，执行控制流将绕行或 “跳过” 没有过程性能的过程。

从图形来说，在输入连接箭头附近代表条件的一个带有小写字母 “c” 注释的条件输入-指定效果关联意味着条件输出-指定效果关联。

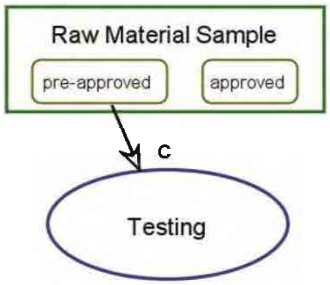
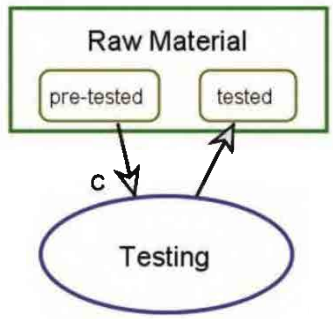
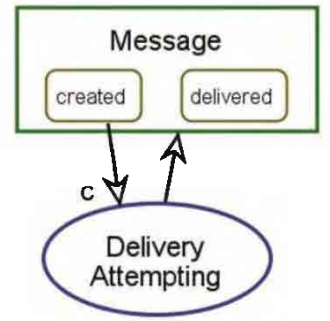
条件输出-指定效果OPL句子的语法应为：如果对象存在，则过程发生，在此情况下，过程将对象变为输出-状态，否则过程将被跳过去。

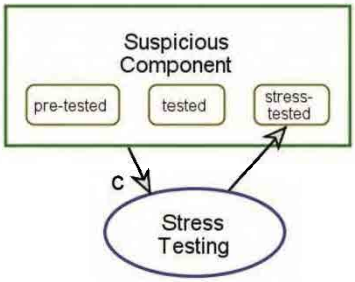
条件输出-指定效果OPL句子的一个替代语法应为：如果对象存在，则过程将对象改变为输出-状态，否则过程将被绕行。

8.5.3.3.5 条件状态-指定转换关联小结

表12 总结了条件状态-指定的转换关联。

表12　条件状态-指定的准换关联小结

名称	语义	OPD & OPL 例子	来源	目标
条件状态-指定消费关联	如果对象处于源于连接的状态中，则过程运行，否则过程将被跳过去。	 如果原材料样品获得预先批准，则测试发生，在这种情况下，原材料样品被消耗掉，否则测试将被跳过去。	制约对象指定状态	制约过程
条件输入-输出-指定效果关联	如果对象处于输入状态（连接源于此处），过程运行并将对象从其输入状态变为其输出状态，否则过程将被跳过去。	 如果原材料样品得到预先测试，则测试发生，在这种情况下，测试将原材料从预先测试改变为已被测试，否则测试将被跳过去。	制约对象指定输入状态	制约过程
条件输入-指定效果关联	如果对象处于输入状态（连接源于此处），则过程运行并将对象从其输入状态变为其状态的其中任何一种状态，否则过程将被跳过去。	 如果报文被创建，则发送尝试发生，在此情况下，发送尝试将改变报文的创建状态，否则发送尝试将被跳过去。	制约对象指定输入状态	制约过程

名称	语义	OPD & OPL 例子	来源	目标
条件输出-指定的效果关联	如果对象存在并将对象从其输入状态改变为其输出状态，则过程运行，否则过程将被跳过去。	 如果可疑成分存在，则应力测试发生，在这种情况下，应力测试将可疑成分改变为应力测试，否则，应力测试将被跳过去。	约束对象	约束过程

8.5.3.4 条件状态-指定使能关联

8.5.3.4.1 条件状态-指定的代理关联

一个条件状态-指定代理关联应是一个代理指定状态到一个过程的带标注状态-指定代理关联。如果一个处于指定状态的代理运行实例在一个事件启动过程时存在，则该代理的出现就会满足相对于该对象的过程先决条件。如果整个前过程对象集的评估满足前置条件，则该过程开始且该代理负责运行。然而，如果一个代理运行实例在一个事件启动过程时不存在，则过程前置条件评估失败，执行控制流将绕行或“跳过”没有过程性能的过程。

从图形来说，在过程终端旁带有一个表示条件的小写字母 “c” 注释的状态-指定代理关联意味着一个条件状态-指定代理关联。

条件状态-指定代理关联OPL句子的语法应为：如果代理是指定-状态，则代理处理过程，否则跳过过程。

条件状态-指定代理关联OPL句子的一个替代语法应为：如果指定-状态代理存在，则代理处理过程，否则跳过过程。

8.5.3.4.2 条件状态-指定的仪器关联

一个条件状态-指定的仪器关联应是一个从仪器指定状态到一个过程的带标注状态-指定仪器关联。如果一个处于指定状态的仪器运行实例在一个事件启动过程时存在，则该仪器的出现满足关于该仪器的过程前置条件。如果整个前过程对象集的评估满足前置条件，则过程开始。然而，如果一个仪器运行实例在一个事件启动过程时不存在，则过程前置条件评估失败，执行控制流将绕行或 “跳过” 没有过程性能的过程。

从图形来说，在过程终端旁带有一个代表条件的小写字母 “c” 注释的状态-指定仪器关联意味着一个条件状态-指定仪器关联。

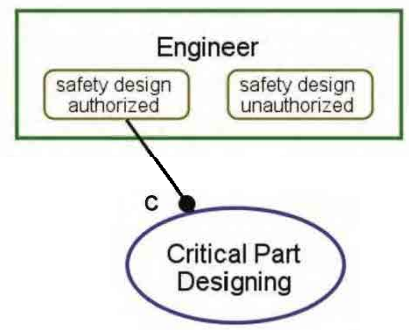
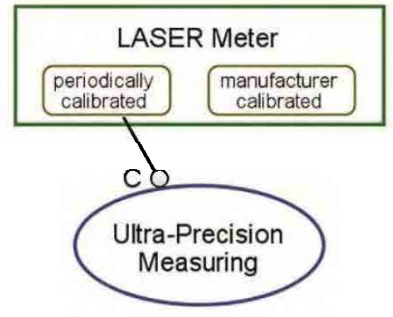
条件状态-指定仪器关联的OPL句子的语法应为：如果仪器是指定-状态，则过程发生，否则跳过过程。

条件状态-指定仪器关联OPL句子的一个替代语法应为：如果是指定-状态的仪器，则过程发生，否则跳过过程。

8.5.3.4.3 条件状态-指定使能关联小结

表13 总结了条件状态-指定使能关联。

表13 条件状态-指定使能关联小结

名称	语义	OPD & OPL 例子	来源	目标
状态-指定代理条件关联	如果代理处于一个指定的状态，则代理启用过程，否则，跳过过程。	 如果工程师被授权进行安全设计，则工程师进行关键性部件的设计，否则关键部件设计将被跳过去。	制约代理指定状态	制约过程
状态-指定仪器条件关联	如果处于指定的状态，则仪器启用过程，否则跳过过程。	 如果激光仪是定期进行校准的，则超精密测量发生，否则精确测量将被跳过去。	制约仪器指定状态	制约过程

8.5.4 异常关联

8.5.4.1 最短、预期和最长过程持续时间和持续时间分配

一个过程可拥有一个表达时间单位值的持续时间属性。持续时间可特分为最短持续时间、预计持续时间和最长持续时间。

最短持续时间和最长持续时间应指定过程完成的最短和最长允许时间单位。一个过程的预期持续时间应是该过程持续时间的统计均值。

持续时间可拥有一个可选择持续时间分布属性，其带有识别名称的值和用于相关于过程持续时间的概率分布功能参数。在运行期间，持续时间值通过从持续时间分配这个过程中的采样而分别确定每个过程实例（即，用于每个单独发生的过程）。

注：参见附录D有关过程持续时间和系统运行时间的讨论及实例。

8.5.4.2 超时异常关联

超时异常关联应将源过程与一个超时处理目标过程相连接以明确指定：如果运行期间的源过程实例运行超过其最大持续时间值，则一个事件就将启动目标过程。

从图形上看，一根朝着连接了源和目标的过程线先倾斜，之后又朝目标过程倾斜的单一短棒表示超时异常关联。

鉴于max-durantion（最长持续时间）是Maximal Duration(最长持续时间)的值，且时间单元是一个可允许的时间测量单位，所以超时异常关联的语法应为：如果源过程持续时间超过max-durantion（最长持续时间的时间单位），则超时处理目标过程就会发生。

8.5.4.3 时间不足异常关联

时间不足异常关联应将源过程与一个不足时间处理目标过程相连接以明确指定：如果运行时间的源过程实例的运行少于其最小时间值，则一个事件将启动目标过程。

从图形来看，两根朝着连接了源和目标过程线先倾斜，之后又朝目标过程倾斜的平行短棒表示时间不足异常连接。

鉴于min-durantion(最短持续时间)是Minimal Duration(最短持续时间)的值，并且时间单位是一个可允许时间测量单位，时间不足的异常关联的语法应为：如果源过程持续时间小于最短持续时间的时间单位，则时间不足处理目标过程就会发生。

注：与调用关联相同，两个时间异常关联是直接连接了两个过程的程序关联，而不像大多数将一个对象与一个过程连接的程序关联。事实上，有一个由OPM过程执行机制所创建的临时对象超时异常报文或一个时间不足异常报文，该执行机制意识到过程在最大指派时间内未能结束或还没有到最小指派时间就提前结束。由于OPM运行机制创建并立即消耗这些对象，所以没有必要在模型中对它们进行描述。

9 结构连接

9.1 结构连接类型

结构关联指定了系统中静态、不依赖于时间的持久性关系。一种结构关联将两个或多个对象或两个或多个过程连接起来，但不是将一个对象与一个过程相连接，除非在展示-表征关联（见10.3.3）的情况下。这两种结构关联应是标签结构关联和聚合-分散、展示-表征、泛化-特化和类-实例化这些基本结构关联。

9.2 标签结构关联

9.2.1 单向带标签结构关联

一个单向带标签结构关联应有一个关于从一个事物到另一个事物的关系本质的用户-定义语义。一个以文本短语形式出现具有意义的标签应表达连接对象之间或连接过程之间的结构关系的本质。当放置在OPL句子中时，标签应将该意思表达出来。

从图形上看，一个带有以开放箭头的箭状符号和一个位于柄旁边的带标注注释应表示一个单向标标签结构关联。

单向带标签结构关联OPL句子的语法应为：源-事物加标签于目标-事物。

注：由于标签是建模者添加到模型的一个标记，所以在OPL句子中标签短语以粗体形式出现以便与其隐式于句法结构中的其它词句区分开来。

9.2.2 单向空标签结构关联

单向零标记结构关联应是一个不具备标签附注的单向带标签结构关联，表示使用了默认单向标签。默认标签应是 “与……相关” 。

单向零标记结构关联OPL句子的语法应为：源-事物与目标-事物相关。

注：建模者有权为一个特定系统或一组系统来选择未以粗体字母出现的设置默认单向标签。

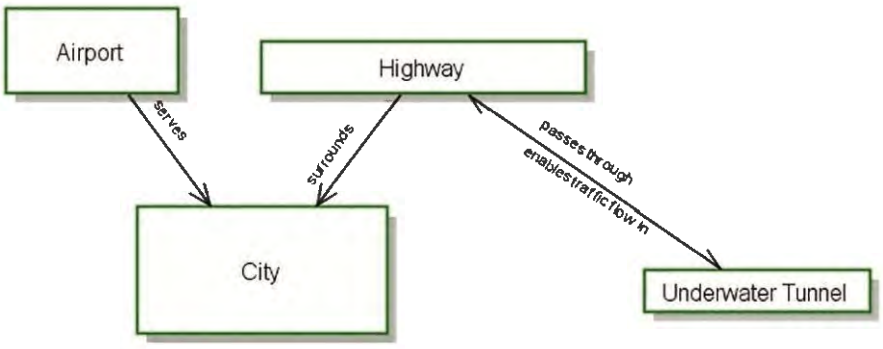
9.2.3 双向带标签结构关联

因为事物之间的关系是双向的，所以每个标签结构关联在相反方向都具有相应的标签结构关联。当处在两个方向中的标签具有意义且不为彼此的逆向时，它们可由置于一个单一双向带标签结构关联任何一侧的两个标签来进行注释。

从图形上看，一条在连接两端的相对两侧带有鱼叉形状的箭状符号的线表示一个双向带标签结构关联。每个标签应与伸出箭头外的带有鱼叉边缘的箭状符号一侧对齐，毫无它义地明确其中每个关系所使用的方向。

所产生的标签结构关联的语法应是两个分开的单向带标签结构关联的OPL句子，每一个句子用于每一个方向。

示例：图 14 显示了标签结构关联的两种类型。



飞机场服务于城市；

高速公路环绕城市；

高速公路通过水下通道；

水下通道使得高速公路中的交通移动。

图14 两种标签结构关联类型

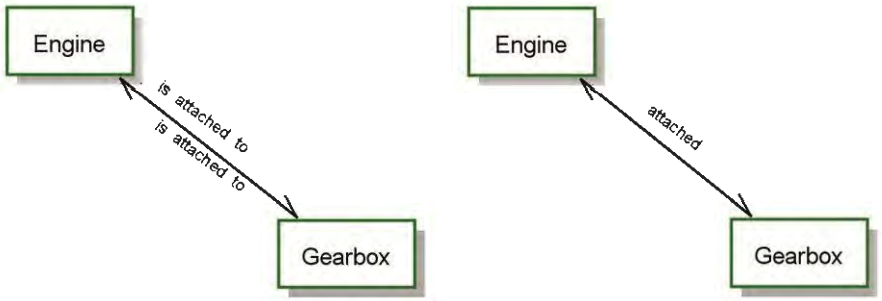
9.2.4 相互关联标签结构关联

相互关联标签结构关联应是一个只带有一个标签或无标签的双向带标签结构关联。在任何一种情况下，相互关联性应表明，一个具有双向结构关联的标签对于每一个关系的方向都拥有相同的语义。当无标签出现时，默认标签应是 “有关联的 ”。

仅带有一个相互关联标签结构关联的语法应是：源-事物和目标事物都是相互关联-标签。

不带标签的相互关联标签结构关联的语法应是：源-事物和目标-事物是相互关联的。

示例：在图 15 中，右侧是相互关联结构关联，等同于左侧在每个方向都拥有同样标签的双向带标签结构关联。



发动机附连到齿轮箱。
齿轮箱附连到发动机。

发动机和齿轮箱互连。

图15　双向（左侧）和其等效相互关联标签结构关联（右侧）

注：如图15所示，来自双向带标签结构关联的动词或名词形式中的一种变化通常对于满足相互影响的带标注结构关联的语法是有必要的。

9.3　基本结构关系

9.3.1　基本结构关系类型

基本结构关系是OPM事物中最常见的结构关系，并且对于确定和理解系统具有特别意义。每一个基本关系都应详细阐述或将一个源事物，即可细化物细化为一个或多个目标事物，即细化物的集合体。

基本结构关系应是：

——聚合-分散，指定了一个整体与其部分之间的关系；

——展示-表征，指定了一个展示物，即展示了一个或多个特点（属性和/或运行）的展示物与展示了展示物特性的事物之间的关系；

——泛化-特化，指定了一个一般事物和其特化之间的关系； 以及

——类化-实例化，指定了一个事物类与和该类的一个细化物实例之间的关系。

聚合、展示、泛化和类化应是细化关系标识符，即与关系相关的标识符，正如我们从可细化物的角度所看到的那样。分散、表征、特化和实例化应是相应的互补关系标识符，即从其细化物的角度所看到的关系标识符。

除了展示-表征外，细化物目标事物应全部具有与可细化源事物相同的韧性，即或全都是具有静态韧性的对象或全都是具有动态韧性的过程。

将细化物折叠应是隐藏了一个可细化物的那些细化物，将可细化物展开应是表达该可细化物的细化物（见13.2.1.2）。

由于基本结构关系是双向的，相关的OPL段落可以为每个方向提供句子。然而，由于这些句子中的其中一个句子总是另一个句子的结果，所以一个基本结构关系的OPL表达式应限于这两个可能句子的其中之一。每一类基本结构关系的表达式包括只用于两种可能性句子之一的默认OPL句子的规范。表14总结了这些默认句子。

用于OPD中一些可细化物建模的细化物的集合体也许是完整或不完整的，即在图形图所明确描绘且相应文本也清晰地表达的那样，仅那些结构关联出现在其中的相关OPD的事物。

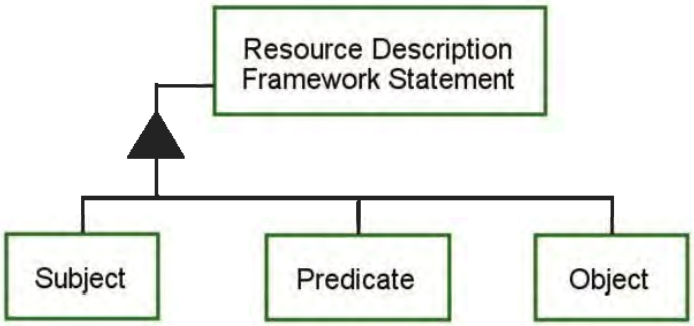
9.3.2　聚合-分散关系关联

基本结构关系聚合-分散应表示一个可细化物，一个整体将一个或多个细化物，即部分聚合在一起。

从图形来看，一个黑色固体（填充）三角形的顶点由一条线连接到整体以及部分由若干条线连接到相反的水平底边应表示聚合-分散关系关联。

聚合-分散关系关联的语法应为：整体-事物由部分-事物1、部分-事物2、...，以及部分-事物n组成。

示例1：



资源描述框架陈述由主语、谓语和宾语组成。

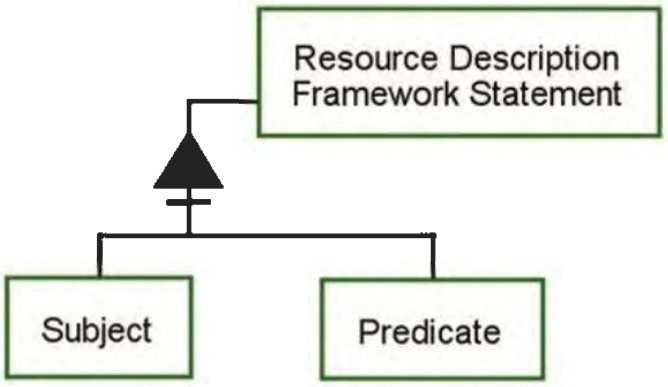
图16　聚合-分散关系关联

当处于特定范围细节中的部分集合体的表达式是不完整时，聚合-分散关系关联应使用一个注释来表示不完整表达式。

从图形上看，一个位于黑色三角形下方横跨垂直线的短平棒应表示不完整的聚合-分散关系关联。

表示部分的一个局部集合体、其中至少遗漏了一个部分的聚合-分散关系关联的语法应是：整体-事物由事物1部分、事物2部分、...事物k部分以及至少一个其它部分组成。

示例2：在图 17 中，来自图 16 中的对象缺失。处于黑色三角形下方横跨垂直线的短平棒表示所缺失的事物。



资源描述框架陈述由主语、谓语和宾语以及至少另一个部分组成。

图17　带有部分细化物集的聚合-分散关系关联例子

示例3：例 3 在图 18 中的左侧，消耗过程消耗了包含其 B 部分和 D 部分的整体，而 A 部分和 C 部分仍作为单独的对象而保留了下来。在图 18 中的右侧，使用了部分聚合的精简版本显示消耗过程所消耗的整体以及仅有的 B 部分和 D 部分，而整体的其它部分作为独特的对象而保留了下来。

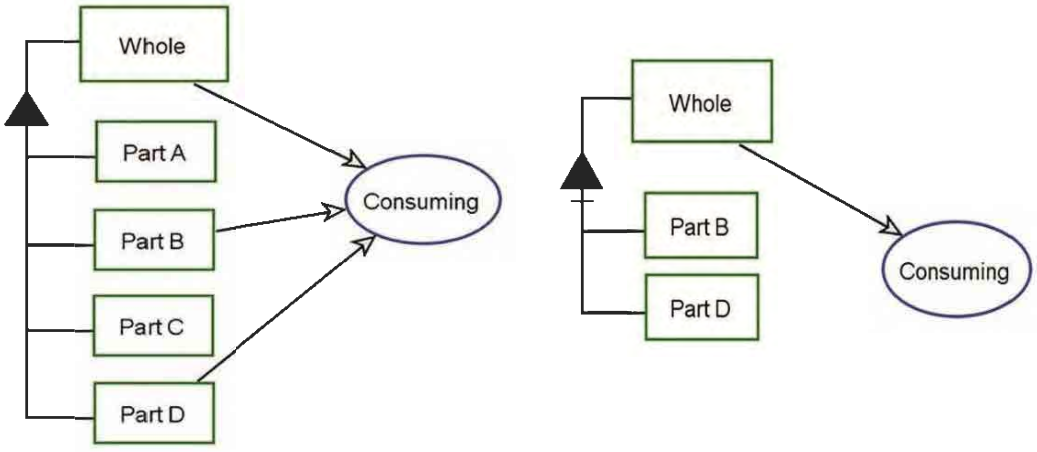


图18　部分聚合消耗

注：一个工具可在建模者更改细化物集时跟踪每个可细化物的细化物集并调整符号以及相应OPL句子（每个基本结构关系关联均已在下面进行了指定）。

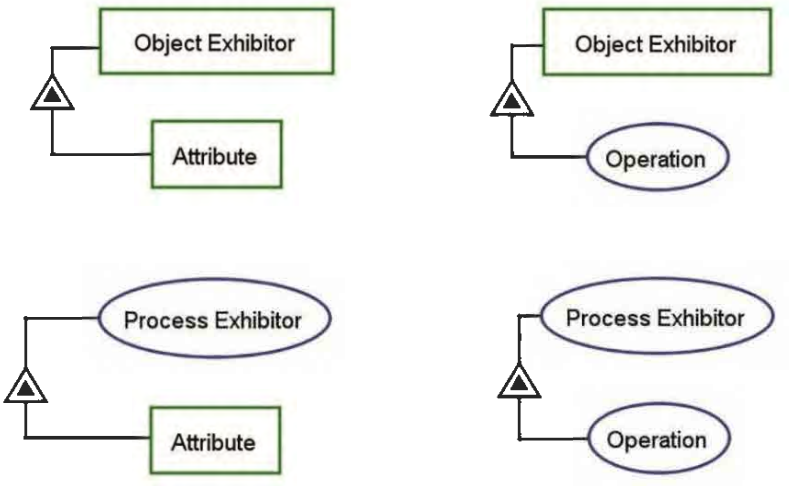
9.3.3　展示-表征关联

9.3.3.1　展示-表征关系关联表达

基本结构关系的展示-表征是指一个可细化物，即展示物，展示了一个或多个表征展示物，即细化物的特性。特性应表征展示物的特性。

一个特性应是一个事物。一个属性应是一个对象的特性。一个操作应是一个过程的特性。一个过程展示物和一个对象展示物每个都应具有至少一种特性，并且有可能具有两种属性，它们的对象特性和操作以及它们的过程特性。

展示-表征关系可以将对象和过程的四种展示物的特性组合（参见图19）结合在一起。



对象展示物展示属性。　　对象展示物展示操作。

过程展示物展示属性。　　过程展示物展示操作。

图19　四种展示-表征特性组合

从图形上看，一个置于较大黑色空心三角形、其顶点被一条线连接到展示物的小型黑色三角形，其属性连接到相反的（水平）底边，应意味展示-表征关系关联（见图19）。

用于一个具有n属性和m操作的全部集合体的对象展示物的展示-表征关系关联的语法应是：对象-展示物展示属性1，属性2，...和属性n以及操作1，操作员2，.....和操作员m。

用于一个具有n操作特性和m属性特性的全部集合体的过程展示物的展示-表征关系关联的语法应是：过程-展示物展示了操做1、操作员2,.....和操作员n以及属性1，属性2，… and 属性m。

注1：在用于展示-表征的 OPL 中，对于一个对象展示物来说，属性列表位于操作列表之前，而对于一个过程展示物来说，操作列表处于属性列表之前。

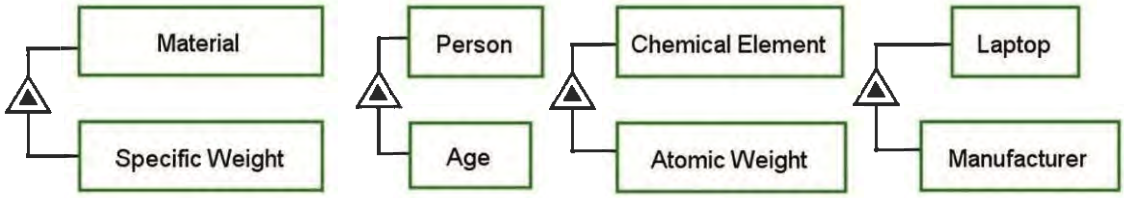
当处于特定细节范围的特性集合体的表达为不完整时，展示-特性关系关联应意味着带有一个注释的不完整表达式。

从图形上看，一个穿越位于较大空三角性下方垂直线的短横杠表示不完整的展示-表征关系关联。

用于一个具有j属性特性和k操作特性的局部集合体的对象展示物的展示-表征关系关联的语法应是：对象-展示物-事物展示属性1、属性员2...，属性员j和至少一个其它属性以及操作1、操作员2，…，操作员k,和至少另一个操作。

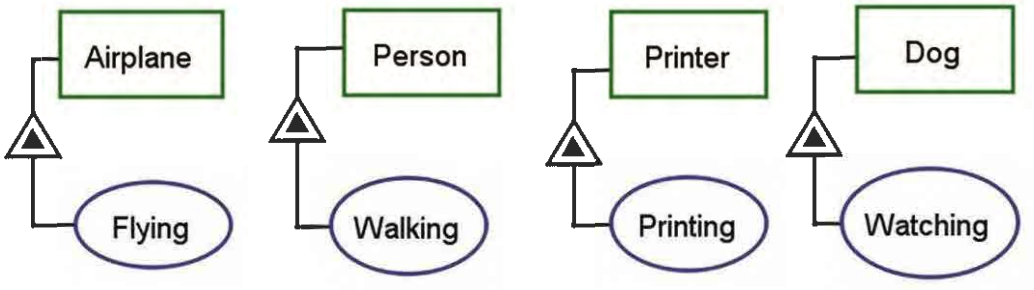
用于一个具有j操作特性和k属性特性的局部集合体的过程展示物的展示-表征关系连接的语法应是：对象-展示物展示了操做1、操作员2，...、操作员j和至少另外一个操作以及属性1、属性员2、…、属性员k,和至少另外一个属性。

示例：图 20 到图 23 展示了对象和过程的四种展示物-特性组合。



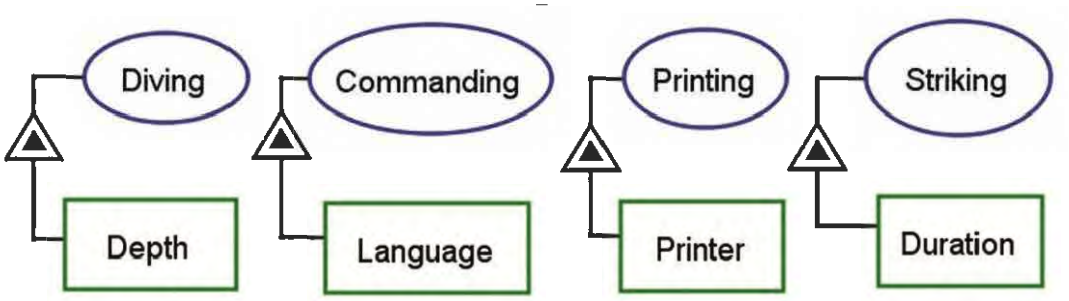
材料展示了特定重量。　人展示了年龄。　化学元素展示了原子量。　笔记本电脑展示了制造商。

图20　对象属性例子



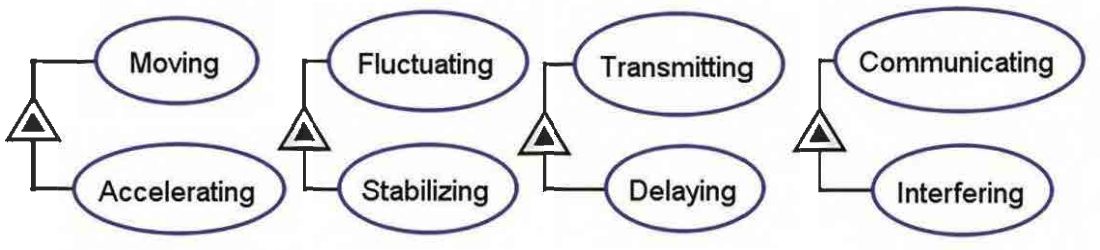
飞机展示飞行。　人展示步行。　打印机展示打印。　狗展示观看。

图21　带有操作例子的对象展示物



跳水展示深度。　　命令展示语言。　　印刷展示打印机。　　击打展示持续时间。

图22　带有属性例子的过程展示物



移动展示加速。　　波动展示稳定。　　传输展示延迟。　　通信展示干扰。

图23　带有操作例子的过程参展物

注2：一个工具可在建模者更改细化物集合体时跟踪每个可细化物集并调整符号和相应的 OPL 句子（以下指定了每个基本结构关系关联）。

9.3.3.2　属性状态和展示物特性

9.3.3.2.1　作为值的属性状态

一个属性状态，即细化物属性的对象状态应是该属性的一个值。静态概念模型应确定该属性所有可能的值。有些也许是取值范围，而动态的运行实例模型应表明在检查属性时的实际属性值（见9.3.5.1中例子1和2）。

9.3.3.2.2　展示物-特性关系表达

当表达一个属性的特性或值时，该模型应识别具有该特性或值的展示物。为指定展示物的特性，关系 “的” 应在特性与其展示物之间的OPL句子中出现。

一个识别展示物-特性的OPL句子的语法应是：展示物的特性。。。。

示例1：通过其金属粉末混合展示物来显示属性具体重量所有权的 OPL 句子为：金属粉末混合物的具体重量范围从7.545 到 7.537 克/每立方米。

示例2：图 25 中的 OPL 句子通过其船舶展示物所显示的旅行介质属性的所有权为：船舶的旅行介质是水表面。

9.3.4　泛化-特化和继承性

9.3.4.1 泛化-特化关系关联

基本结构关系的泛化-特化应意味着一个可细化物，即一个通用类，将一个或多个通用类的特化细化物进行概括。这种泛化-特化关系将一个或多个具有韧性的特化作为通用类而绑定在一起，这样，通用类和其全部特化都是对象或通用类和其全部特化都是过程。

从图形来看，一个顶点被一条线连接到通用类的三角形和由几条线连接到相反底边的特化应表示泛化-特化关系关联（见图24）。

对于一个对象通用类的一个完整n个特化集合体来说，泛化-特化关系关联OPL句子的语法应为：特化-对象1、特化-对象2.....、以及特化-对象n是通用类-对象。

对于一个过程通用类的一个完整n个特化集合体来说，泛化-特化关系关联OPL句子的语法应为：特化-过程1，特化-过程2，…、和特化-过程n是通用类-过程。

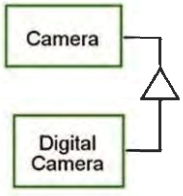
当一个特定细节范围内的特化集合体的表达式为不完整时，泛化-特化关系关联应意味着一个带有注释的不完整表达式。

从图形上看，一个位于空三角形的跨越垂直线的短平棒表示不完整的泛化-特化关系关联。

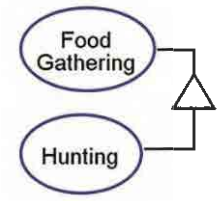
对于一个对象通用类的k个特化的一个不完整集来说，泛化-特化关系关联的OPL句子的语法应是：特化-对象1、特化-对象2、。。。特化-对象k,以及至少一个其它特化是通用类-对象。

对于一个过程的通用类的k个特化的不完整集来说，泛化-特化关系关联的OPL句子的语法应为：特化-过程1、特化-过程2,.....、特化-过程k以及至少一个其它特化是通用类-过程。

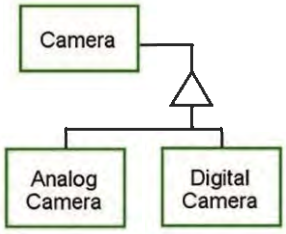
示例：图 24 显示了对象和过程的单个和复数的特化。



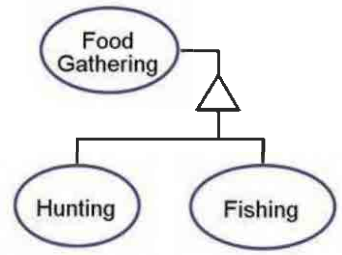
数码相机是一个相机



打猎是采集食物



模拟相机和数码相机是相机



打猎和钓鱼是采集食物

图24 对象和过程的单个和复数特化

注：一个工具可以在建模者更改细化物集合体时为每一个可细化物跟踪细化物集并为每一个基本结构关系关联调整符号及相应的OPL句子。

9.3.4.2 通过特化的继承

继承应是一个通用类到其特化的OPM要素、事物和关联。

一个特化事物应是通过泛化-特化关联从通用事物中继承以下存在的四种可继承要素的其中之一种：

——来自于其聚合-分散关联的一个通用类的所有部分；

——来自于其展示-表征关联的一个通用类的全部特性；

——通用事物所连接的所有标签结构关联； 以及

——通用事物所连接的所有程序关联。

OPM应通过允许一个事物去继承来自多个通用事物的每个细化物为多重继承性提供机会-四种为该通用事物而存在的可继承要素（分散物、特征、标签结构关联及程序关联）。

建模者可通过指定来自一个通用类，即一个带有不同名称和不同状态集（见9.3.4.3）的该分散物的特化来覆盖任何被特化所默认地继承下来的通用事物的分散物（见9.3.4.3）。

注：当一个泛化-特化关系关联存在时，运行时的特定事物实例在缺少其所指定的更多通用事物实例以及它从中所继承的四种可继承要素之一时是不存在的。

为创建一个从一个或多个候选特化的通用类,通用于每一个候选特化的可继承要素应被迁移到一个泛化的事物中去。可继承要素的操作应按如下所示：

——将特化的所有共同特性和共同分散物组合到一个新创建的通用类中去；

——使用泛化-特化关系关联将新的通用类连接到特化中去；

——删除所有特化当前从新通用类所继承的共同特性和共同参与；以及

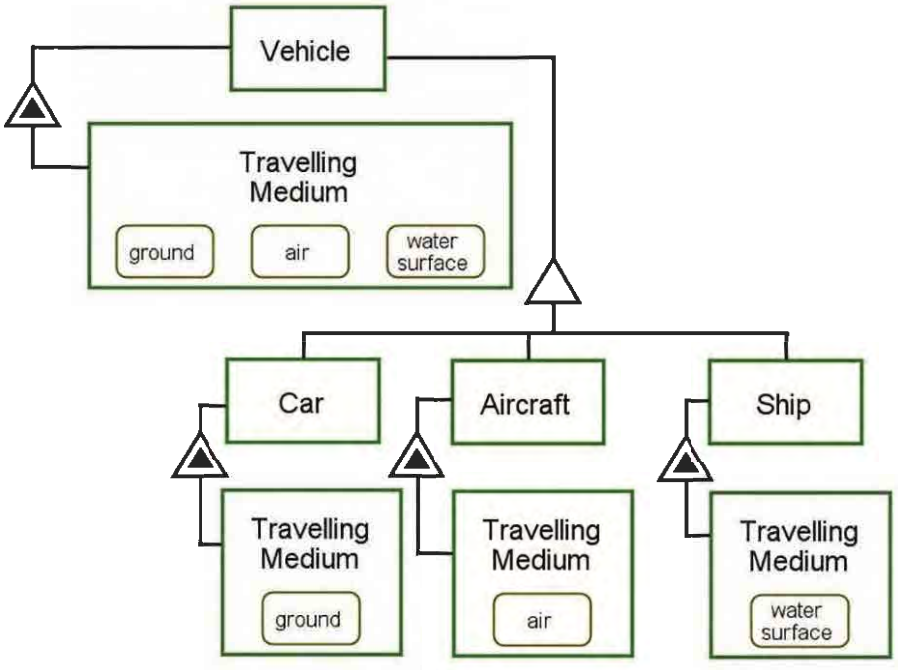
——将任何共同标签结构关联和连接到所有特化的共同程序关联边缘从特化迁移到通用类。

9.3.4.3 通过区分属性的特化限制

从一个通用类所继承的属性可取值有可能会限制一个特化的允许值。一个具有约束相应特化表征不同值的继承属性应是一个可识别属性。

注：一个特化继承了其泛化的特性和可能的属性值。通过细化对通用类进行详细阐述使得对所继承的属性可以进行更精确的估价，包括那些适合于通过其所继承的展示-表征细化的属性值特化（见9.4.1）。

示例1：图 25 显示了一个 OPD 中车辆所展示的移动介质属性，具有的值包含地面、空气和水表面。移动介质是车辆的可识别属性，因为它将车辆的特化约束到其移动介质的值。车辆拥有特化汽车、飞机和船舶，并带有相应的移动介质值，地面、空气和水表面。



车辆展示移动介质；

车辆的移动介质 可以是地面、天空和水面；

汽车、飞机和船舶是车辆；

车辆的移动介质是地面；

飞机的移动介质是天空；

船舶的移动介质是水面。

图25　识别属性移动介质和其特化

一个通用类可以有多个识别属性。具有一个以上识别属性的特化的最大数目应是每一个识别属性的可能取值数量的笛卡儿积，其中一些属性值的组合也许是无效的。

示例2：如果将图 25 的内容扩展的话，车辆的另一个属性可能是具有民用和军用这两个值的用途。基于这两个值就有两种车辆的特化：民用车辆和军用车辆。由于多重继承，其结果是一个继承格，其中最具体的特化数量将是如下所示的 3×2= 6：民用汽车、民用飞机、民用船舶、军用汽车，军用飞机和军用船舶。

9.3.5　类化-实例化关联

9.3.5.1　类化-实例化关系关联

基本结构关系的类化-实例化意指一个可细化物，即类，将一个或多个细化物，即类的实例进行分类。作为一个对象类或过程类的类化是一个用于与一个或多个目的事物相连接的事物的源模式，这些事物也就是源事物模式的实例，即模式所指定的质量要求具有清晰的值去对实例事物进行类化。这种关系为建模者提供了一个明确的机制来表达取值创建的一个类与其实例之间的关系。

注1：在考虑一个概念类的实例集的成员时，使用术语实例被称之为“细化物实例”以将它们与一个操作模型的“运行实例”加以区分。每一个细化物实例都拥有一个或多个可能性操作实例。

注2：在一个概念模型中所表达的所有 OPM 事物都是一个在模型评估或运行期间所预期将会出现的那个事物实例的类模式。通过在概念模型中建立一个事物，建模者所隐喻的是该事物的至少一个操作实例或该事物的一个特化可在系统运行过程中的某一时刻存在。

如果类模式包括一个指定了具有一个可允许取值范围的细化物属性的展示-表征关联的话，那么该类的一个细化物实例的每个操作实例相应属性值都应处于其类属性特性取值范围的标准之内。

从图形上来看，一个处于较大空三角形中、顶部由一条线连接到一个类事物的较小黑色圆圈与被几条线连接到对面（平行）底边的实例事物应表示类化-实例化关系关联。

一个对象类与一个单个实例之间的类化-实例化关系关联的语法应为：实例-对象是一个类-对象的实例。

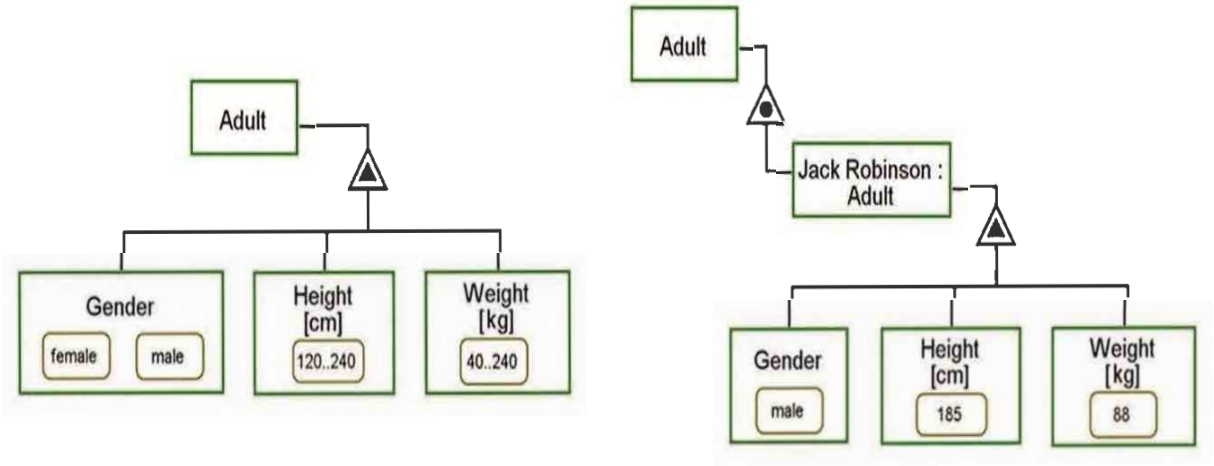
一个过程类和一个单个实例之间的类化-实例化关系关联的语法应为：实例-过程是一个类-过程的实例。

一个过程类与n个实例之间的类化-实例化关系关联的语法应是：实例-对象1、实例-对象2和实例-对象n是类-对象的实例。

一个过程类和n个实例之间的类化-实例化关系关联的语法应是：实例-过程1、实例-过程2和实例-过程n是类-过程的实例。

注3：由于任何类的实例数量可能是一个未知的先验，并可在系统操作期间发生变化，所以对于类化-实例化关系而言目标事物的完整与和不完整集合体之间是没有明显区别的。

示例1：图 26 中的成人是一个具有三种属性的类：性别、带有可能值男性和女性和以厘米为单位、拥有可能值 120..240 的高度以及以公斤为单位、拥有可能值 40..240 的体重。杰克·罗宾逊是成人的一个实例，具有性别值男性，身高值是 185 厘米及体重值是 88 公斤。



成人展现了性别、厘米身高和公斤重量。

成人的性别可以是男性或女性。

成人的身高以厘米为单位，范围从 120 到 240 厘米。

成人的体重以公斤为单位，范围在 40～240 公斤。

杰克·罗宾逊是成人的一个实例。

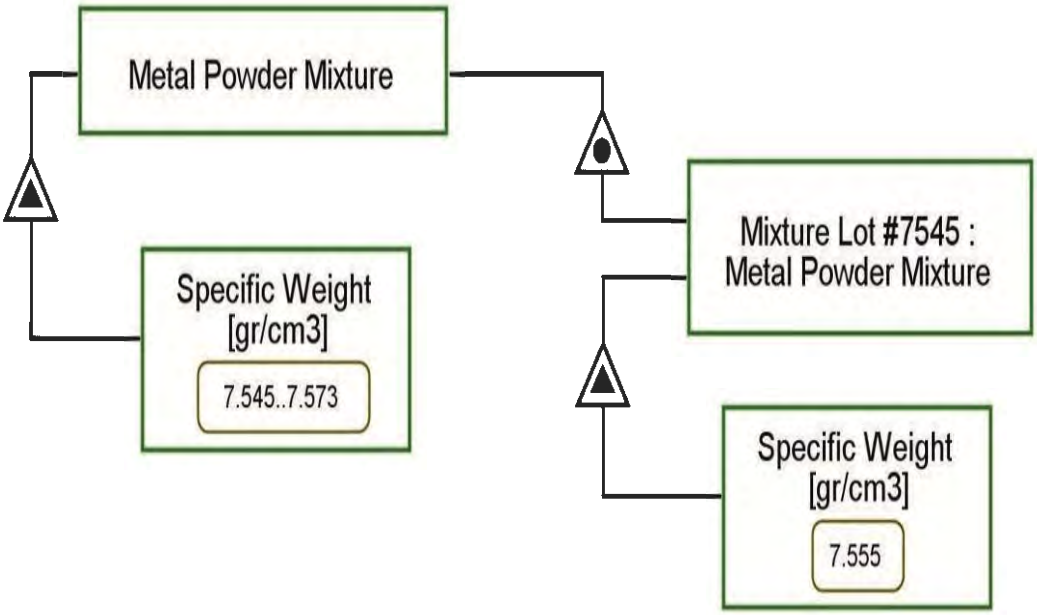
杰克·罗宾逊的性别为男性。

杰克·罗宾逊的身高为 240.185 厘米。

杰克·罗宾逊的体重为 88 公斤。

图26　具有取值范围的类化-实例化 （类在左侧，实例在右侧）

示例2：图 27 左侧的 OPD 是金属粉末混合物的一个概念模型，表明其指定重量属性值的范围可从每立方厘米 7.545 克到 7.537 克。图 27 是一个金属粉末混合物实例的操作实例（运行时）模型，表明其指定重量属性值为每立方厘米 7.555 克。这个值处于允许范围之内。



金属粉末混合物 展示了以每立方厘米每克为单位的指定重量；

以每立方厘米每克为单位的金属粉末混合物的范围从7.545 to 7.537；

混合物段#7545 是一个金属粉末混合物的实例；

混合物段#7545的以每立方厘米每克为单位的指定重量是 7.555。

图27　作为值的属性状态：概念模型与操作模型

注4：OPL 句子 “混合物段＃7545 展示了以每立方厘米每克的指定重量” 没有出现在图 27 的 OPL 中，因为这句话隐喻地表达了“混合物段＃7545 是金属粉末混合物的一个实例”的这样一个事实，因此，混合物段＃7545 从金属粉末混合物中继承了这个属性。

9.3.5.2　对象类和过程类实例

一个对象类和过程类是两种不同类型的类。一个类的实例应是一个该类具有相同类化标识符类的特定可识别实例的化身。

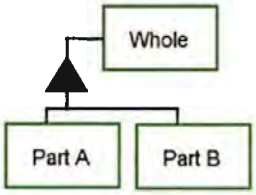
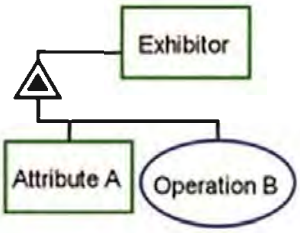
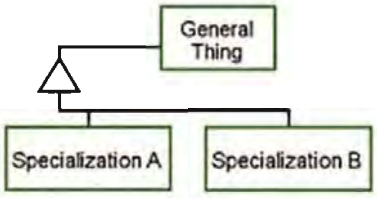
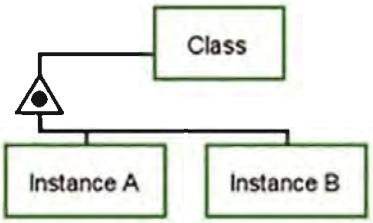
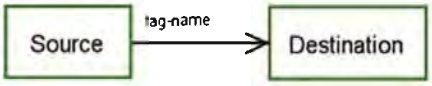
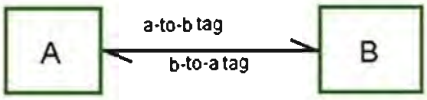
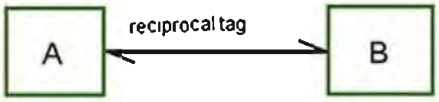
一个单个细化物对象应是一个对象实例，而所有实例所依附的对象模式应是一个对象类，即可细化物。

一个过程类应是一个正在发生（子过程序列）的模式，它涉及到术语前过程和后过程对象集成员的对象类。一个遵循了这种模式并在其前过程和后过程对象集中涉及了特定对象实例的过程发生应是一个过程实例。因此，一个过程实例应是一个该实例所属的过程类的具体现象。任何过程实例都应具有一组相应的前过程和后过程对象实例集的不同集。

注：过程类概念的强大之处在于它能使一个过程的建模可作为一个对象类经历某些转换的模板或协议。这种转型既不包括时空框架，也不包含过程实例相关联的对象实例的特定集。

9.3.6　结构关系关联和带标签结构关联小结

表14　结构关系和关联小结

结构关系前推-倒退（可细化物到细化物；粗体为简称）	OPD 符号	OPL 句子	
		前推 可细化物到细化物	倒退 （细化物到可细化物
聚合-分散		整体由 A 部分和 B 部分组成。	–
展示-表征		展示者展示了属性 A 及操作 B.	–
泛化-特化		–	特化 A 和特化 B 是通用物。
类化-实例化		–	实例 A 和实例 B 是类的实例。
单向标签 ［单向空标签］		源标签-名称目标。 ［源与目标相关。］	
双向标签		**A** a-到-b 标签 B. **B** b-到-a 标签 A.	
相互关联标签 ［相互关联空标签］		A 和 B 是相互关联标签。 ［A 与 B 相关。］	

9.4 状态-指定结构关系和关联

9.4.1 状态-指定表征关系关联

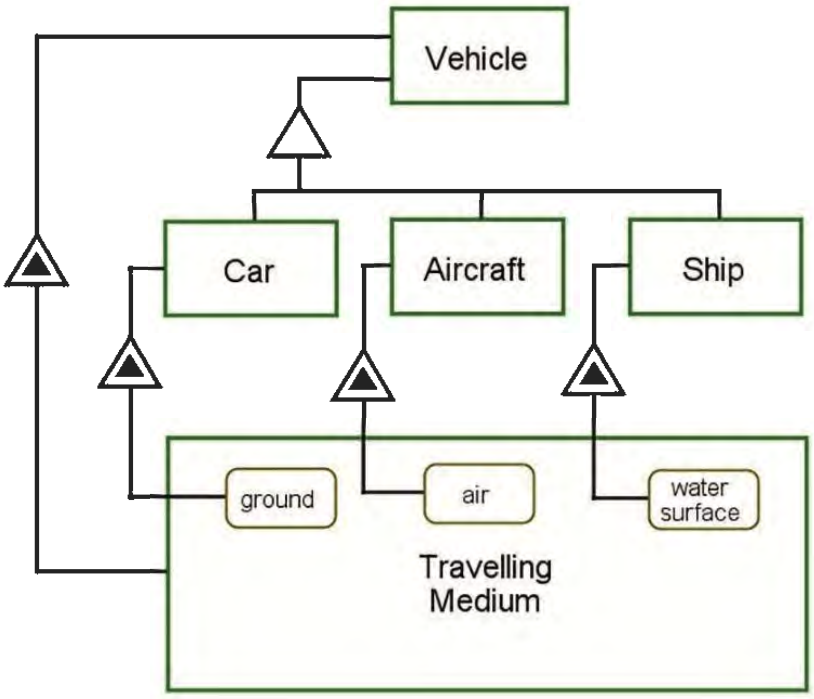
一个状态-指定表征关系关联应是一个来自于一个展示属性值以区分其泛化属性的特定对象的展示-表征关系关联，意味着特定对象应只拥有其所继承属性的那个值。

从图形上看，顶部连接到指定的对象和向对面的底边连接到一个显示为状态值的展示-表征关系关联三角型符号表示状态-指定表征关系关联。

注：尽管没有必要，但如果具有识别属性的通用类的展示-表征关联也同样出现在OPD中的话，OPD会变得更易于理解（见图28）。

状态-指定表征关系关联的语法应为：特定-对象展示了值-名称属性-名称。

示例：使用状态-指定表征关系关联，图 28 中的 OPD 比其在图 25.中相等的 OPD 明显地更为紧凑。这里，识别属性车辆的运行介质具有的值地面，空中和水面仅出现了一次，相对于图 25 中所出现的四倍。汽车、飞机和船舶的模型是车辆的特化，将每一个带有一个状态-指定表征关系关联的特化分别连接到相应的地面、空中和水面的运行介质中去。



车辆展示了运行介质；

车辆的运行介质 可以是地面、空中和水面；

汽车、飞机和船舶是车辆；

汽车展示了地面运行介质；

飞机展示了空中运行介质；

船舶展示了水面运行介质。

图28 状态-指定表征关联例子

9.4.2 状态-指定的带标签结构关系

9.4.2.1 状态-指定的带标签结构关联

一个状态-指定加标签关联应是一个对象状态或属性值与另一个对象、对象状态或属性值之间的标签结构关联，意味这两个带有表达关系语义标签的事物之间的一种关系。在一个空标签的情况下，即没有明确标签规范的情况下，相应的OPL应使用默认空标签（参见9.2.2）。

应有三种状态-指定带标签结构关联存在：源状态-指定带标签结构关联；目标状态-指定带标签结构关联和源-和-目标状态-指定带标签结构关联。每一种关联应包括单向、双向和相互关联标签结构关联，由此而产生出七种状态-指定带标签结构关系关联以及相应的OPL句子，如表15所示。

9.4.2.2 单向源状态-指定的加标签结构关联

一个单向源状态-指定带标签结构关联应是一个从一个源对象的特定状态到一个没有状态指定的目标对象的单向带标签结构关联。

从图形来说，一个从源对象状态连接到目标对象的带有开放箭头的箭状符号与一个柄旁边的标签名称注释应表示一个单向源状态-指定带标签结构关联。

单向源状态-指定带标签结构关联的OPL句子的语法应为：指定-状态 源-对象 标签-名称 目标-对象。

注：一个空标签使用了默认标签-名称“和……相关”，非粗体，除非建模者进行了修改。

9.4.2.3 单向目标状态-指定带标签结构关联

一个单向目标状态-指定带标签结构关联应是一个来自无状态的源对象到目标对象的一个特定状态的单向带标签结构关联。

从图形上看，一个从一个源对象连接到目标对象的一个特定状态的带有开放箭头的箭状符号与一个轴旁边的标签-名称注释应表示一个单向目标状态-指定带标签结构关联。

单向目标状态-指定带标签结构关联的OPL句子的语法应为：源-对象 标签-名称指定-状态 目标-对象。

注：一个空标签使用默认标签-名称“和。。。相关”，非粗体，除非建模者进行了修改。

9.4.2.4 单向源和目标状态-指定带标签结构关联

一个单向源和目标状态-指定带标签结构关联应是一个来自源对象的特定状态到一个目标对象特定状态的单向带标签结构关联。

从图形上看，一个从一个源对象的特定状态连接到一个目标对象的特定状态的带有一个开放箭头的箭状符号与一个柄旁边的标签-名称注释应表示源-和-目标的单向、指定状态的带标签结构关联。

单向源-和-目标状态-指定带标签结构关联的OPL句子的语法应为：源-指定-状态 源-对象 标签-名称 目标-指定-状态 目标-对象。

注：一个空标签使用默认标签-名称“和。。。有关”，非粗体，除非建模者将其修改。

9.4.2.5 双向源-或-目标状态指定的带标签结构关联

一个双向源-或-目标指定状态的带标签结构关联应是一个用于源或目标对象但并非两者兼具的、具有一个特定状态的双向带标签结构关联。

从图形来看，一条处于关联的相对两端的鱼叉形箭头的线，一头连接到一个对象或对象状态，另一头分别连接到对象状态或对象。它应表示的是一个双向带标签结构关联。每个标签-名称应与伸出箭头外的鱼叉边缘的箭状符号一侧对齐，无二义性地确定每个关系对其所适用的方向。

所获得的双向源-或-目标的指定状态的带标签结构关联的语法应是两个独立的单向带标签结构关联的OPL句子，每一个方向都有相应状态说明。

9.4.2.6 双向源-和-目标状态-指定带标签结构关联

一个双向源-和-目标状态-指定带标签结构关联应是一个用于源和目标对象两者的具有一个特定状态的双向带标签结构关联。

从图形来看，一条处于关联的相对两端的鱼叉形箭头的线，将一个对象的指定状态连接到另一个对象的指定状态，应应表示一个双向带标签结构关联。每个标签-名称应与伸出箭头外的鱼叉边缘的箭状符号一侧对齐，无二义性地确定每个关系对其所适用的方向。

所获得的双向源-和-目标状态-指定带标签结构关联的语法应为两个独立的单向源-和-目标标签结构连接的OPL句子，每个句子使用相应状态规范和标签-名称来用于一个方向。

9.4.2.7 相互关联的源-或-目标状态-指定带标签结构关联

一个相互关联的源-或-目标带标签结构关联应是一个具有用于所涉及到对象之一但非两者兼顾的、且只有一个相互关联-标签或无标签的特定状态的双向源-或-目标结构关联。在任何一种情况下，相互关联性应表示一个相互关联的源-或-目标状态-指定带标签结构关联的标签对于关系的每一个方向都具有相同的语义。当没有标签出现时，则默认标签应是“与。。相关” 。

从图形上看，一条位于关联的两个相对端点、拥有鱼叉状箭头的线将一个对象的特定状态连接到另一个没有状态规范的对象并且仅仅描述了与箭头对齐的一个标签-名称应表示一个相互关联的源或目标状态-指定带标签结构关联。

仅带有一个标签的相互关联的源-或-目标状态-指定带标签结构关联的语法应为：源-指定-状态 源-对象以及目标-对象是相互关联标签；或为,源-对象和目标-指定-状态 目标-对象是相互关联标签。

9.4.2.8 相互关联的源-和-目标状态-指定带标签结构关联

一个相互关联的源-和-目标带标签结构关联应是一个双向源-和-目标带标签结构关联,其具有一个用于两个所涉及对象的特定状态并拥有一个相互关联标签或无标签。在任何一种情况下，相互关联性应表示一个相互关联的源-和目标状态-指定带标签结构关联的标签对关系的每一个方向都具有相同语义。当标签未出现时，默认标签应是“与。。有关”。

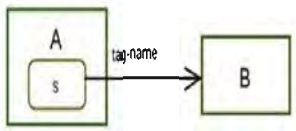
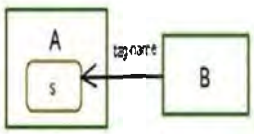
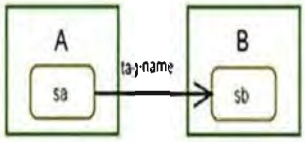
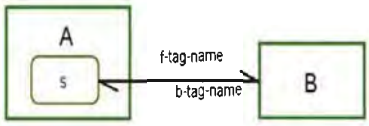
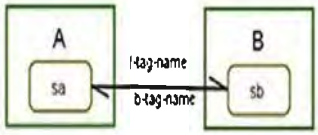
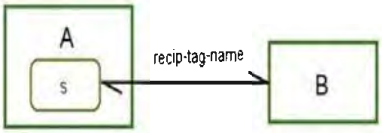
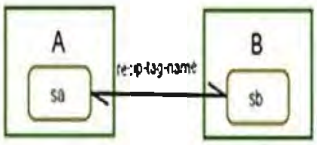
从图形来看，一条在关联的两个相对终端、将一个对象的指定状态连接到另一个对象的指定状态并仅描绘了一个与箭头对齐的标签-名称的鱼叉型箭头的线应表示一个相互关联的源-或-目标状态-指定带标签结构关联。

仅带有一个标签-名称的相互关联的源-和-目标状态-指定带标签结构关联的语法应为：源-指定状态 源-对象和目标-指定-状态 目标对象是相互关联标签。

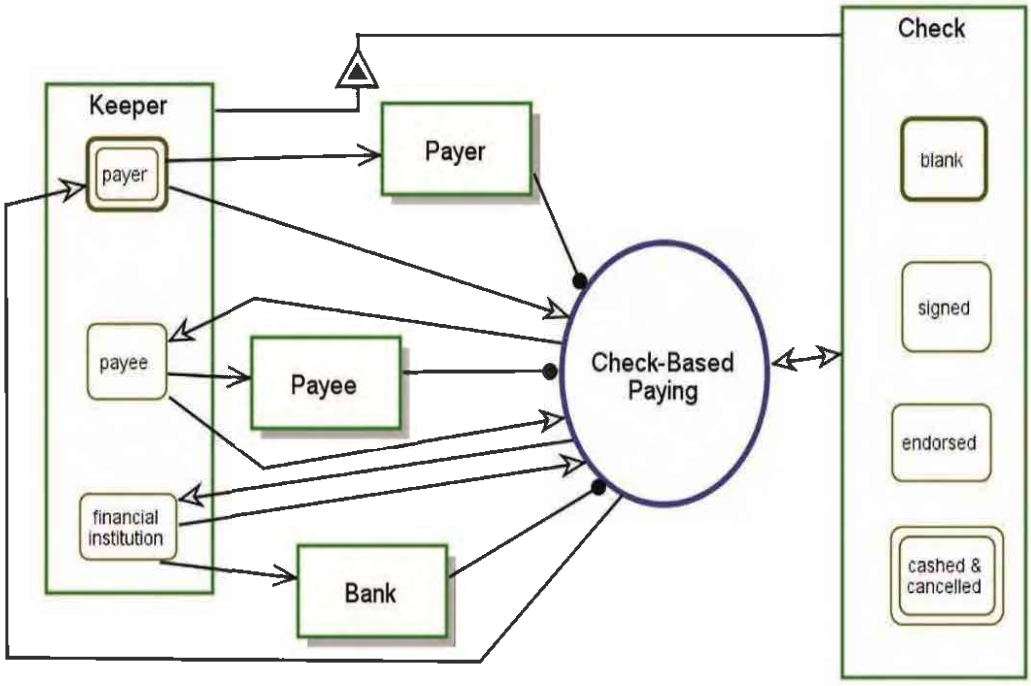
没有标签名称的相互关联的源-和-目标状态-指定带标签结构关联的语法应为：源-指定状态 源-对象和目标-指定-状态 目标-对象是相互关联的。

9.4.2.9 状态-指定带标签结构关联小结

表15　状态-指定结构关系和关联小结

有向性	来源/目标		
	源状态-指定	目标状态-指定	源-和目标状态-指定
单向	 S A 标签-名称 B.	 B 标签-名称 s A.	 Sa A 标签-名称 sb B.
双向	 S A f-标签-名称 B. B b-标签-名称 s A.		 Sa A f-标签-名称 sb B. Sb B b-标签-名称 sa A.
相互关联	 B 和 s A 是相互关联-标签-名称。		 Sa A 和 sb B 是相互关联-标签-名称。

示例1：图 29 的 OPD 中，保管人是支票的属性，具有的值为付款人、收款人和银行。因此，这些值的每一个值也是模型中的一个对象本身。三个单向的源-状态-指定空标签结构关联将每一个值连接到其所对应的对象上去。要注意的是，并没有要求状态或值的名称要与金融机关和银行所展示的那样与相关对象的名称相同。



支票可以是空白、签字、背书或现金兑现和取消；

支票展示了保管人；

保管人可以是付款人、收款人或金融机构；

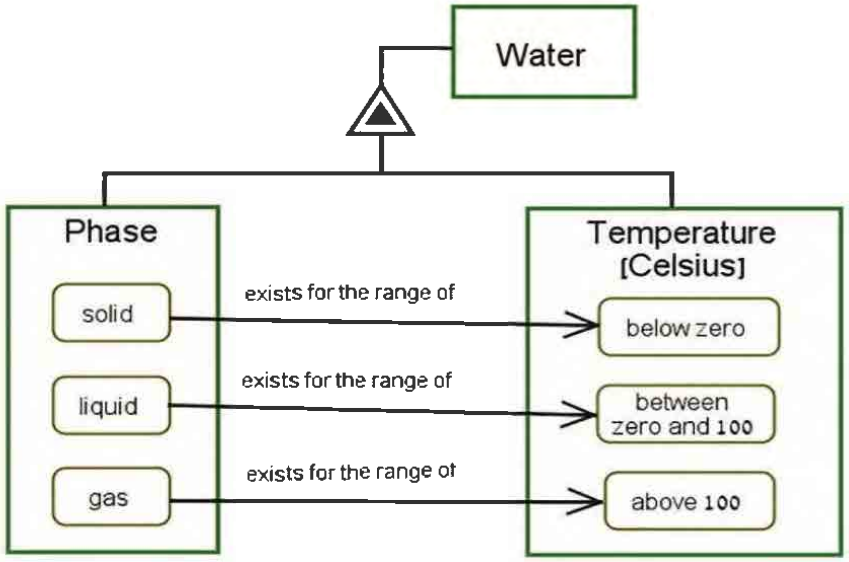
付款人保管人与付款人有关；

收款人保管人与收款人有关；

金融机构保管人与银行有关。（其余的OPL被省略）

图29　通过状态-指定结构关联的具有对象的相关属性值

示例2：图 30 的 OPD 中，水的三个相位值中的每一个值都与其相应的温度取值范围通过三个源-和-目标状态指定标签结构关联的那个 “存在范围” 的标签相关联。



水展示了相位和以摄氏为单位的温度；

水的相位可以是固体、液体或气体；

水的摄氏温度可以是低于零、介于零和 100 之间或高于零；

固体相位存在于低于摄氏零度；

液体相位存在于摄氏零到 100 度之间；

气体相位存在于高于摄氏 100 度。

图30　源-和-目标状态-指定带标签结构关联

10　关系基数

10.1　结构和程序关联中的对象多重性

对象多重性指的是对一个与关联相关的对象操作实例的数量或数目的要求或约束规范，有时也被称为一个参与约束。除非出现一个多重性规范，否则一个关联的每一端应只指定一个对象操作实例。多重性规范也许会在以下几种情况中出现：

a)　为任何一种标签结构关联指定多个源或目标对象操作实例；

b)　指定一个在聚合-分散关联中具有多重操作实例的参与对象，其中不同的参与说明可被连接到整体的某一个部分；　以及

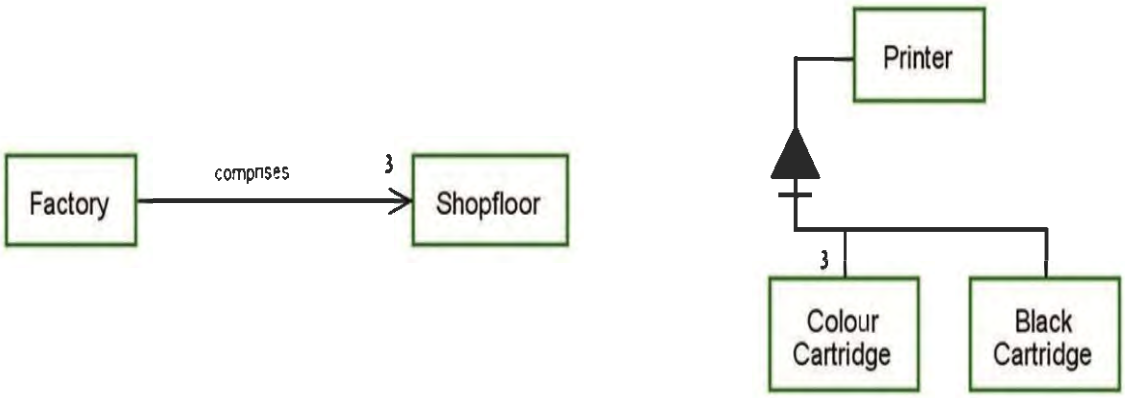
c)　指定一个在程序关系中带有多重操作实例的对象。

对象多重性规范可作为整数或作为模型操作期间解析为整数值的参数符号而出现，并可包含算术表达式。规范可包括一定的取值范围或一组取值范围。

从图形上看，一个整数、一定范围的整数、一个参数符号、一定范围的参数符号或一组整数或参数符号的其中任何一种都可作为注释而在其所适用的关联附近出现，都应表示为对象的多重性。

一个包括具有多重性对象的OPL句子语法应包括先于对象名称的对象多重性，如果基数指定了多个运行实例的话，对象名称则有可能以其复数形式出现。以下例子呈现了OPL句子中许多对象多重性的使用。

示例：图 31 显示了 OPD 左侧的一个置于单向带标签结构关联目标一端的分散约束。OPD 右侧是一个用于聚合-分散关联的两个对象之一的目标（部分）端上的一个分散约束。



工厂包含 3 个车间。　　打印机包含 3 种颜色的墨盒、黑色墨盒和其它部件。

图31　对象多重性例子

对象多重性可以是一个参数或一定范围的参数或一组由逗号分隔的两个或更多数字和/或参数范围。一个范围应被标明为qmin..qmax的闭区间，即包括边界qmin 和qmax。在OPL中，范围符号 “..”的表达式应是 “到 （to）”，以及两个相邻范围分割的逗号表达式应是 “或（or）” 。

对象多重性的规范可作为一个可选参数发生，使用范围符号、星号和问号以以下形式发生：

——“0..1”是指 0 或 1，使用其适用的对象附近的问号（？）注释，在对象之前马上使用一个“一个可选”的 OPL 语法；

——“0 . . *”是指零或多个，使用其适用的对象注释的星号（*），在对象之前马上使用 “可选择” 的 OPL 语法；以及

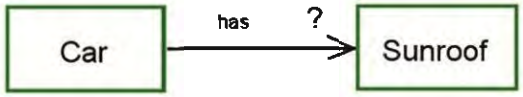
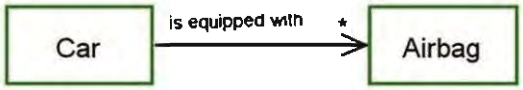
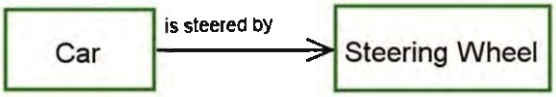
——“1 . . *”是指一个或多个，使用其适用的对象附近的加号（+）注释，在对象之前马上使用“至少一个”的 OPL 语法。

注1：范围符号 “..” 在多重性规范中具有两种用途，一个作为两个边界值之间的分隔符，例如带有“to”的阐释的 Qmin..qmax 和一个作为可选值之间的分隔符，例如带有“或”阐释的“0 . . *”。

注2：在指定基数约束时需要加以小心，这样约束就可适用于指定的对象而非该对象的一个属性。如果对象具有一个计量单位，那么多重性所指的就是该计量的单个单位的计算，例如 32 毫升的水。

表16 总结了关联的可选性。

表16　关联可选性小结

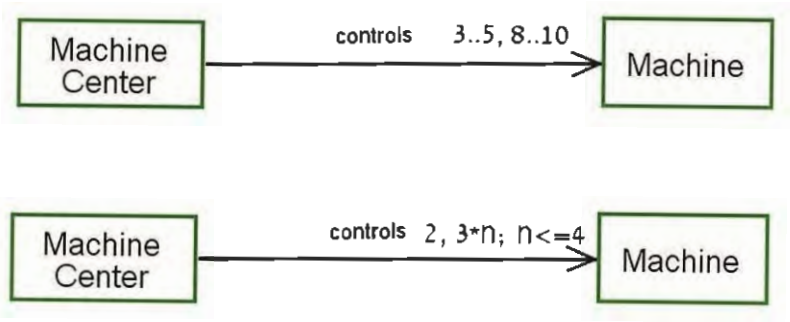
上界和下界 qmin .. qmax	分散约束符号和 OPL 短语	OPD 例子和相应的 OPL 句子
0..1	? 可选	 汽车有一个可选天窗。
0..*	* 可选 (没有到许多)	 汽车拥有可选气囊。
1..1	(无)	 汽车由方向盘操控。
1..*	+ 至少一个	 汽车承载至少一个备用胎。

10.2　对象多重性表达式和约束

对象多重性可包括算术表达式，使用带有其通常语义的运算符号“+”，“－”、“*”、“/”、“（”、和“）” 并应在相应的OPL句子中使用一般的文本对应。

一个整数或一个算术表达式可以约束对象多重性。从图形上看，表达约束应出现在一个分号之后，将它们与它们所约束的表达式隔开，并使用等号/不等号符号“=”、“<”、“>”、“<=” 以及 “>=”、花括号“{”和“ }” 将成员包括进来，并且，处于 “in” 中的成员运算符 ”（。。的要素，∈）全都具有其通常的语义。相应的OPL句子应将以黑体字出现的约束短语放置于约束所适用的以 “约束于” 形式出现的对象之后。

示例1：图 32 提供了具有范围和参数的对象多重性例子。

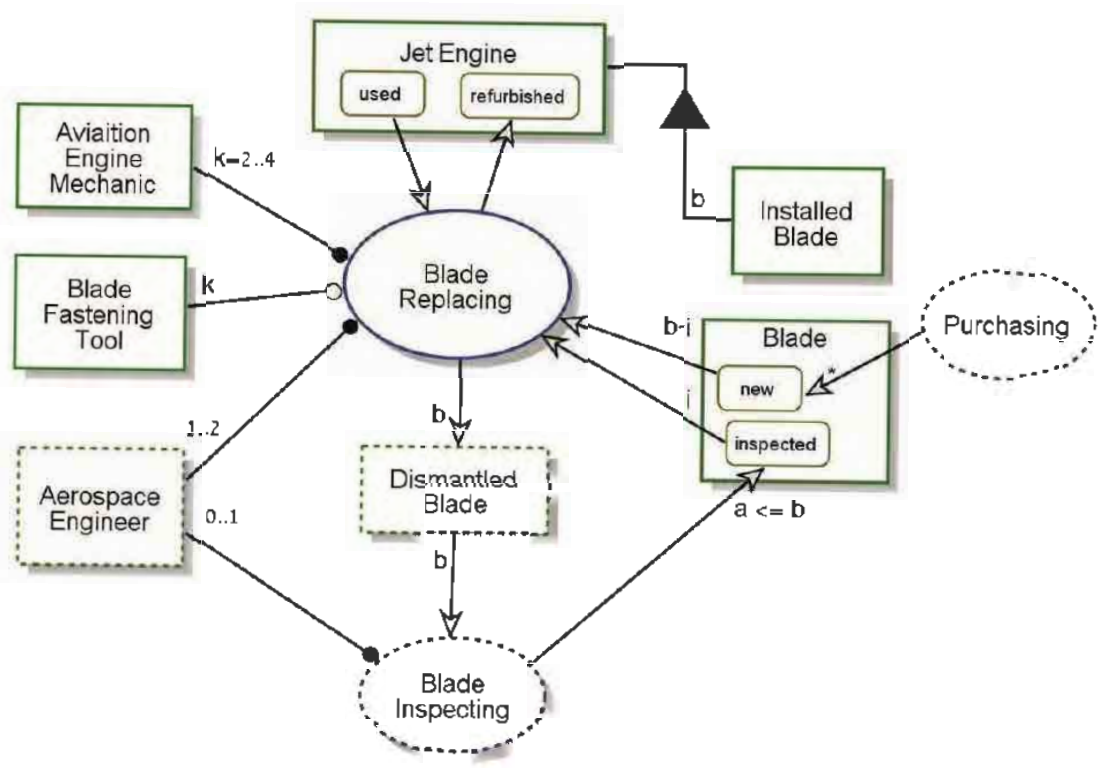


机器中心控制3到5或8到10台机器。

机器中心控制了2 或3*n 台机器，其中n<=4。

图32　具有范围和参数的对象多重性例子

示例2：图 33 模型是一个刀片更换系统，其中一台喷气发动机拥有 b 个安装的刀片。两到四名（到 k 的数字集）航空发动机机械负责处理刀片更换过程，所使用的是 k 个刀片紧固类工具。而且由一或两个航天工程师负责处理刀片更换过程。这个过程产生了 b 个拆卸下来的刀片，其经过刀片检查，产生 a 个被检验过刀片的（多数时候是 b 个)的环境过程。这个过程总共消耗了 b 个刀片，其中 i 个具有经过检验和 b-i 个新的。任何数量的新刀片可以通过购买它们来获得。



k=2 到4 个航空发动机机械负责处理刀片更换；

喷气发动机可以是使用过的或重新装备；

喷气发动机包含b个安装的刀片；

1到2名航天工程师 负责处理刀片更换；

一个可选航天工程师负责处理刀片检验 ；

刀片可以是检验过或新的；

刀片更换 需要k个刀片紧固类工具；

刀片更换 将喷气发动机从旧变为重新装备；

刀片更换消耗 检验过的i个刀片和新的b - i 个刀片；

刀片更换生成了b 个拆卸的刀片；

刀片检验消耗了b 个拆卸的刀片；

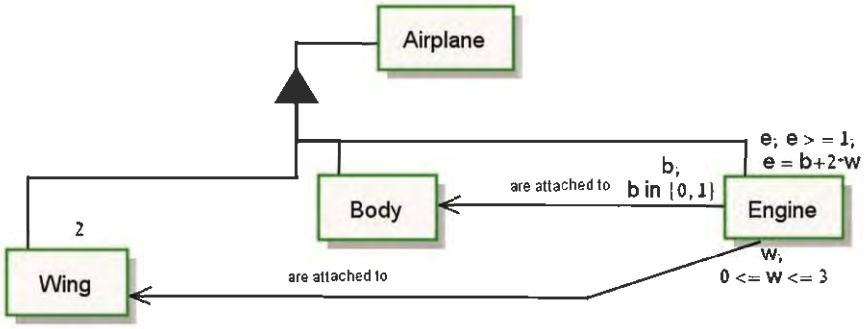
刀片检验 产生了a <= b 个检验过的刀片；

购买生成了许多新的刀片。

图33 对象多重性：算数表达式和约束例子

如果一个对象多重性参数具有一个以上的约束,它们则应在参数之后作为一个由分号分开的约束列表出现。任何约束都可包括模型中出现的任何对象多重性参数。参数名称在整个系统模型应是独一无二的。

示例3：图 34 描述了在一种指定 OPD 和相应 OPL 句子中参数化分散约束的方法。



飞机由机身、2个机翼和E个发动机组成，其中e>=1，E= B+ 2 *w，b个引擎连接到车身，其中b处于 {0，1}中。W个引擎连接到翼，其中0 <= W<= 3。

图34　多重性参数化约束例子

注1：聚合-分散是分散-约束所适应的唯一的基本结构关系。

注2：表达过程多重性并不使用分散约束。相反，表达相同过程的顺序重复性使用了一个带有迭代次数的计数器的反复过程。一个处于放大过程中的并行同步过程或异步过程提供有其它迭代机制。

10.3　属性值和多重性约束

结构关联和程序关联的对象多重性结构表达式指定了解析到整数值的整数值或参数符号。相反，与对象属性或过程相关的值可以是整数值或实数值，或解析到整数或实数值的参数值以及字符串和枚举值。

注1：实际值使用与对象相关的计量单位来适应表达式。

图形上来说，放置在其所属的属性内的一个标签的圆角矩形应表示一个具有对应于标签名称的值或取值范围（整数、实数或字符串字符）的一个属性值。在OPL文本中，属性值应以黑体小写形式出现。

一个具有属性值对象的OPL句子的语法应是：对象的属性是值。

一个具有属性取值范围的对象的OPL句子的语法应为：对象范围的属性是取值范围。

注2：属性取值范围对于对象多重性具有相同的表现力，但可选性除外。

与一个具有一个实数值的属性相连接的结构关联或程序关联可以指定一个不同于一个对象多重性的关系约束。

从图形来说，一个属性值的约束是以一个数字、整数、实数或一个符号参数形式出现的注释，置于关联的属性端附近并与关联对齐。

11　逻辑运算符：AND、XOR 和 OR

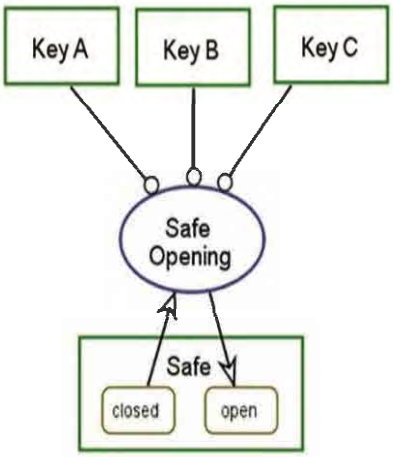
11.1　逻辑 AND 的程序关联

起源于或到达相同过程的一组同类的两个或更多程序关联应具有逻辑AND的语义。

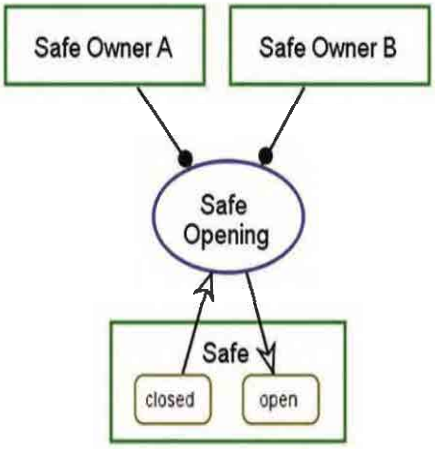
图形上来说，带有AND语义的关联在过程等值线上不会相互接触。

带有AND语义关联的语法应是在每一个关联在OPL句子中使用单个“and”连词的短语而非分开的独立句子。

示例1：图 35（右），保险箱打开的过程既需要保险箱拥有者 A 和保险箱拥有者 B。图 35 中（左侧），打开保险箱需要所有三个键。



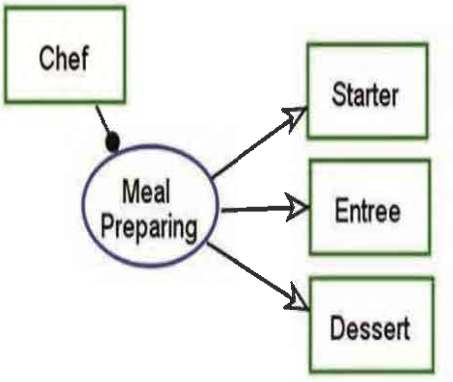
保险箱可以是关或开。
保险箱打开需要 Key A, Key B 和 Key C.
保险箱打开将保险箱从关变为开。



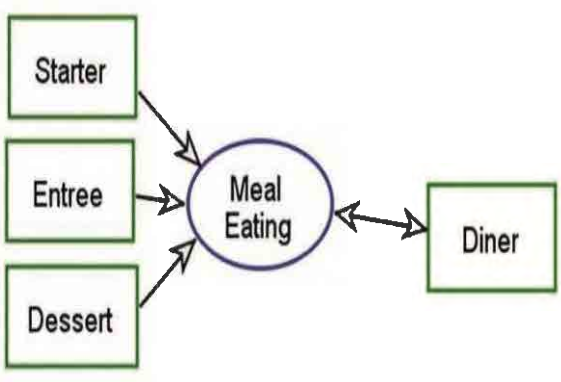
保险箱可以是 closed 或 open.
保险箱拥有者 A 和保险箱拥有者 B 负责处理打开保险箱。
打开保险箱将保险箱从关变为开。

图35　用于代理和仪器关联的逻辑 AND

示例2：图 36 中（左），膳食准备产生了全部三道菜。图 36 中（右），就餐消耗所有三道菜。



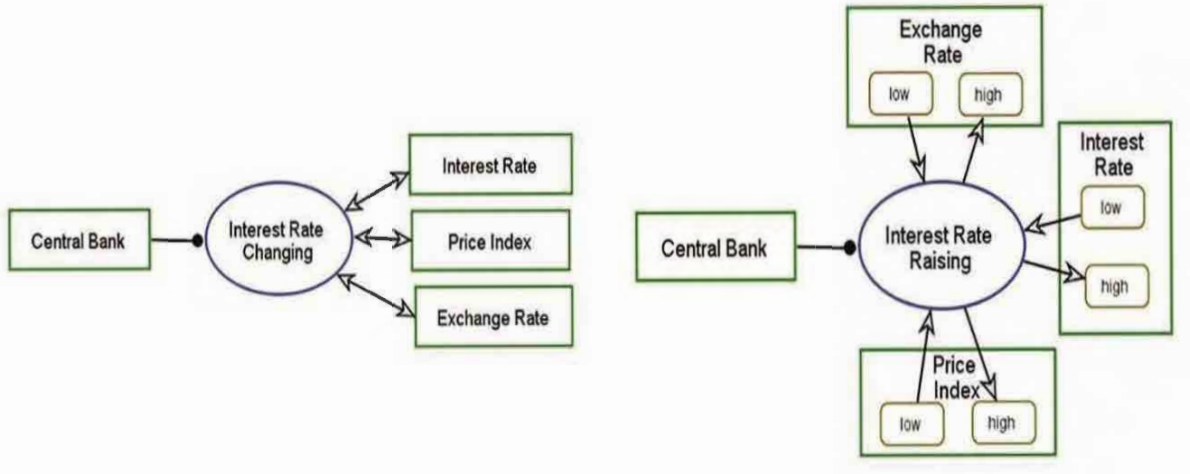
厨师开始用餐准备。
用餐准备产生了开胃菜、正餐和饭后甜点。



就餐涉及到用餐者。
就餐消费饭后甜点、正餐和开胃菜。

图36　用于结果和消耗关联的逻辑 AND

示例3：图 37 左侧的 OPD 中，利率变化影响了汇率，物价指数和利率这三个对象。在右侧的 OPD 中，利息汇率上涨、物价指数和利率的所有三种效果通过三对投入-输出-指定效果关联进行显式说明。



中央银行处理利率变换。

利率变换影响汇率、价格指数

和利率。

中央银行负责处理利率变化。

利率可高可低。

价格指数可高可低。

汇率可高可低。

利率上涨将汇率从低变为高，价格指数从低变为高以及利率从低变为高。

图37　用于效果关联和输入-输出关联对的逻辑 AND

注：参见12条 对AND语法的路径标签所产生的影响。

11.2　逻辑 XOR 和 OR 的程序关联

一组在相同对象或过程上的来源于或到达一个共同点、具有相同类型的两个或多个程序关联应是一个关联扇面。一个关联扇面应遵循XOR或OR运算符的语义。通用于关联的关联扇面端应是趋同型(convergent)关联端。对于关联不通用的关联扇面端应是趋异型(divergent)关联端。

XOR运算符是指在关联扇面的趋同型关联端只有事物的其中之一存在或发生。如果趋异型关联端拥有对象，则仅有一个存在。如果趋异型关联端拥有过程，则仅有一个会发生。

注：OPM中所使用的XOR运算符不同于一些二进制XOR运算符的阐释，其中输出用于输入的一个奇数是1，用于输入的一个偶数是0。

从图形上看，一个跨越关联扇面的连接、焦点位于接触收敛端点的虚线圆弧应表示XOR操作符。

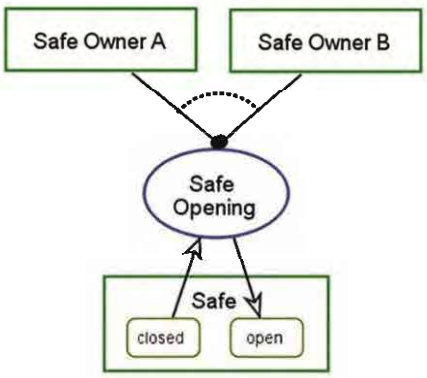
一个带有XOR语义的n个事物的关联扇面的语法应是一个包含以下形式短语的单个OPL句子：事物的事物$_1$、事物$_2$，… 和事物$_n$。。。的至少其中之一。

OR运算符应意指在关联扇面的发散端至少有两个或更多事物存在或发生。如果趋异型关联端具有对象的话，那么至少将会有一个对象存在。如果趋异型端具有过程，则至少一个过程发生。

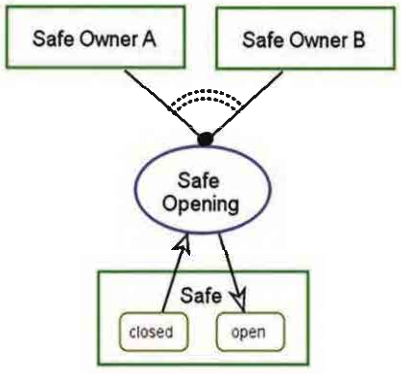
从图形来说，两个横跨关联扇面的连接、焦点处于接触收敛端点的同心虚线圆弧应表示OR操作符。

具有OR语义n个事物的关联扇面的语法应是一个包含一种短语形式的单个OPL句子：事物$_1$、 事物$_2$、…和事物$_n$的至少其中之一。。。

示例：图 38 右侧的 OPD 中，通过使用 XOR，保险箱拥有者 A 和保险箱拥有者 B 中只需一个出现来打开保险箱。在左侧的 OPD 中，通过使用 OR，保险箱拥有者 A 和保险箱拥有者 B 中的至少一个需要出现来打开保险箱。在这两个 OPD 中的关联风扇都是趋同型的，并由两个代理关联组成。



只有保险箱拥有者 A 和保险箱拥有者 B 的其中之一 打开保险箱。



保险箱拥有者 A 和保险箱拥有者 B 至少其中之一打开保险箱。

图38 代理关联的逻辑 OR（左）和逻辑 XOR（右）例子

11.3 趋异型和趋同型 XOR 和 OR 关联

表17显示了当来源事物是对象且目标事物是一个过程时，消费关联扇面是趋同型的，而当来源事物是过程且目标事物是一个对象时，结果关联扇面是趋异型的。

表17 XOR 和 OR 趋同型消耗和结果关联小结

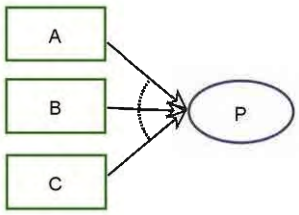
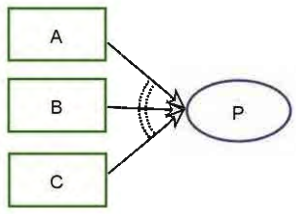
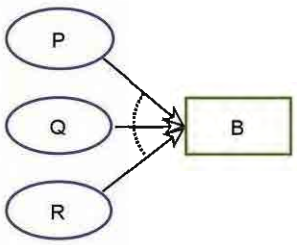
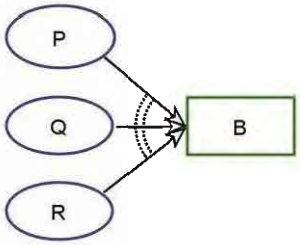
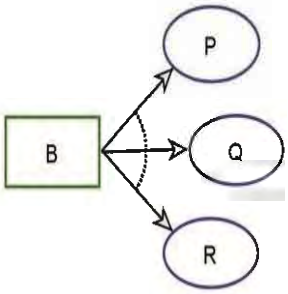
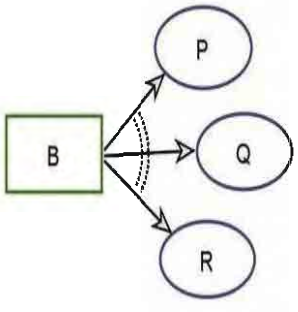
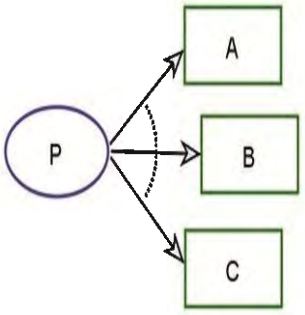
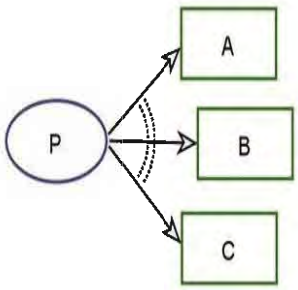
	异或 (XOR)	或 (OR)
趋同型消耗关联扇面	 P 只消耗 A、B 或 C 的其中一个。	 P 消耗 A、B 或 C 的至少其中一个。
趋同型结果关联扇面	 P、Q 或 R 中只有一个产生 B.	 P、Q 或 R 中的至少其中一个产生 B.

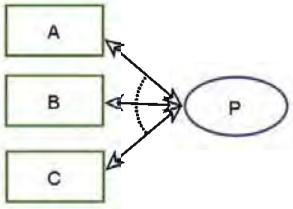
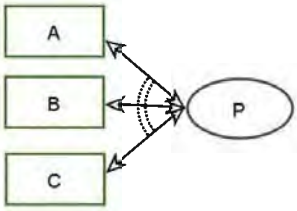
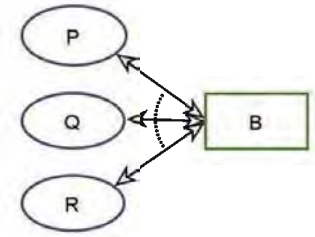
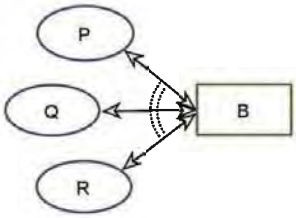
表18 显示当来源事物是一个对象且目标事物是过程时，消耗关联扇面应是趋异型的，而当来源事物是一个过程且目标事物是对象时，结果关联扇面应是趋异型的。

表18 XOR 和 OR 趋异型消耗和结果关联扇面小结

	异或（XOR）	或 (OR)
趋异型消耗关联扇面	 P、Q 或 R 的其中之一 消耗 B.	 P、Q 或 R 至少其中一个消耗 B.
趋异型结果关联扇面	 P 只产生 A、B 或 C 的其中之一。	 P 产生 A、B 或 C 中的至少一个。

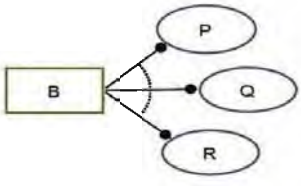
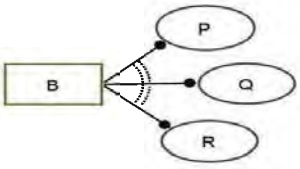
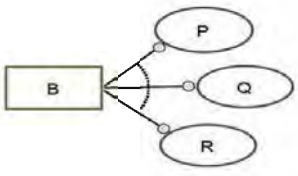
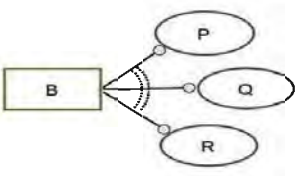
由于一个效果关联是双向的，所以由一个效果关联扇面所连结的事物即是来源也同时是目标，使得趋同型和趋异型关联扇面的定义变得无效。相反，如表19所示，一个关联扇面所连接的有关多个对象或多个过程将会产生区别。

表19　XOR 和 OR 效果关联扇面总结

	异或（XOR）	或（OR）
多重对象效果关联扇面	 P 只影响 A、B 或 C 的其中之一。	 P 影响 A、B 或 C 中的至少一个。
多重过程效果关联扇面	 P、Q 或 R 中的至少一个影响 B。	 P、Q 或 R 中的至少一个影响 B.

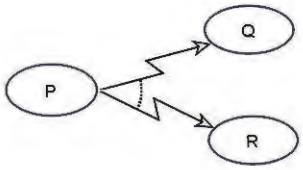
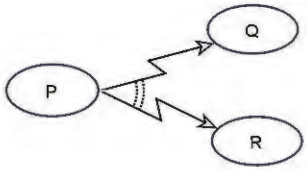
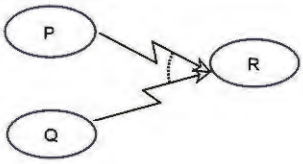
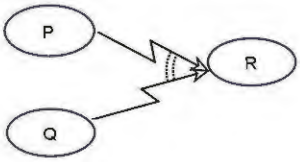
由于一个使能器是一个对象，如表 20 所示，代理关联扇面和仪器关联扇面两者都应是趋异型的，具有多个作为目标的过程。

表20　代理人和仪器关联扇面小结

	异或（XOR）	或（OR）
代理关联扇面	 B 只处理 P、Q 或 R 的其中之一。	 B 处理 P、Q 或 R 中的至少一个。
仪器关联扇面	 P、Q 或 R 中只有一个需要 B。	 P、Q 或 R 中至少一个需要 B。

用于XOR和OR的调用关联扇面都可以是趋异型或趋同型的， 如表21所示。

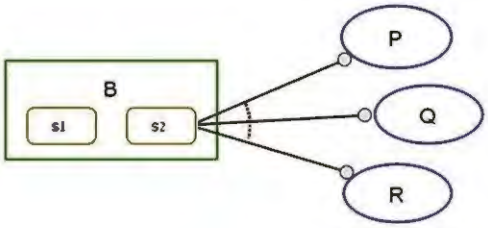
表21　调用关联扇面小结

	异或（XOR）	或（OR）
趋异型调用关联扇面	 P 只调用 Q 或 R 中的一个。	 P 调用 Q 或 R 中的至少一个。
趋同型调用关联扇面	 P 或 Q 中只有一个调用 R。	 P 或 Q 中至少一个调用 R.

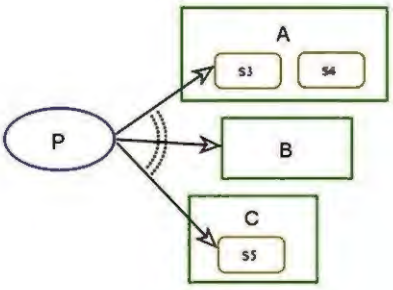
11.4　状态-指定的 XOR 和 OR 关联扇面

11.3 中的每一个关联扇面都应具有一个相应的状态-指定版本，其中源和目标可以是一个无状态规范的指定对象状态或对象。将状态-指定和无状态关联组合成一个关联的源和目标是有可能发生的。

示例：如图 39 显示了左侧的一个 XOR 状态-指定仪器关联扇面和右侧的一个 OR 混合的结果关联扇面，其中关联是为对象 A 和 C 而非 B 来进行状态-指定的。



P、Q 或 R 中只有其中之一需要 s2 B.



P 产生了 s3 A、B 或 s5 C 中的至少一个。

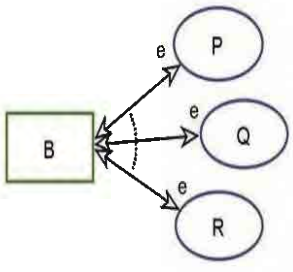
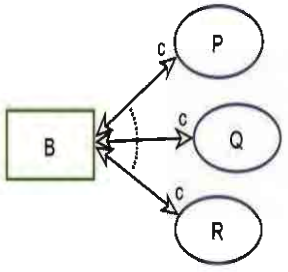
图39　状态-指定 XOR 和 OR 关联例子

11.5　控制-修正的关联扇面

每一个用于消耗、结果、效果和使能关联以及其状态-指定版本的XOR关联扇面都应具有一个相应的控制-修正关联扇面：一个事件关联扇面和一个条件关联扇面。

表22 呈现了事件和条件效果关联扇面，以用来代表修正了的关联扇面的基本（非状态-指定）关联版本。

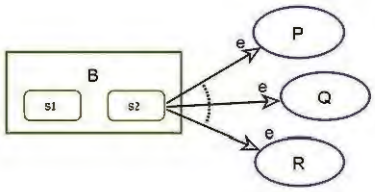
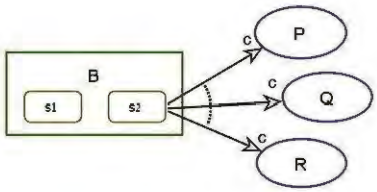
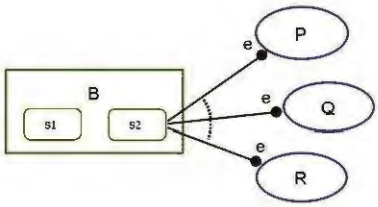
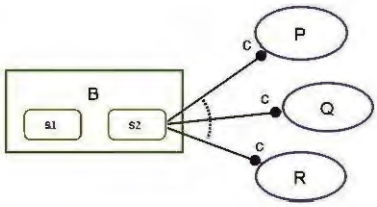
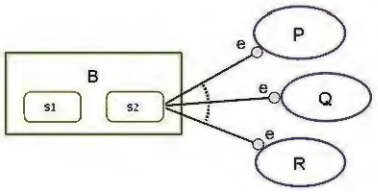
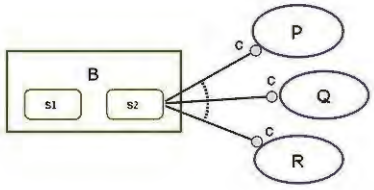
表22　事件和条件效果关联扇面

事件	条件
 B 只启动了 P、Q 或 R 的其中之一，在此情况下所发生的过程影响到 B。	 如果 B 存在，则只有 P、Q 或 R 中的一个发生，在此情况下，发生的过程影响 B，否则这些过程将被跳过去。

11.6　状态-指定的控制-修正关联扇面

每一个控制-修正关联扇面除了控制-修订效果关联扇面外应具有一个相应的状态-指定控制-修正关联扇面。由于这些状态-指定版本比其非状态-指定版本更加复杂，表23给出了状态-指定版本的OPD和OPL以及以下用于每一个状态-指定版本的相应无状态版本。

表23　状态-指定和无状态控制-修正关联扇面

	事件控制修饰符	条件控制修饰符
消耗关联扇面	 S2 B 仅启动 P，Q 或 R 的其中之一，其消耗 B。 无状态情况： B 仅启动 P、Q 或 R 的其中之一，其消耗 B.	 如果 B 是 s2，仅 P、Q 或 R 的其中一个发生。在此情况下，发生的过程消耗 B，否则这些过程将被跳过。 无状态情况： 如果 B 存在的话，仅 P、Q 或 R 的其中一个发生，在此情况下，发生的过程消耗 B，否则这些过程将被跳过。
代理关联扇面	 S2 B 只启动和处理 P、Q 或 R 中的一个。 无状态情况： B 只启动和处理 P、Q 或 R 中的一个。	 如果 B 是 s2，B 只处理 P，Q，or R 的其中一个，否则这些过程将被跳过去。 无状态情况： 如果 B 存在的话，B 只处理 P、Q 或 R 的其中一个，否则这些过程将被跳过去。
仪器关联扇面	 S2 B 只启动 P、Q 或 R 中的一个，需要有 s2 B. 无状态情况： B 只启动 P、Q 或 R 中的一个，需要有 B。	 P、Q 或 R 中只有一个要求 B 为 s2，否则这些过程将被跳过去。 无状态情况： P、Q 或 R 中只有一个要求 B 存在，否则这些过程将被跳过去。

在表22和表23中的每个XOR关联扇面都应拥有其带有一个相应OPL句子OR的对应物（由一个两条虚线弧形指定），其中所保留的短语 “至少” 来取代 “仅有”。

11.7 关联概率和概率关联扇面

一个拥有结果关联的过程P生成了一个具有s_1 到s_n n个状态的有状态对象B。未指定一个特定状态应表示在任何一个特定状态所生成的概率B应是1/n。在此情况下，到达对象的单个结果关联应取代到达其每个状态的结果关联扇面。

示例1：图 40 左侧的 OPD 中，结果关联从 P 到 B 有三种状态，意味着 P 将创建具有相同概率的 B，Pr =1/3， 用于每个状态期间的创建。图 40 右侧的 OPD 显示了表示相同状况的更为繁琐的方式。

A 可以是 s1、s2 或 s3。B 可以是 s1、s2 或 s3。

P 生成 B。

P 只生成 s1 B、s2 B 或 s3 B 的其中一个。

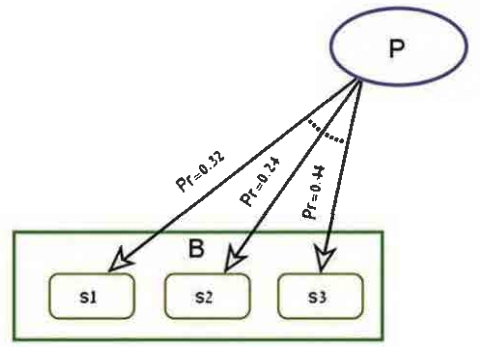
图40 结果关联与一组 XOR 状态 - 指定结果关联之间的等价性

通常，关联扇面中的依附于一个特定关联的那些概率是不相等的。关联概率可以是一个分派给一个处于XOR趋异型关联扇面中的关联属性值，其指定了关联扇面中可能性关联内该特定关联之后的概率。一个概率关联扇面应是一个在每一个用于其概率属性的扇面关联上具有注释的关联扇面，其中概率之和应恰好是1。

从图形上来说，沿着每一个具有一个概率属性的关联属性，一个注释应以Pr=p的形式出现，其中p是关联概率数值或一个参数，表示选择和遵循该扇面的特定关联的系统执行控制的概率。

相应的OPL句话应是没有关联概率的XOR趋异型关联扇面句子，省略了短语 “只有一个...”并在每一个拥有一个概率注释“Pr=p” 的分散事物名称之后插入一个短语 “具有概率P.....”。

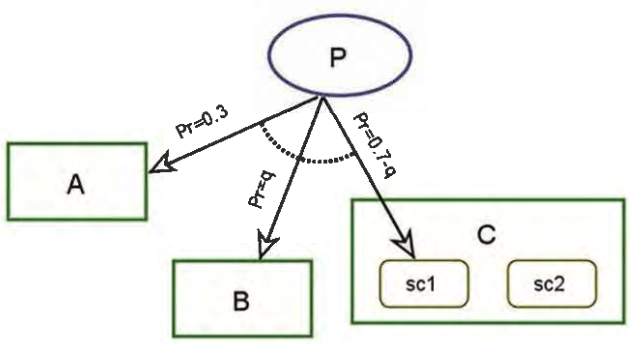
示例2：图 41 显示了两个概率状态-指定对象创建的实例及其决定性的类比。在左侧的 OPD 中，过程 P 能够在 s1, s2, 或 s3 这三种可能状态中创建对象 B，具有结果连接扇面的每一个结果关联所显示的相应概率 0.32、0.24，和 0.44。在右侧的 OPD 中，P 可以创建处于状态 sc1 的 C 或对象 A、B 中的一个，与结果关联扇面的每一个结果关联一同显示了概率。



P 生成具有概率 0.32 的 s1 B，具有概率 0.24 的 s2 B，或具有概率 0.44 的 s3 B.

类似确定性案例：

P 仅生成 s1 B、s2 B 或 s3 B 的其中之一。



P 生成具有概率 0.3 的 A 、具有概率 q 的 B 或具有概 率 0.7-q 的 sc1 C。

类似确定性案例：

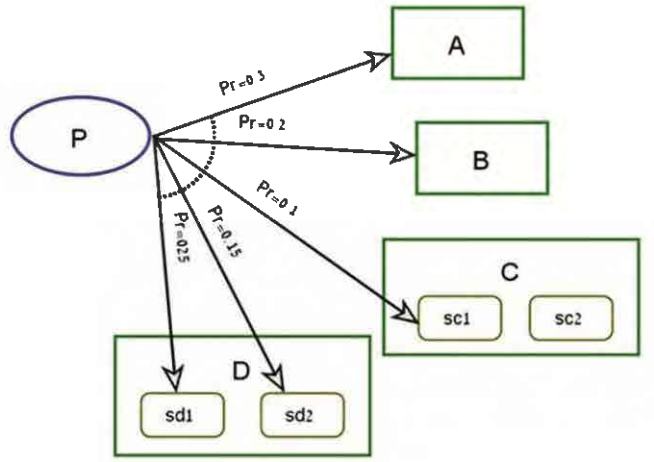
P 仅生成 A、B 或 sc1 C 的其中之一。

图41 概率状态-指定对象创建例子

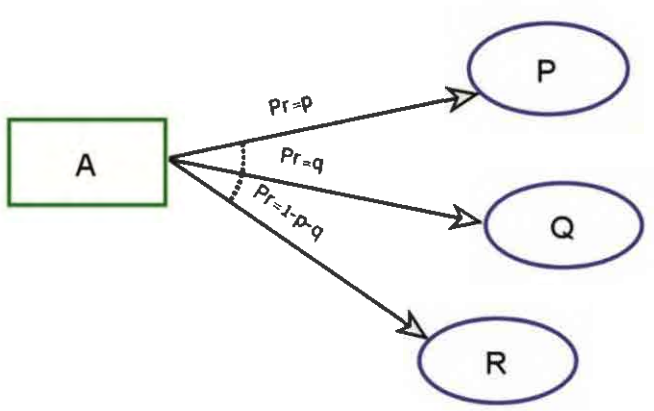
对于一个能够生成带有s_1 到 s_n有状态对象B并拥有初始状态si的具有一个结果关联的过程P来说，P应创建一个处于s_i 状态具有概率1.0的B。然而，假如B拥有m，带有m < n的初始状态，则P应创建处在初始状态其中之一的具有概率 1/m的B。

对于一个概率结果关联扇面来说，任何一个结果物都可以是一个有特定状态或无特定状态的对象。对于所有包含了其它程序关联类（包括那些带有事件和条件控制修饰符）、其中关联扇面中的关联目标是过程的关联扇面来说，源都可以是一个对象或一个对象的指定状态。

示例3：图 42 顶部的 OPD 显示了一个概率结果关联扇面，其中 P 使用指定的概率生成处于 sc1 或 sd2 状态的对象 C 或 D，或 A 或 B 中一个对象。在图 42 的中间的 OPD 显示了一个概率消耗关联扇面，其中 A 被 P 或 Q 或 R 这三个过程的其中之一过程以指定概率被消耗。下面的 OPD 也以额外的事实同样表达了 A 需要处于 s2 这种状态。



P生成了一个具有概率为0.3的A、概率为0.2的B，概率为0.1 的 sc1 C，概率为0.25 的sd1 D或概率为0.15的sd2 D。



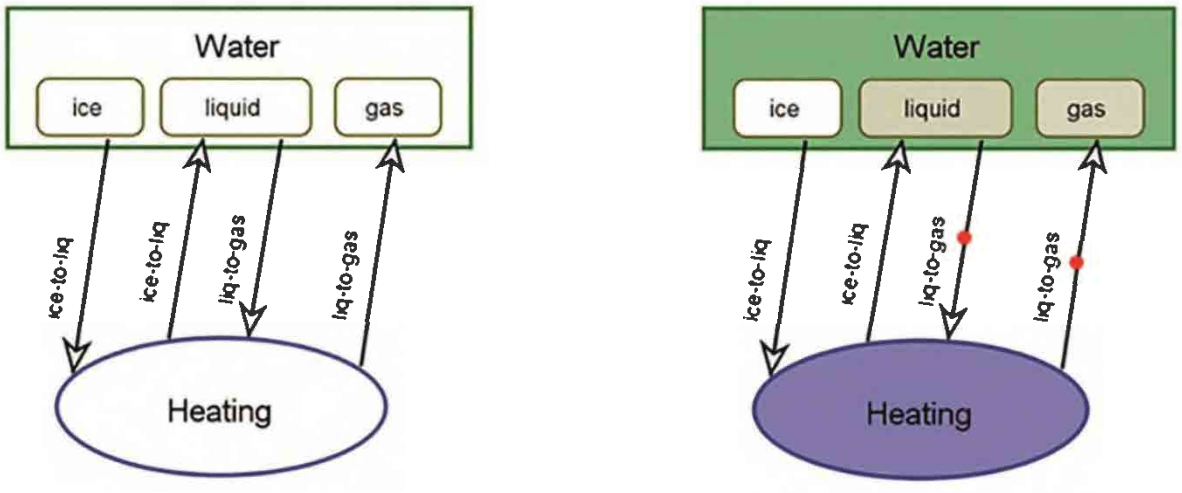
具有概率p 的P、具有概率 q的Q 或具有概率1-p-q 的R 消耗A。

图42　　作为一个概率关联扇面的源和目标的有指定状态和无指定状态的对象

12　执行路径和路径标签

一个路径标签应是平行于一对程序关联的一个关联属性和相应注释。当过程前置条件涉及到一个具有路径标签关联连接的对象以及后过程对象集拥有用于目标对象的多种可能性时，适当的后过程对象集目标应是那个通过使用与前过程对象集使用的相同路径标签而获得的目标。

示例1：在图 43 中，有两个输出关联：一个是从加热到水的液体状态和其它到气体的状态。当从冰状态进入到加热状态时，并不清楚结果状态是否是液体还是气体。路径标签顺着程序关联通过独特地确定过程出口上的适当关联来化解这种困境，如左侧的动画模拟所示。



水可以是冰、液体或气体。

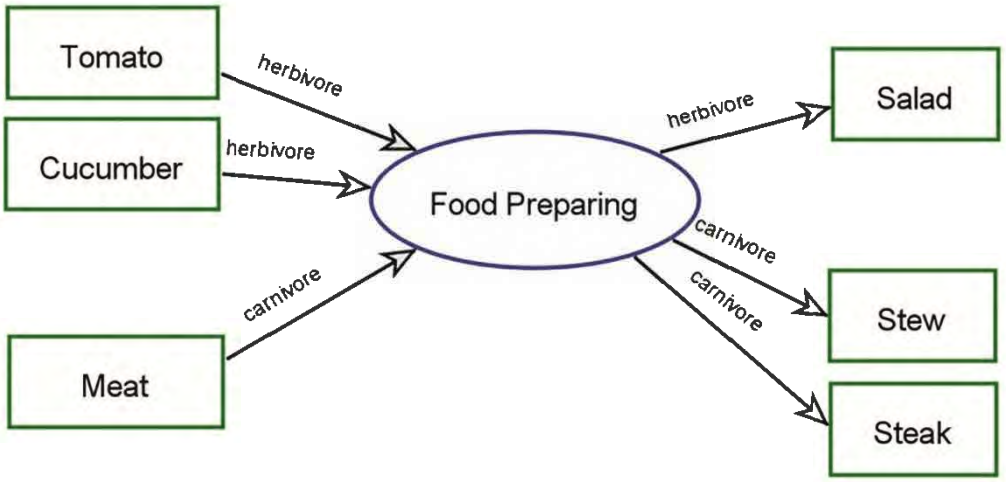
路径冰到液体之后，加热将水从冰变为液体。

路径液体到气体之后，加热将水从液体变为气体。

图43　执行路径和路径标签

注：一个路径标签是一个程序关联上的标签。其通过明确指出要跟踪的关联是一个具有与启动过程相同的标签来消除由于多个输出程序关联而导致的模糊状态。

示例2：图 44 展示了消耗和结果关联上使用路径标签，之后紧跟 OPL 段落。



路径肉食者之后，食品准备消耗肉。

路径食草动物之后，食品准备消耗黄瓜和西红柿。

路径 肉食者之后，食品准备生成炖菜和牛排。

路径食草动物之后，食品准备生成沙拉。

图44　消耗和结果关联上展示的路径标签

13 使用 OPM 管理上下文

13.1 完成系统图（SD）

用于边界、利益相关者、先决条件及后置条件的系统目的、范围和功能的定义应是确定其它的要素，包含环境事物是否应出现于模型中的基础。

系统图（SD）是一个OPD，对下列进行建模：

——利益相关者，特别是受益者；

——一个传达受益人所期望得到的功能值的过程；以及

——其它需要创建一个简洁的相应 OPL 段落的环境和系统事物。

相应的OPL段落应为系统运行提供情境背景。

函数值的表达可以是：

——显式的，通过识别受益人的源输入和目标输出状态或其属性的一种或多种初始值和最终值；或

——隐式的，通过表明受益人将会受到系统功能影响。

系统图（SD）应只包含核心和重要的事物 - 这些事物对于理解系统的功能和上下文是必不可少的。建模者应使用 OPM 的细化机制以逐步求精地揭示有关系统图（SD）内容的事物细节。

示例：在一个制造设施中，受益人开发并部署了一个预防性维护系统。系统功能预防性维护执行将制造设施的停机属性从 “高” 变成了 “低” 。这种变化为制造设施增加了功能值，因为它具有更多的运行时间来制造产品，并以开发和操作预防性维护系统的投资为代价而增加销售和收入。

13.2 实现模型内涵

13.2.1 OPM 细化-抽象机制

OPM应提供抽象和细化机制来管理对于模型清晰度和完整性的表示。这些机制使得置于上下文段中的说明有可能作为独立却互相连接的若干OPD，这两者放在一起则将组成一个具有一定功能价值的系统模型。这些机制应能在共同对象、过程和关系间相互关联的不同背景下展示和观看所建模系统及其所包含的要素。这套明确且兼容互联的OPD应全部将整个系统指定到一个适当范围的细节中去并为该系统的综合表示提供一个对象过程语言（OPL）模型的相应文本说明。

OPM细化-抽象机制应为以下三对：状态表达与抑制、展开与折叠以及放大与缩小。

13.2.1.1 状态表达和状态抑制

明确地描绘OPD中的一个对象状态可能会导致一个图中的过于拥挤或眼花缭乱的情景，使其变得难以阅读或难于理解。

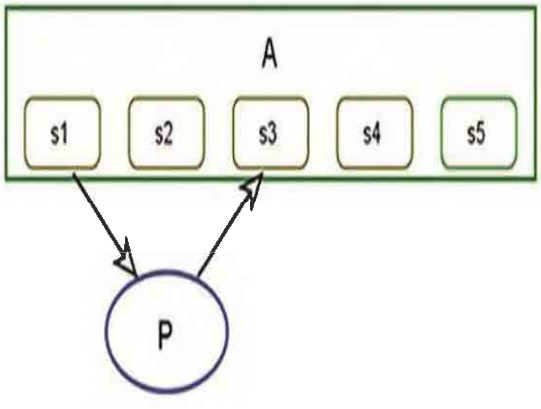
OPM应提供一种状态抑制选项，当该OPD情景中那些状态变得没有必要时，它可抑制一个处于特定OPD中所代表对象的一些状态或所有状态的出现。

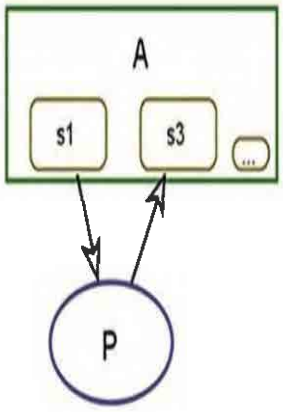
状态抑制的反义应是状态表达，其揭示有关可能对象状态的信息。对应于一个OPD的OPL应仅像OPD描绘的那样来表示对象状态。

建模者可以在OPM中抑制任何状态子集。然而，用于一个对象的一整套对象状态应是那个出现在整个OPM模型的所有OPD中的相同对象的状态结合。

从图形来说，表示一个对象展示了其状态的一个适当子集（即至少一个但不是全部）的注释应是对象右下角中的一个小的状态抑制符号。该符号以作为一个带有一个椭圆形的小状态而出现，意味着视图存在正在进行抑制的一种多种状态的。状态抑制符号的文本相同性应是所保留的短语 “或其它状态”。

示例：图 45 显示了一个表达全部状态的有状态对象及一个抑制版本。





A可以是s1、s2、s3、s4，或s5。

s1变成s3。

A 可以是s1、s3或其它状态。P 将A 从s1变成s3.P 将A 从P 将A 从s1 变为s3。

图45　一个具有表全部状态（左）和一个部分抑制版本(右）的有状态对象

13.2.1.2　展开与折叠

展开应是一个用于细化、精化或分解的机制。展开应展示一组事物，可细化物，其与展开事物，即可细化物是相关联的。展开的结果应是一个层次树，其根部应是展开的事物。与根部相连事物应构成对展开事物的详尽说明。

相反，折叠应是一个用于抽象或构成的机制，其适用于一个展开层次树。折叠应隐藏展开事物集，只留下根部。

四个基本结构关系关联的每一个关联都可应用于展开和折叠。四种展开-折叠对应是：

——聚合展开：揭示一个整体的部分，分散折叠：隐藏一个整体的部分；

——展示展开：揭示展示物的特征，表征折叠：隐藏展示物的特征；

——泛化展开：解释一个通用类的特化，特化折叠：隐藏一个通用类的特化；以及

——类化展开：揭示类实例，实例化折叠：隐藏类实例。

图内展开应在可细化物和其细化物以展开形式出现在相同OPD中时发生。由于展开使用了基本结构关联，所以图内展开在图形、语法和语义上等同于使用基本结构关联。

图内展开应在可细化物和其细化物以展开形式出现在一个新的OPD中时发生。

从图形来说，对于一个以折叠形式出现而没有细化物的可细化物所处的抽象OPD中，以及在一个以展开形式出现并连接到其带有一个或更多基本结构关联的可细化物所处的更详细的新OPD情境中，可细化物都应具有一个粗略的轮廓 。

用于新图OPD、其中可细化物具有n个细化物的相应OPL句子应是：可细化物展开至细化物$_1$、细化物$_2$，…以及细化物$_n$。

注1：可将展开更精确指定为部分-展开、特征-展开、特化-展开以及实例-展开（见 A.4.7.2）。

建模者对于是否使用图内或新图展开的决定应在添加将造成目前 OPD 的混杂状况与为展示细化物及其相关的关联而建一个新的对象过程图(OPD)这两种需求之间找到一种折中方案。

注2：展开常常作为一种新图和图内展开的组合而发生，以表示多个阐述或分解情况。

注3：局部展开可以与描述一个局部基本结构关系关联相同的方式进行描述。

为满足一个OPD的特定情景关联，一个建模者可以选择哪个细化物以展开形式出现。符合OPD的OPL应在OPM双模态表示法之后仅表达那些出现在该OPD中的细化物。

注4：局部折叠等同于部分展开，所展示和隐藏的细化物集的集合体在此是相辅相成的。

注5：展开揭示了更细化的结构细节，而非行为，即不发生执行控制的转移，请参见 13.2.2。然而，涉及程序关联的层次依赖性可导致与展开事物的使用相关的行为变化。

13.2.1.3 放大与缩小

放大应是一种将聚合-分散和展示-表征与附加语义相结合的展开类型。对于过程来说，放大可为子过程、其时间顺序、它们与对象的交互以及将执行控制传递至并从该关联传递回来的情景进行建模。对于对象来说，放大创建了一种独特的可为各组成对象的空间或逻辑顺序来建模的环境。

从图形来看，对于图内和新图过程放大这两者来说，可细化物的椭圆增大以便容纳用于细化物及其处于放大情景中的关联的符号。在新图放大的情况下，可细化物应在一个没有细化物的可细化物所出现的更抽象OPD中，也可在一个可细化物围绕子过程细化物及随之而来的对象所处的具详细的新OPD情景中拥有一个粗略的轮廓。

相应过程放大的OPL句子应为：过程放大至子流程A、子流程B和子进程C，以此排序。

注1：放大可通过显示抽象的 OPD 名称和更详细的 OPD 名称（见 A.4.7.4）被更精确地指定。

一个放大过程的上下文应包括子过程，就是被放大过程的部分和也许是放大过程属性的内部对象。放大过程的关联情景应是可细化物、其子过程、属性和如OPD中所描绘的关联 。

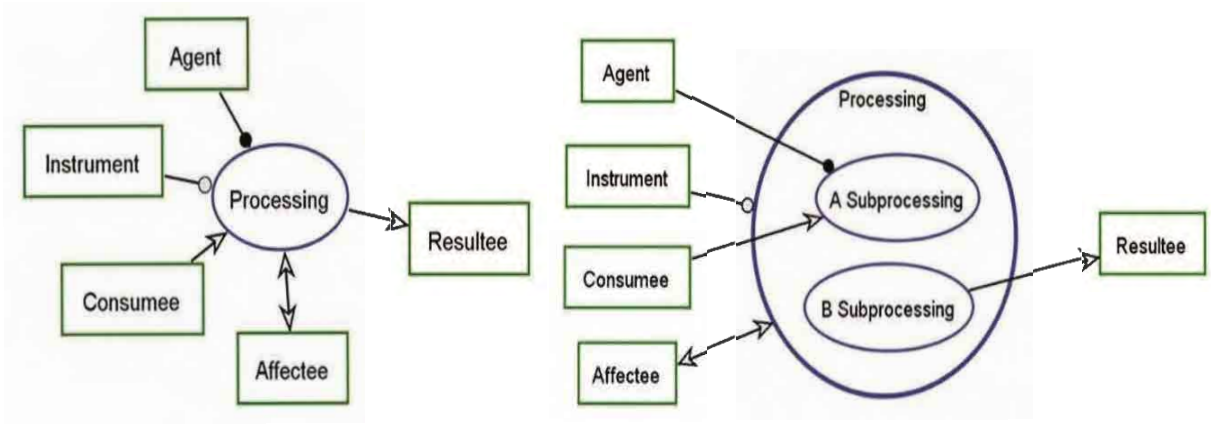
处于一个放大过程上下文内的执行时间轴应从其放大过程的椭圆符号顶端流动到该椭圆的底部。这个时间轴应描绘子过程调用的顺序。处于外部过程椭圆内的子过程椭圆形符号最高点的垂直排列应显示过程上下文内子过程的名义上的执行顺序。

与过程放大相类似，对象放大应将展示构成对象作为放大对象部分，并展示那些在放大对象情景范围内是被放大对象运行的可能性临时过程。与放大一个过程不同的是，放大一个对象不会导致一个执行控制的转移。新图对象放大的后果是一种情景转移，从作为更大OPD上下文部分的对象转换到作为整个OPD情景的对象，其中对象的组成部分被显示并在空间或逻辑上被排序。

从图形来说，放大对象的矩形扩大以容纳细化物的符号以及它们其中的关联。处于放大对象情景内的对象矩形排列应展示对象的空间排列或逻辑顺序。这使得数据，如一个向量或矩阵进行有序枚举。

相应的对象放大的OPL句子应是：对象放大至子对象A、子对象B和子对象C，以此排序。

示例1：图 46 描绘了 SD 中的抽象过程处理、系统图和放大到过程处理之后 SD1 中的过程处理细节、显示了它的两个子过程。



SD

代理 处理 P过程。

过程 需要有仪器。

过程消费被消耗物。

过程影响 受影响物。

过程产生结果物。

SD1

过程需要有仪器。

过程影响受影响物。

过程放大从 A 子过程到B 子过程，以此排序。

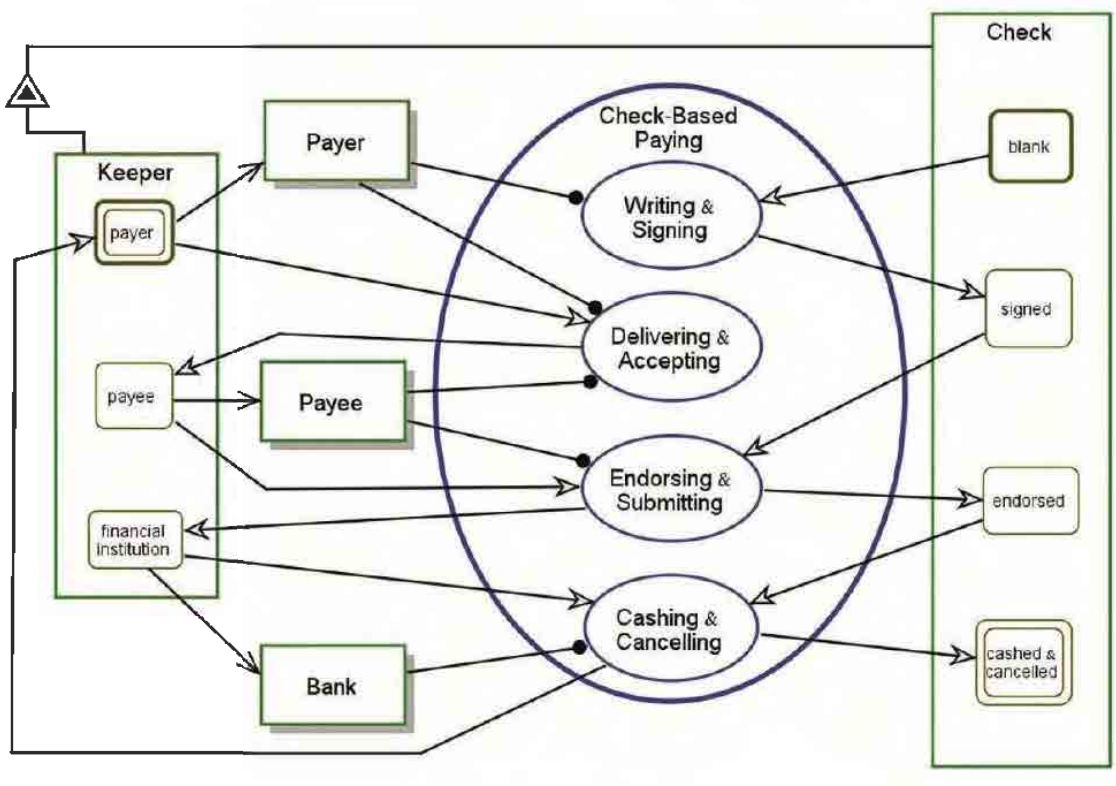
代理处理 A 子过程。

子过程消费被消耗物。

子过程产生结果物。

图46 新图放大类属例子

示例2：图 47 描绘了图 29 的基于支票的支付过程，进行放大以揭示子过程的顺序和从过程到其子过程的关联分派。



支票展示了保管人。

支票可为空白、签字、背书或现金兑现和取消。

支票的空白状态是初始状态。

支票的兑现和取消状态是最终状态。

保管人可以是付款人、收款人或金融机构。

保管人的付款人状态是初始和最终状态。

付款人保管人与付款人有关。

付款人保管人与收款人有关。

金融机构保管人与银行有关。

基于支票支付放大至写和签字、交付和接收、背书和提交以及兑现和取消这种顺序。

付款人进行写和签字以及交付和接收。

收款人进行交付和接收以及背书和提交。

银行处理兑现和取消。

写和签字将支票从空白变为签字。

交付和接收将保管人从付款人变为收款人。

背书和提交将支票从签字变为背书并将保管人从收款人变为金融机构。

兑现和取消将支票从背书变为兑现和取消并将保管人从金融机构变为付款人。

图47　放大展示其四个连续子过程的基于支票的付款过程

注2：放大表达了结构关联和程序关联结果的过程行为，显示了 OPD 中的一个执行控制的动态转移。操作执行环境从过程转移到放大的 OPD，然后再回到过程去。

13.2.2　放大过程情景内的控制（操作）语义

13.2.2.1 隐式调用关联

放大一个过程应指定一个执行控制到不同细节范围的子过程的转移。执行一个具有放大背景的过程应将执行控制递归到过程环境内最顶端的子过程（若干）中去，其在新图放大的情况下处于一个不同的OPD中。执行控制应在其最终启用的子过程完成后返回到放大过程中去。

隐式调用关联应是一个过程与一个放大子过程之间、一个放大过程范围内的两个子过程之间或一个放大子过程与其父过程之间的一个调用关联。与其显式对应物相类似，隐式调用关联应表示一个随后过程或同时开始过程的调用。

一旦到达一个放大过程情景，执行控制应立即转移到处于该过程放大情景中具有最高椭圆（椭圆形）顶点的子过程中去。从一个过程到一个最顶点的放大子过程的隐式调用关联将执行控制进行转移。沿着过程时间轴，一个源的子过程结束将会使用隐式调用关联立即调用随后的子过程（若干）。一旦具有椭圆最顶点，也是该放大情景内最低点的子过程结束，执行控制就应沿着隐式调用关联返回到放大过程中去。

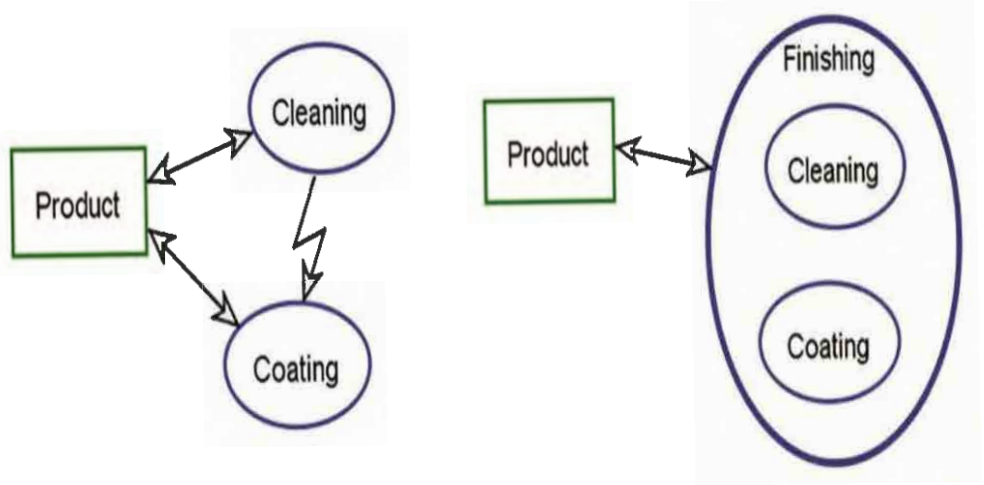
由于调用是一个事件，所以有必要满足每一个子过程的前置条件以允许那个子过程操作。

当两个或两个以上的子过程拥有同一高度的最顶层椭圆点时，隐式调用关联则应启动每一个过程，并应在单独前置条件得到满足时并行开始。最后完成的那个过程应启动下一个过程或下一组并行的子进程。

从图形上看并没有符号明确地表示隐式调用关联。在放大过程情景中子过程的椭圆符号最顶点的这种自上而下的垂直排列应表示一个该排列的若干连续子过程之间的隐式调用关联。

一个隐式调用关联OPL句子的语法应为：过程放大至子过程A到子过程B ，以此排序。

示例：在图 48 左侧的 OPD 中，清洗调用涂层，因此清洗首先影响到产品，然后涂层影响到产品。调用关联指挥该过程顺序。在图 48 右侧的相同 OPD 中，结束放大到清洗和涂层，将先前的椭圆顶点置于后者之上，所以当结束开始时，执行控制立即转移到清洗，而当清洗结束时，隐式调用关联则调用涂层。这两个 OPD 在语义上相等，所不同的是左侧的那个没有将结束作为一个封闭的情景，这样从系统的角度来看，其表现力由于使用了更多的图形要素而变得更弱一些。



清洗影响产品。

清洗调用涂层。

涂层影响产品。

结束影响产品。

结束放大至清洗和涂层这种顺序。

图48 调用关联（左侧）和隐式调用关联（右侧）

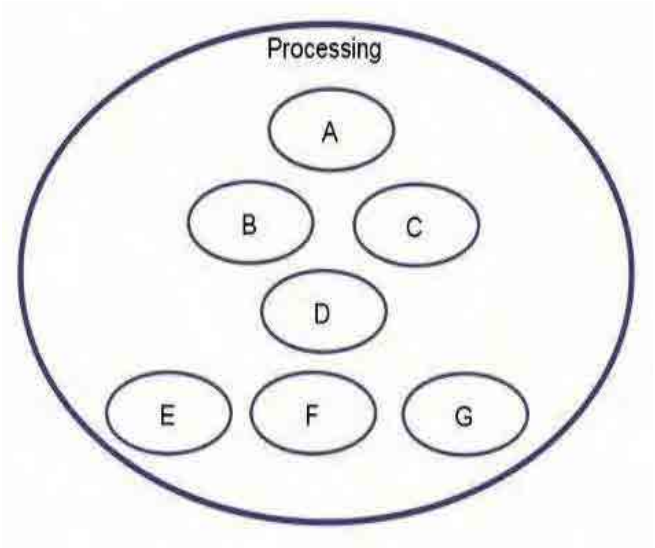
13.2.2.2 隐式并行调用关联集

从图形来说，当一个放大过程范围内的两个或多个子过程的椭圆顶点处于同一高度（在可允许的容忍度内）时，这些子过程只要在两者的前置条件都得到满足时就应并行开始。在此情况下，存在有一组从隐式调用关联的源过程到每一个并行过程的隐式调用关联。

闭合的子过程的椭圆顶点高度反映出这些子过程中的偏序。那些椭圆顶点处在同一高度的子过程并行开始。当这些子过程的最后一个结束时，即过程同步发生时，执行控制应尝试调用下一个子过程。如果有两个或更多处于相同高度的更低椭圆顶点的子过程时，执行控制将它们同时进行调用。如果没有更多子过程需要调用的话，执行控制将会返回到放大的可细化过程中去。

隐式平行调用关联OPL句子的语法应为：过程放大至平行的子过程A和子过程B.

示例：图 49 显示了具有下列偏序子过程：A、（B，C），D，（E，F，G）。B 和 C 在 A.完成时开始，D 在 B、C、E，F 中的更长过程结束时开始，而和 G 在 D 结束时开始。执行控制在 E、F 和 G 中的更长过程结束时返回到过程。



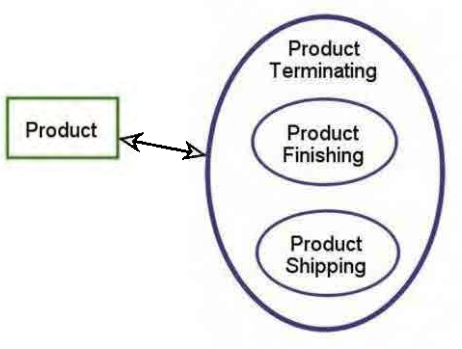
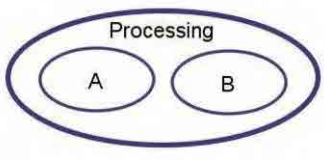
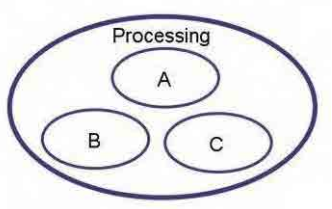
过程放大至A，平行的B和C，D，平行的E、F和G，以此排序。

图49 部分子过程顺序和隐式并行调用关联集

13.2.2.3 隐式调用关联小结

表24 总结了隐式调用关联。

表24　隐式调用关联小结

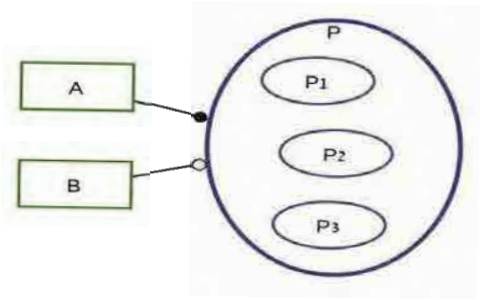
名称	语义	OPD 和 OPL 例子	来源	目 标
隐式调用关联	在一个放大过程的上下文内子过程完成时，子过程就会立即调用它下面的一个（多个）。	 产品终止放大到产品完成及产品运输这种顺序。	启动过程，其椭圆形顶点置于被启动过程上方。	启动过程，其椭圆形顶部置于启动过程的椭圆形顶点以下。
并行隐式调用关联集	上：一旦过程处理开始，子过程 A 和 B 就平行启动。 下: 一旦子过程 A 结束，子过程 B 和 C 就平行启动。	 过程处理并行放大至 A 和 B。  过程处理放大至 A 且并行至 B 和 C 这种顺序。	启动过程，其椭圆顶点置于一组启动进程之上，启动过程的椭圆形顶点处于相同高度（一个预定公差范围内）。	一组启动过程，其椭圆形顶点处于同一高度（容差范围内），并置于启动过程椭圆顶点的下方。

13.2.2.4　跨越情景的关联分配

13.2.2.4.1　关联分配的语义

从图形来看，附着于一个放大过程轮廓的程序关联具有分配性语义。离开一个附着于放大过程轮廓的关联应指该关联被分配并被附挂到每一个子过程中去。放大过程的轮廓具有类似于乘法符号之后的代数括号的语义，其将乘法运算符号分配给括号内的表达式。

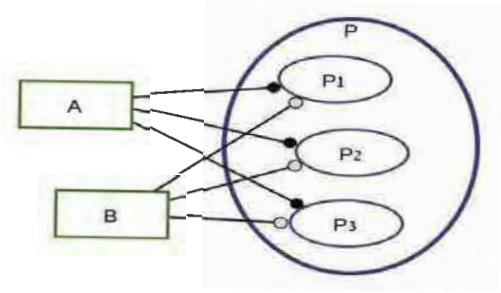
示例1：在图 50 中，OPD 的左侧和右侧是相等的，但左侧的一个更加清晰且不太杂乱。一个从 A 到 P 的代理关联意味着 A 处理子过程 P1、P2 和 P3。一个从 B 至 P 的仪器关联意味着子过程 P1、P2 和 P3 需要 B。 与代数中的相类似，假设代理（或仪器）关联是乘法运算符，A 则是乘数而放大则是加数，那么 A*P = A*(P1 + P2 + P3) = A*P1 + A*P2 + A*P3。



A 处理P。

P 需要B。

P 放大至P1、P2和P3，以此排序。



P 放大至P1、P2和P3，以此排序。

A 处理P1、P2和 P3。

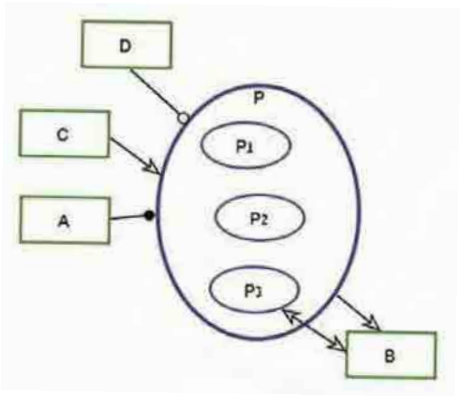
P1，P2和 P3需要 B。

图50　放大关联分配

如果一个使能器连接到一个放大轮廓的外部轮廓，它应至少连接到其子过程的其中一个。消耗和结果关联不应附着到一个放大过程的外部轮廓上去，因为这违反时间逻辑条件。在具有一个分布式消耗关联时，应尝试去消耗一个已经并非是第一次被运行的子过程所消耗的对象。同样地，一个分布式结果关联会试图创建一个已经存在的对象实例。

注1：当多个过程创建同一个对象，即对象的多个操作实例存在或多个过程影响或消耗相同对象时，建模者需要当心。 OPM 建模工具需要跟踪每一个对象和每一个过程的操作实例数量和身份以便能够实施模拟操作。

示例2：图 51 中左侧的 OPD 包含无效的消耗和结果关联，如 OPL 中所附注的。消耗关联导致 OPL 句子“P 消耗 C”。运用关联分配，其后果则是三个 OPL 句子 “P1 消耗 C”、“P2 消耗 C” 和 “P3 消耗 C”。然而，由于 P1 根据其时间顺序首先消耗 C，当 P2 或 P3 运行时，C 的相同实例就不存在，因此 P2 或 P3 就不能再次消耗 C。同样，B 的相同操作实例只有一次结果。右侧的 OPD 通过指定 P 的哪一个子过程来消耗 C 以及哪一个来生成 B（P2）来描绘有效关联。



A处理P。

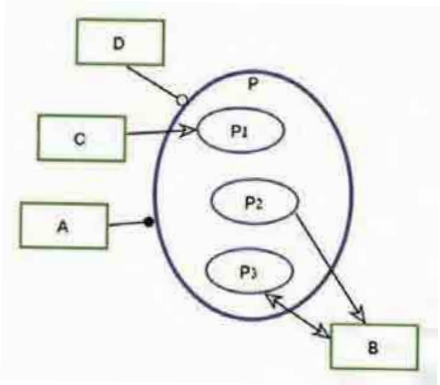
P 需要D。

P 放大至P1、P2和P3，以此排序。

P 消耗 C.。 - 无效!

P 生车B。 - 无效!

P3 影响B。



A 处理P。

P 需要D。

P 放大至P1、P2和P3，以该排序。

P1消耗 C。

P2 生成B。

P3 影响B。

图51　消耗和结果关联的关联分配限制

由于将消耗或结果关联附着到一个放大过程中是无效的，所以当一个过程被放大时，所有被连接到其的消耗和结果关联都应在初始时被默认地连接到其第一个子过程中去。

注2：建模工具能够自动建立默认语义，建模者可对其进行修改。

示例3：图 51 中，建模者只要一将 P 放大并将 P1 插入到其情景中，来自 C 的消耗关联的目标端就会从 P 迁移到 P1。同样，结果关联到 B 的源端也从 P 迁移到 P1。当建模者增加 P2 时就可将消耗关联的目标终端和/或结果关联的源的终端从 P1 迁移到 P2，如图 51 所示。

13.2.2.4.2 事件和条件关联约束

一个来自系统的对象或状态的事件关联不应从该过程的外侧跨越一个放大过程的边界以启动其在任何级别的某一个子过程，因为这就等同于尝试对同步（见14.2.3.5）放大过程所规定的时间顺序进行干扰。如果穿越事件关联从一个环境对象或状态发出的话，建模者应该对如何处理这种偶然性建模。

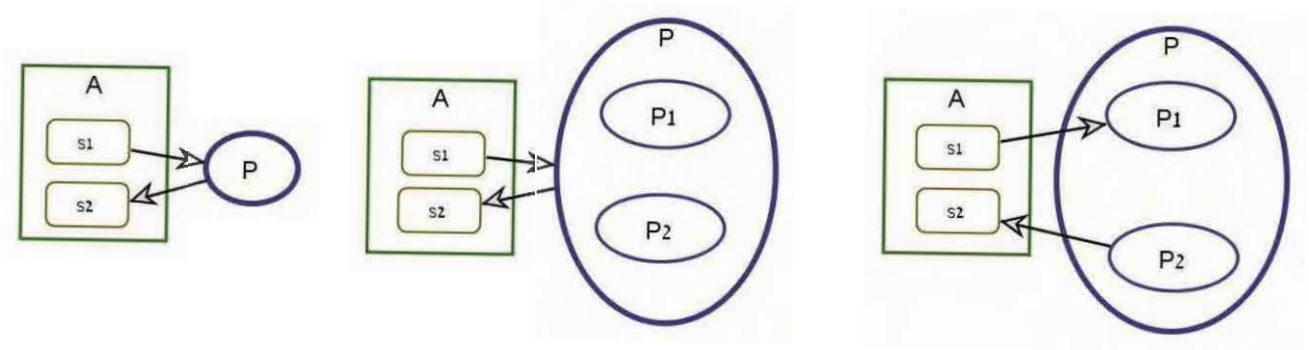
如果被跳过去的过程置于一个被放大的情景内的话，并且在该情景中有一个随后过程，那么执行控制就会启动该过程，否则执行控制移回到在放大的过程中去。

13.2.2.4.3 拆分状态-指定的转换关联

当一个将对象从输入状态改变成输出状态的过程被放大且包含多个子过程时，对象过程图（OPD）的图内或新图这两者之一将不被指定。为了恢复规范，建模者应将状态-指定输入关联和状态-指定输出关联这两者以一种时间上可行的方式附着到子过程的其中之一中去。将输入 - 输出指定关联对拆分为两个关联应表明拆分状态-指定转化关联对。

从图形上看，连接到一个拥有两个或更多状态的对象的两个关联跨越一个过程轮廓连接到拥有一个状态-指定输入关联和拥有一个状态-指定输出关联的不同子过程应表示拆分状态-转换关联。

示例1：图 52 中位于中间的 OPD 将左侧的 OPD 过程 P 放大但未被指定，因为 P2 或 P1 的每一个都有可能将 A 从 s1 变为 s2，或 P1 将从 s1 改变 A，P2 可将 A 变为 s2。右侧的 OPD 将最后这种情况建模从而产生一个从 A 的 s1 到 P1 的新的拆分输入关联和一个从 P2 到 s2 的新的拆分输出关联。



A 可以是s1或s2。
P 将A从s1变为s2。

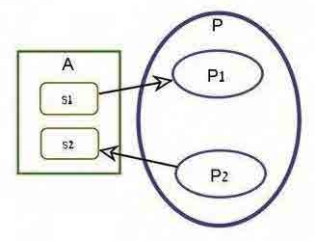
A可以是s1 或s2。
P 放大至P1和P2，以此排序。
P 将A 从s1变为s2。
- 未指定！

A 可以是s1 或s2。
P放大至P1和P2, 以此排序。
P1将A 从s1进行了改变。
P2 将A变为s2。

图52 拆分状态-指定转换关联以按规范求解

表 25 总结了拆分输入-输出-指定效果关联对。

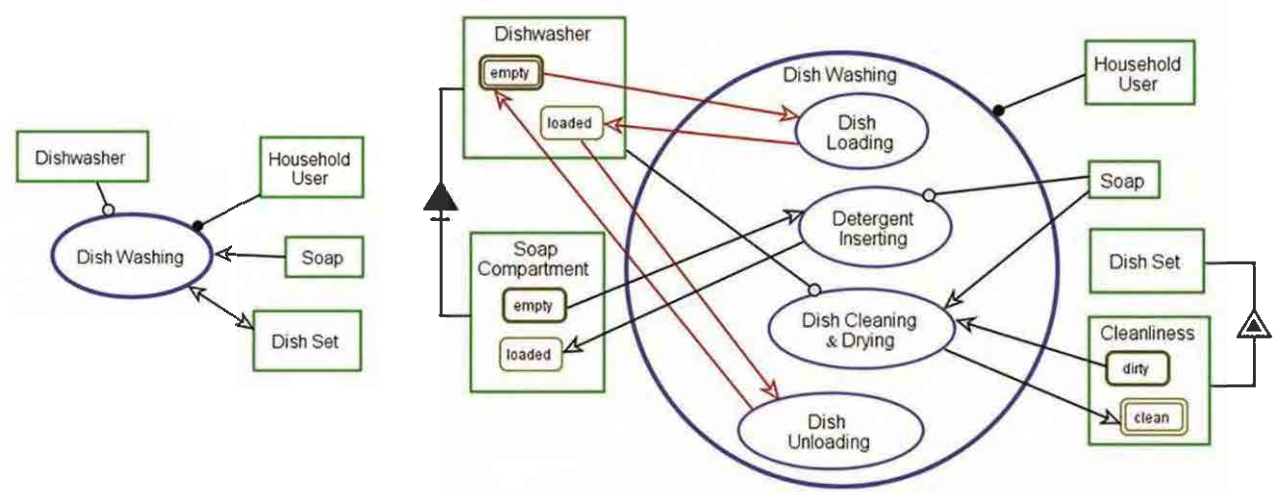
表25 拆分输入-输出指定效果关联对小结

名称	语义	OPD 和 OPL 例子	来源	目标
拆分输入-输出指定效果关联对 顶部箭头：拆分输入-指定的效果关联 底部箭头：拆分输出-指定的效果关联	一个被放大过程的早期子过程将对象从其输入状态中拿出。 同样被放大过程的后期子过程将对象改变至其输出状态。	 P1 将 A 从 s1 做了改变。 P2 将 A 变为 s2。	顶部箭头：一个受影响对象的输入状态 底部箭头：一个被放大过程的后期子过程	顶部箭头：一个被放大过程的早期子过程 底部箭头：受影响对象的输出状态

注1：没有拆分输入-指定效果关联的控制-关联版本 。

注2：一个对象在一个抽象 OPD 中可以起到一个仪器的作用，而在另一个派生物中发挥一个更详细和更具体的 OPD 被转换物的作用。在抽象 OPD 中，过程似乎不会影响对象，因为对象的初始状态与其最终状态相同。因此，抽象 OPD 中的对象是一个仪器，正如一个仪器关联所显示的那样。然而，在一个派生物具有更具体化的 OPD 中，同样的过程似乎确实将该对象状态从初始状态变成再次返回到初始状态。

示例2：图 53 中左侧的 SD（SD：洗碗系统），一个洗碗机的对象对于碗碟洗涤过程是一个仪器，因为在该抽象程度上洗碗机的状态无明显变化。在派生物的 OPD（SD1：洗涤被放大）中，洗涤放大到载荷（一套脏碟的）、清洗（将一套碗碟从脏变为净）和卸载。载荷将洗碗机从空的状态变为载满的状态，而卸载又将满载的状态变回到空状态，因此空既是初始也是最终状态（棕色关联重点）。虽然洗碗机在更详细的派生物 OPD 的系统图（SD）中是一个仪器，但洗碗机是一个受影响物，它变为满载，然后又变为空载。在系统图（SD）中唯一可见的效果是对一套碗碟的影响。



SD：洗碗系统

家庭用户处理洗碗。

洗碗需要洗碗机。

洗碗消耗肥皂。

洗碗影响一套碗碟。

SD1 洗碗放大

洗碗机由肥皂盒和其它零件组成。

洗碗机可为空或满。

洗碗机的空状态是初始状态和最终状态。

肥皂盒可为空状态或满状态。

肥皂盒空状态是初始状态。

碗碟展示了洁净性。

碗碟的清洁性可为净或脏。

碗碟洁净状态脏是初始状态。

碗碟清洁的干净状态是最终状态。

家庭用户处理洗碗。

洗碗放大至装载、洗涤液加入、洗涤和烘干以及碗碟卸载这种顺序。

碗碟装载将洗碗机从空状态变为满状态。

加入洗涤液需要肥皂。

洗涤液加入将肥皂从空状态变为满状态。

碗碟清洗和烘干需要洗碗机。

碗碟清洗和烘干需要肥皂。

碗碟清洗和烘干将碗碟的清洁性从脏变为干净。

碗碟卸载将洗碗机从满状态变为空状态。

图53　带有拆分状态转换关联的抽象作用

13.2.2.4.4　所涉及对象集的操作实例

作为关联分配的一种结果，以下约束应适用于被转换物的操作实例：

——处于一个过程的前过程对象集中的每一个被消耗物的操作实例应在该过程最详细的子过程的开始阶段不再存在，其消耗操作实例，并且操作实例不在该过程的后过程对象集中；

——处于一个过程的前过程对象集中、而此过程将该操作实例改变为一个过程性能的结果的每一个受影响物操作实例应在改变受影响物最详细的子过程开始时退出其输入状态；

——处于一个过程的后过程对象集中、而此过程将该操作实例改变为一个过程性能的结果的每一个受影响物操作实例应在改变受影响物最详细的子过程完成时进入其输出状态；以及，

——处于一个过程的后过程对象集中的每一个受影响物操作实例应在生成了结果物操作实例的最详细的子过程完成时以及操作实例还未处于该过程的前过程对象集时开始存在。

注1：一个过程 P 的执行将会为此而影响 B 状态变化的有状态对象 B 在改变 B 的 P 之最详细的子过程开始时退出输入状态，并进入到处于 P 或 P 的相同随后子过程的尾端进入到输出状态。由于执行过程 P 一定会花费一些时间，所以该对象 B 处于从其输入状态到其输出状态之间的转换：也就是它已经离开其输入状态，但尚未到达其输出状态。

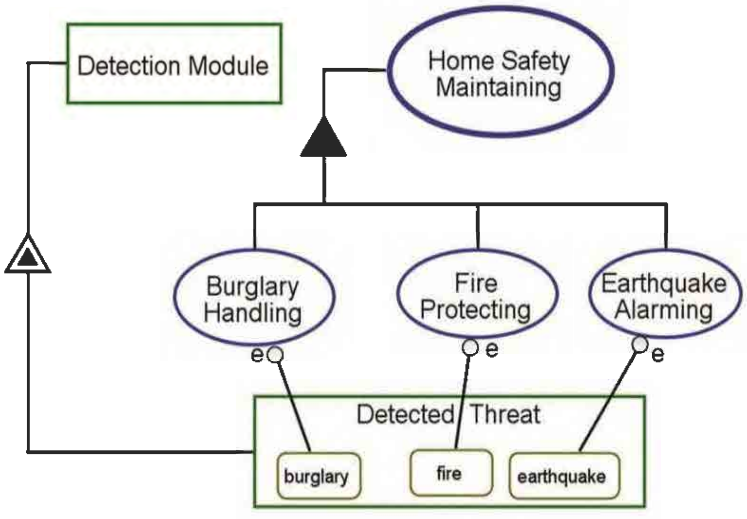
13.2.2.5 同步与异步过程细化

由于聚合-分散的基本结构关系没有规定任何过程性能的 “偏序”，所以同步过程细化建模应使用放大。

示例1：图 53 中的系统是同步的：在清洗碗盘的放大情景内有每一个子过程的固定及明确定义的顺序。

异步过程细化的建模应通过过程的图内聚合展开或作为一个新图聚合展开来使用聚合-分散基本结构关联。

示例2：图 54 描绘了一个执行家居安全维护功能的家居安全系统的一个部分，其包括子过程入室盗窃处理、消防保护以及地震报警。由于这三个子过程的顺序是未知的，所以 OPD 使用了来自该功能的具有一个聚合-分散关联的图内聚合展开而不是一个放大版的家居安全维护。家居安全维护放大到一个反复出现的系统过程（未显示）监测与检测，对其来说，检测模块是一个仪器，威胁出现是一个环境过程。



家居安全维护包含入室盗窃处理、消防保护和地震报警。

检测模块展示检测威胁。

检测威胁可以是入室盗窃、火灾或地震。

入室检测到的威胁引发入室盗窃处理，这需要入室检测到的威胁。

火灾检测到的威胁启动火灾保护，这需要火灾检测到的威胁。

地震检测到的威胁启动地震报警，这需要地震检测到的威胁。

图54 家居安全维护是一个异步系统

13.2.2.6 系统上下文表达

13.2.2.6.1 导航系统上下文

13.2.2.6.1.1 对象过程图（OPD）过程树

一个对象过程图（OPD）过程树也被称为OPD树，应是一个具有SD，即系统图的根结点以及其它将其OPD标签作为结点的OPD有向树的图。一个OPD树的有向边所具有的标签应是每个边从包含可细化过程的父对象过程图(OPD)指向子对象过程图（OPD），该子对象过程图通过用于同步子过程的新图放大或用于异步子过程的新图聚合展开详细阐述了一个对象过程图（OPD）中的可细化过程 。

13.2.2.6.1.2 对象过程图（OPD ）的对象树

与拥有单个树根的OPD过程树不同，OPD对象树更像是拥有很多树木的森林，每一个都产生于展开或放大的十分清晰的可细化物对象以将细节呈现出来。与辨别OPD过程树中发现的可能执行控制流程不同，OPD对象树应将一个对象作为一个层次结构的信息进行封装。系统执行应保持处于OPD对象树元素中及OPD对象树之间的依赖关系。

注：OPM工具提供了加强建模期间的依赖性维护的模型构建规则。

13.2.2.6.1.3 OPM 图标签

OPM系统名称应是指定了系统的OPM模型名称。一个OPD名称是辨别OPD过程树中的每一个OPD的名称。

系统图（SD）应是OPD树层次中用于OPD根的标签标识。该SD占据了OPD层次树中的0层，并应只拥有一个过程 ；更高数字的层级，即那些对应于连续细化的层则可具有一个或多个过程。 SD应只包含一个系统过程，其代表了向利益相关者提供功能值的总体系统功能。SD可含有一个或多个环境过程。

13.2.2.6.1.4 OPD 过程树边标签

由于一个OPD过程树中的每个详尽过程都拥有一个独特的名称，所以每个边标签都应参照处于另一个OPD中该过程的细化。OPD过程树中的每个边都应有一个标签。该标签应表示对应于隐式调用关联或展开关系的细化关系。考虑到每一个OPD都是一个对象，且整个OPD过程树是一个单一的OPD，每个边都应是一个带有 “放大可细化物名称而被细化的” 或 “展开可细化名称而被细化”标签的单向带标签结构关联。

一个OPD细化的OPL句子应是一个描述tierN层OPD中的可细化物与tierN+1层细化OPD之间的细化关系的OPL句子。

一个放大OPD细化的OPL句子的语法应为:TierN层OPD标签通过 TierN+1 层OPD标签中的放大可细化过程名称而被细化的。

一个展开OPD细化的OPL句子的语法应为：TierN 层 OPD标签是由TierN+1 层OPD标签中的展开可细化过程名称而被细化的。

注：第C.6条的几个OPD显示了边标签语法的使用。

13.2.2.6.1.5 系统地图和模型视图

一个系统地图应是一个OPD过程树，其明确描绘了每一个OPD（结点）的元素（事物和关联）内容。由于系统地图可能会变得非常大且笨重，所以机制应允许访问模型内容和要素中的关联。这些机制被统称为模型的视图模型，由模型事实组成，应包括所有事物以及它们在其中出现的OPD、OPD过程树以及OPD对象树的一个列表。

此外，OPM工具集应提供一种用于创建视图的机制，如相关OPL句子的OPD、对象和符合特定条件的过程。这些视图可以包括用于最小系统执行持续时间的关键路径或系统代理和工具的列表，或涉及到关联的一个特定类型或一组关联的对象和过程的OPD。

示例：一个 OPD 可通过以下来创建

a) 细化（展开或放大）一个对象，或

b) 收集并呈现出现于不同 OPD 中的一个新的 OPD 事物以表达系统子功能到系统模块对象的分配。

13.2.2.6.2 整个系统的 OPL 规范

一个OPL段落应是以文本形式共同指定相应OPD语义表达式的OPL句子的集合体。

注1：一个 OPL 句子名称使用 OPD 名称可先于每一个 OPL 段落的第一个 OPL 句子。

一个OPM系统模型应是对应于集合于OPD所呈现的连续性OPL段落的集合体。

一个系统的整体OPL规范应起始于一个OPL规范开头的标题。OPL段落紧跟着连续性模块中的标题，每一个都使用一个相应的OPD开始新的一行，之后紧接着OPL段落句子。

注2：OPL 段落的顺序一般始于系统图（SD）并遵循广度优先的顺序，除非建模者指定了一个与之不同的顺序。

示例：表 26 包含图 53 中 OPM 模型的整个 OPL 规范。

表26　清洗碗碟系统的整个 OPL 系统

碗碟清洗系统的 OPL 规范。

SD: 碗碟清洗系统。

家庭用户处理碗碟清洗。

碗碟清洗需要洗碗机。

碗碟清洗消费肥皂。

碗碟清洗影响碗碟集。

SD 通过放大 SD1 中的碗碟清洗 in SD1 来细化.

SD1: 碗碟清洗被放大。

碗碟清洗由肥皂盒和其它部件组成。

洗碗机可为空或满。

洗碗机的状态空状态是初始和最终状态。

肥皂盒可为空或满。

肥皂盒的状态空是初始状态。

一套碗碟展示了清洁性。

一套碗碟的清洁性可为净或脏。

一套碗碟的清洁性的状态脏是初始状态。

一套碗碟的清洁性的状态净是最终状态。

家庭用户处理碗碟清洗。

碗碟清洗放大至 碗碟载荷、放入洗涤液、碗碟清晰和烘干以及碗碟卸载，以这种顺序进行。碗碟载荷 将洗碗机从空变为满。

放入洗涤液需要肥皂。

放入洗涤液将肥皂盒从空变为满。

碗碟清洗和烘干需要洗碗机。

碗碟清洗和烘干消耗肥皂。

碗碟清洗和烘干将一套碗碟的清洁性从脏变为净。

碗碟卸载将洗碗机从满变为空。碗碟清洗系统 OPL 规范结束

13.2.3　OPM 事实一致性原理

OPM原理的事实一致性规定：

a)　出现于一个 OPD 中的模型事实应忠实于 OPM 系统模型中 OPD 的整个集合体，以及

b)　OPD 过程树中的 OPD 或一个 OPD 对象都不应包含与一个处于相同 OPD 或另一个 OPD 中的模型事实相矛盾的模型事实。

一个OPD中的事实可以是一个处于相同OPM系统模型内不同OPD中的事实的细化或抽象化。

注：该原理不排除建模者所期望的在OPD中尽情地表达任何数量的任何模型元素的可能性。由于一个关联在没有其所连接的事物的情况下无法存在，所以，如果一个关联出现的话，则位于其两端的两个事物也需要出现。

示例：一个 OPD 不可能表达“P 生成 A”这个事实，且处于同样 OPD 树中的一个相同或另一个 OPD 也不可能表达“P 消耗 A” 这个事实。然而，却允许一个 OPD 表达 “P 影响 A” 这个事实以及处于同样 OPD 树中的另一个 OPD 表达 “P 将 A 从 s1 变成 s2” 这个事实，因为后者的事实是一个细化，而非前者的一个矛盾。

13.2.4 程序关联的抽象模糊决定

13.2.4.1 抽象和程序关联优先级

缩小会将相关事物的集合体以及细化物和相关的关联抽象为一个可细化物。当建模者实施抽象化时，细化物与非细化物事物之间的程序关联应迁移到描绘了可细化物的OPD的上下文去。这种迁移可能会导致不同类型的两个或多个程序关联与一个对象和一个过程进行连接的这种情况发生。根据OPM原理（参见7.1.2）的程序关联独特性，一个对象或对象状态只由一个程序关联连接到一个过程中去。为维护这一原理，建模者应解决备选关联之间发生的冲突以确定哪个关联保留或哪个新的关联来接替处于抽象OPD中的候选者。详细信息的丢失与抽象化概念是一致的。

示例：图 55 展示了程序关联抽象化问题。在 SD1 中，从 P1 到 B 的结果关联比从 P2 到 B 的效果关联更为显著，所以当 SD1 被缩小到 SD 时，结果关联占有优势。

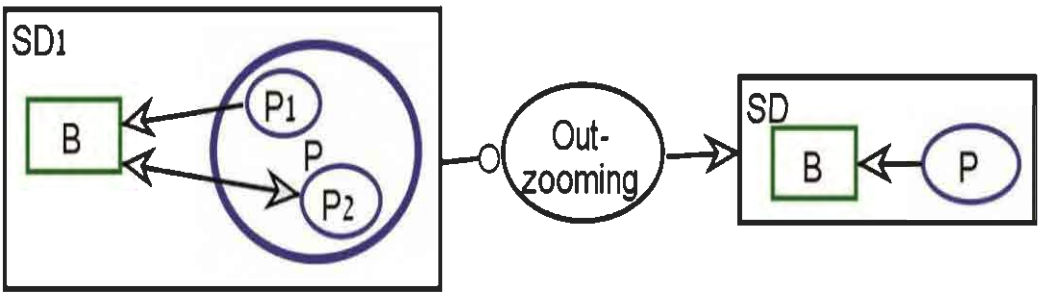


图55 抽象程序关联

语义强度和连接程序优先级是用来引导确定一个OPD被缩小或被折叠时哪种关联该保留和哪种关联该隐藏的两种概念。

一个程序关联的语义强度应是关联所携带信息的重要性。一个有关在创建或消除这种存在中所产生变化的信息比一个对有关现存事物所产生变化的信息更为重要。这两个相互矛盾程序关联的相对语义强度应确定关联的优先级。当两个或多个程序关联竞相保留其在一个细化的OPD抽象中的代表性时，占优势的那个关联就是具有最高强度的语义。

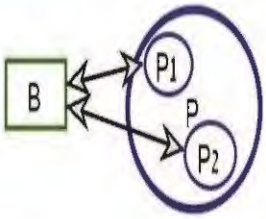
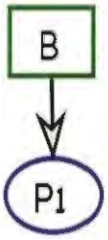
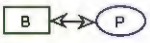
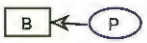
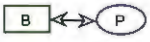
注：关联优先级概念使得建模者能够解决OPD上下文内表达式中的冲突，并能引导建模者建立不同细节程度上的恰当的程序关联。

13.2.4.2 转换关联中的优先级

转换关联包括结果、效果和消耗关联。鉴于对象的创建和消耗在语义上更为强大，即它们通过改变其状态具有比影响对象更高的语义强度，所以结果和消耗关联优先于效果连接，如在图55所示。然而，由于结果和消耗关联在语义上相等，当他们进行竞争时，占优势的关联则应为效果关联，因为效果关联可被认作是隐式地将一个对象从其存在状态变为其不存在状态，或反之亦然。

表27展示了转化关联优先级：左上角的P被缩小。纵列标题显示了P1与B之间的三种可能的转化关联，而表格单元则显示了P2和B之间的三种可能的关联。表格单元显示了P被缩小后B与P之间的占有优势的关联。具体来说，表27显示了效果、结果与消耗关联之间的冲突是如何得到解决的。例如，如果B到P1关联是消耗（中间列）以及B到P2的关联是结果（底行）的话，则P被放大后的B与P之间的关联是效果关联。被标记为 “无效” 的单元表明组合是不可能的。例如，看一下中间单元，我们注意到，如果P1消耗B，则B在P2稍后再次尝试消耗它时就不存在了。因此，两个消耗关联的组合是无效的。

表27　转换关联优先级：解决效果、结果和消耗关联之间的冲突

<table>
<tr><td colspan="2">放大的过程 P</td><td colspan="3">B-到-P1 关联</td></tr>
<tr><td colspan="2"> 
</td><td> 
</td><td> 
</td><td> 
</td></tr>
<tr><td rowspan="3">B-到-P2 关联</td><td> 
</td><td> 
</td><td>无效</td><td> 
</td></tr>
<tr><td> 
</td><td> 
</td><td>无效</td><td> 
</td></tr>
<tr><td> 
</td><td>无效</td><td> 
</td><td>无效</td></tr>
</table>

13.2.4.3　转换和使能关联中的优先级

转换关联在语义上比使能关联更强，因为转换关联表示被连接对象的创建、消耗或变化，而使能关联则仅表示启用性。一个转换关联应优先于一个使能关联，如图56中所示。

在使能关联内部，一个代理关联应优先于仪器关联，因为在人工系统中，过程以人类为中心，他们需要保证系统的正常运行。此外，只要有人的交互，就应存在有一个接口，并应将该信息提供给一个可细化物的建模者以便他们可因此而进行规划。

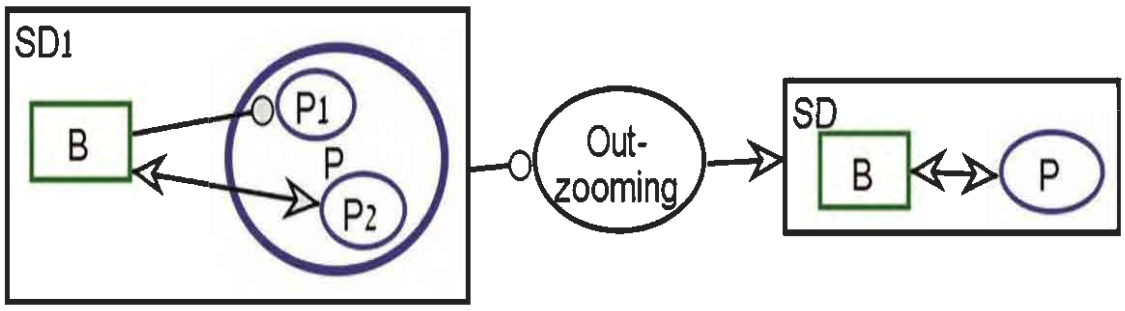


图56　用于转换和使能关联的关联优先级

将程序非控制关联的语义强度进行一个总结的话，优先级的优先级首要顺序应为：消耗 = 结果>效果>代理>仪器，其中 = 和 > 指的是关联的语义强度。状态-指定关联应比没有指定状态的基本关联具有更高的优先级。

13.2.4.4　同类非控制关联和控制关联中的次要优先级

几乎每一个非控制关联类型都具有一个有利于区分每一种程序关联内好的、中等的优先级的相应事件和条件关联。在优先级首要顺序的每一名成员内部用于次要优先级顺序的相对语义强度应拥有比其相应非控制关联更强的语义强度事件关联，而条件关联应具有比其对应的非控制关联更弱的语义强度。

一个事件关联的语义强度应比其相应非控制关联的语义强度更强，因为任何一个关联都即拥有其相应的非控制关联的语义又具有能够启动一个过程的事件的语义。一个条件关联的语义强度应比其相应的非控制关联的语义强度更弱，因为条件修饰符弱化了连接过程的前置条件满足标准。

13.2.4.5　程序关联语义强度总结

将基于主要与次要优先级之间的区别的程序关联语义强度进行总结的话，那么，优先级的完整顺序应按如下所示：

1. 消耗事件 > 消耗
2. 消耗 = 结果
3. 结果 > 消耗条件
4. 消耗条件 > 效果事件
5. 效果事件 > 效果
6. 效果 > 效果条件
7. 效果条件 > 代理事件
8. 代理事件 > 代理
9. 代理 > 代理条件
10. 代理条件 > 仪器事件
11. 仪器事件 > 仪器
12. 仪器 > 仪器条件

附 录 A
（资料性附录）
EBNF 中的 OPL 形式化语法

A.1 通则

OPL 是一个英语子集，其应以文本形式来表达 OPD 集用图形形式所表达的 OPM 规范。

对象过程语言（OPL）是一种具有双重目的的语言。首先，它服务于从事分析和设计一个系统的域专家和系统建筑师，如电子商务系统或基于 Web 的企业资源规划系统。第二，它为自动生成所设计应用程序提供了的坚实基础。

对象过程语言（OPL）是图形对象过程方法（OPM）系统规范的文字对应物，对应于 OPD 集中的图解说明。OPL 是自动生成的系统文本描述，是自然英语的一个子集。由于没有编程语言的特性及过度隐晦的细节，OPL 句子使那些没有技术或编程经验的人能够理解。

由于 OPM 支持的模型表达具有众多种类，下面的 EBNF 中的 OPL 语法表达自然是不完整的，如 12.7 中关于概率和第 13 条的执行路径管理的陈述可能均缺乏 EBNF 表达式。众多参与约束的多样性，特别是那些可用数学公式来表达的，在该附录中均不具有形式化规范。

A.2 处于OPD背景中的OPL

本附录提供了一个符合 ISO/ IEC14977：1996 的 OPL 形式化规范，该规范源自于第 6 至 13 条中出现的不同 OPD 图形结构。为给读者提供帮助，本附录引用了相应 OPD 子条款，其中适当的条款/子条款标题有助于根据建模要素语法形式将 EBNF 区分开来。

A.3 初级结构

A.3.1 EBNF 语法

下列语法使用 ISO/ IEC14977：1996 中所描述的 EBNF 符号：代表扩展巴科斯范式(EBNF)的每一个操作符及其隐式优先级的每一个操作符的正常特性应是（顶部最高优先级）：

* repetition-symbol
- except-symbol
, concatenate-symbol
| definition-separator-symbol
= defining-symbol
; terminator-symbol

正常优先级应被下面的方括号覆盖：
' first-quote-symbol '
" second-quote-symbol "

(* start-comment-symbol end-comment-symbol *)

(start-group-symbol end-group-symbol)
[start-option-symbol end-option-symbol]
{ start-repeat-symbol end-repeat-symbol }
? special-sequence-symbol ?

注1：用引号“”括起来的空格字符”表示需要一个文字空格字符，否则空格字符和行尾的（所谓的空白）则无意义。

注2：一个元标识符可出现在一个规则的左右两侧，因此要启用递归式。

注3：第一个引号-符号标识了 OPL 变量标签的语法要素，其是出现在 OPD 图形模型和 OPL 句子中的名称和值。这些特殊语法要素只能在 A.3.2 中找到。

注4：第二个引号-符号标识了 OPL 常量语法要素，是作为图形要素组态的释译和一个 OPD 中的连接标签的单词和短语而出现的。

注5：从 A.3.2 开始直到附件 A 的剩余部分，除标题外的所有文本都符合 ISO/ IEC14977：1996。

A.3.2 基本申明

(* Region OPL EBNF *)
(* 区域基本申明: 下列基本申明定义了某些串：*)

non zero digit = '1' | '2' | '3' | '4' | '5' | '6' | '7' | '8' | '9' ;
decimal digit = '0' | non zero digit ;
positive integer = non zero digit, {decimal digit} ;
positive real number = {decimal digit}, ".", decimal digit, {decimal digit} ;
upper case letter = 'A' | 'B' | 'C' | 'D' | 'E' | 'F' | 'G' | 'H' | 'I' | 'J' | 'K' | 'L' | 'M'
| 'N' | 'O' | 'P' | 'Q' | 'R' | 'S' | 'T' | 'U' | 'V' | 'W' | 'X' | 'Y' | 'Z' ;
lower case letter = 'a' | 'b' | 'c' | 'd' | 'e' | 'f' | 'g' | 'h' | 'i' | 'j' | 'k' | 'l' | 'm'
| 'n' | 'o' | 'p' | 'q' | 'r' | 's' | 't' | 'u' | 'v' | 'w' | 'x' | 'y' | 'z' ;
letter = upper case letter | lower case letter ; string character = letter | decimal digit | '_' | '-' | '&' | '/'
| ' ' ;(* 注：一个串字符可是一个空格*)
name = letter, {string character} ; (* 注：第一个字符是一个字母 *)
大写单词 = 大写字母 {string character} ;
非大写单词 = 小写字母 {string character} ;
非大写短句 = 非大写单词 { ' ', (non capitalized word | capitalized word) } ;
类型标识符 = " boolean"
| " string"
| number type
| " enumerated" ;
prefix = " unsigned" ;
number type = [prefix], " integer"
| " float"
| " double"
|"short"
| " long" ;

分散限制参与限制= positive integer | positive real number;
参与约束 = lower single
| upper single
| lower plural
| upper plural
| ("0" | participation limit, [" to ", participation limit]) ;
表达约束 = " where ", name, ((logical operation, value name)
| (logical begin set, (name | value name), { ", ", [(name | value name)] }, logical end set)) ;
lower single = "a " | "an " | "an optional " | "at least one " ;
upper single = "A " | "An " | "An optional " | "At least one " ;
lower plural = "optional " | "many " ;
upper plural = "Optional " | "Many " ;
range clause = " is ", value name | " ranges from ", value name, " to ", value name ;
logical operation = "=" | "<" | ">" | "<=" | " >=" ;
logical begin set = " in { " ;
logical end set = " }" ;

(* 分散约束具有多种表达形式，基本声明并不包括那些形式的全部。 *)

(* 在 OPL 陈述中所保留的字和符号由第二个引用符号限定 *)

(* EndRegion: 基本申明*)

A.3.3 OPL 特殊序列

(* 区域特殊顺序 - 该区定义了所有特定顺序，如新行、复数对象和过程*)

新行 new line = ?应用程序特定字符序列产生了一行的反馈，紧接着返回到该行的第一个字符位置 ? ;
测量单位 = ?任何指定或通常所理解的时间、空间、数量或质量的测量 ? ;
值名称 = ? 一个适用于相关测量单位的数字或名称? ;
单数对象名称=? 大写单数名词短语? ; (* 见 6.1.2 *)
plural object name = ? capitalized plural noun phrase ? ;
单个过程名称 = ? capitalized gerund phrase ? | ? capitalized singular noun phrase ? ;
plural process name = ? capitalized gerund phrase ? | ? capitalized plural noun phrase ? ; (* 见 6.2.2 *)
parent OPD = ? OPD from which a new-diagram in-zooming or new diagram unfolding occurs ? ;
child OPD = ? OPD resulting from a new-diagram in-zooming or new diagram unfolding ? ;
max duration time units = ? value of maximum duration in time units for process execution ? ;
min duration time units = ? value of minimum duration in time units for process execution ? ;
(* EndRegion: Special Sequences *)

A.4 OPL语法

A.4.1 OPL 文档结构

(* Region OPL document *)

OPL 句子 = OPL 句子, { new line, OPL sentence} ;
OPL 句子 = OPL 形式化句子, "." ;
OPL 形式化句子 = 事物描述句子
| 程序句子
| 结构句子
| 上下文管理句子;

A.4.2 OPL标识符

(* 区域: 标识符 – 该区域定义了贯穿于语法中所使用的所有标识符 *)

对象标识符 = singular object name, [" in ", measurement unit], [range clause]
| singular object name, " object", [" in ", measurement unit], [range clause]
| plural object name, [" in ", measurement unit], [range clause]
| plural object name, " objects", [" in ", measurement unit], [range clause] ;
过程标识符 = singular process name
| singular process name, " process"
| plural process name
| plural process name, " processes" ;

事物标识符 = object identifier
| process identifier ; (* 见 6.1 和 6.2 *)
状态标识符 = non capitalized word ;
tag expression = non capitalized phrase ;

(* EndRegion: Identifiers *)

A.4.3 OPL列表

(* 区域: 列表 – 该区域定义了不同的列表：对象列表、过程列表、具有可选状态列表的对象*)

过程列表 = 过程标识符
| process identifier, [{", ", process identifier}], " and ", process identifier ; (* see11.1 *)
process Or list = process identifier, [{", ", process identifier}], " or ", process identifier ;
process Xor list at beginning = "One of ", process Or list ;
process Xor list at end = "one of ", process Or list ;

object list = object identifier
| object identifier, [{", ", object identifier}], " and ", object identifier ; (* see 11.1 *)
具有可选状态的对象 = [state identifier, " "], object identifier ;
(* 具有可选状态的对象可使用对象标识符来取代许多 OPL 表达式中的对象标识符 *)

具有可选状态列表的对象 = 具有可选状态的对象
| 具有可选状态的对象, [{", ", object with optional state}],

" and ", object with optional state ;

对象 Or 列表 = object with optional state, [{", ", object with optional state}], " or ", object with optional state; (* see 11.2 *)

对象 Or 无状态列表= object identifier, [{", ", object identifier}], " or ", object identifier ;

初始时对象 Xor 列表 = "One of ", object Or list ;
结束时对象 Xor 列表 = "one of ", object Or list ;
结束时无状态对象 Xor 列表= "one of ", object list ;

状态列表 = state identifier
| state identifier, [{", ", state identifier}], " and ", state identifier ;
state Or list = state identifier, [{", ", state identifier}], " or ", state identifier ;
state Xor list at end = "one of ", state Or list ;

(* EndRegion: Lists *)

A.4.4 OPL事物表述

A.4.4.1 事物描述句子

(* Region: 事物描述 – 该区域定义了所有事物描述的句子*)

事物描述句子 = 通用属性句子
| 类型属性句子
| 状态描述句子;

A.4.4.2 通用属性句子

通用属性句子=事物标识符,
" is ", [essence], [affiliation], [perseverance] ; (* 见 6.3.3 *)

本质 = "Informatical" | "Physical" ; (* 物理是本质的非默认值，其默认值为信息性的。*)

隶属 = "Systemic" | "Environmental" ; (* 环境是所属的非默认值，其默认值是系统性的。*)

韧性 = "Persistent" | "Transient" ; (* 瞬间是韧性的非默认值，其默认值为持久性。*)

A.4.4.3 类型描述句子

类型描述句子 = 对象标识符," 是。。的类型 ", 类型标识符;

A.4.4.4 状态描述句子

状态描述句子 = 状态枚举句子
| 初始状态句子
| 结束状态句子

| 默认状态句子
| 组合状态句子; (* 见 6.3.5 *)
状态枚举句子 = 对象标识符, " is ", 状态标识符
| object identifier, " can be ", state identifier, [{", ", state identifier}], " and ", state identifier
| object identifier, " can be ", state identifier, [{", ", state identifier}], " and other states" ;
initial states sentence = single initial states sentence
| 多重初始状态句子;
单个初始状态句子 = "State ", state identifier, " of ", object identifier, " is initial" ;
multiple initial states sentence = "States ", state list " of ", object identifier, " are initial" ;
final states sentence = single final state sentence
| 多重最终状态句子;
单个最终状态句子 = "State ", state identifier, " of ", object identifier, " is final" ;
multiple final state sentence = "States ", state list, " of ", object identifier, " are final" ;
default state sentence = "State " state identifier, " of ", object identifier, " is default" ;
combined state sentence = object identifier, {" is initially ", [state identifier | state identifier,
{" and ", state identifier}], " and finally ", state OR list } ;
input state = state identifier ; (* the state or states of the associated object in a process precondition set *)
Output state = state identifier ; (* the state or states of the associated object in a process
postcondition set *)
active process identifier = process identifier ;
(* EndRegion: Thing Description *)

A.4.5 OPL 程序句子

A.4.5.1 程序句子

(* 区域: 程序句子.– 该区域定义了所有程序句子*)
程序句子 = 转换句子
| 启动句子
| 控制句子; (* 见 7.1.1 *)

A.4.5.2 OPL 转换

A.4.5.2.1 转换句子

(* 区域: 转换句子 – 该区域定义了消耗、结果、效果和变化句子以及其变量 *)
转换句子 = 消耗句子
|结果句子
| 效果句子
| 改变句子; (* 见 8.1.1 和 8.3.3 *)

A.4.5.2.2 消耗句子

消耗句子 = (过程标识符, " 消耗 ", 具有可选状态列表的对象)
| 消耗选择句子; (* 见 8.1.2 *)
消耗选择句子消费选择句子 = 消耗 Or 句子
| 消耗 Xor 句子; (* 见 11.3 *)
消耗或句子消耗 Or 句子 = 消耗源 Or 句子
| 消耗目标 Or 句子;
消耗源 Or 句子 = 过程标识符, " 消耗至少句子。。之一", 对象或清单 Or 列表;
消耗目标或句子 = "至少其中之一", 过程 Or 列表,
" 消耗", 具有可选状态的对象;

消耗异或句子= 消耗源 XOR 句子
| 消耗目标 XOR 句子;
消耗源 XOR 句子 = 过程标识符, " 仅消耗一个", 终端的对象 XOR 列表;
消耗目标 XOR 句子 = "仅一个", " 消耗",
具有选择状态的对象;

A.4.5.2.3 结果句子

结果句子 = (过程标识符, " 生成", 带有选择状态清单的对象)
| 结果选择句子; (* 见 8.1.3 *)
结果选择句子 = 结果或句子
| 结果 Xor 句子; (* 见 11.3 *)
result Or 句子 = 结果源 Or 句子
|结果目标 Or 句子;
结果源 Or 句子 = "At least one of ", process Or list, " yields ", object with optional state ;
result destination Or sentence = process identifier, " yields at least one of ", object Or list ;
result Xor sentence = result source Xor sentence
| 结果目标 Xor 句子;
结果源 Xor 句子 = "Exactly ", 过程 Xor 终端列表, " yields ", 具有可选状态对象; 结果目标 Xor 句子 = 过程标识符 , " yields exactly ", object Xor list at end ;

A.4.5.2.4 效果句子

效果句子 = (process identifier, " affects ", object list)
| effect select sentence ; (* 见 8.1.4 *)
效果选择句子 = 效果 Or 句子
| 效果 Xor 句子;
效果 Or 句子 = 效果对象 Or 句子
| 效果过程 Or 句子; (* 见 11.3 *)
effect 对象 Or 句子 = 过程标识符, " affects at least one of ", object Or list nostates ;
effect process Or sentence = "At least one of ", process Or list, " affects ", object identifier ;
effect Xor sentence = effect object Xor sentence

| effect process Xor sentence ;
effect object Xor sentence = process identifier, " affects exactly ", object nostates Xor list at end ;
effect process Xor sentence = "Exactly ", process Xor list at end, " affects ", object identifier ;

A.4.5.2.5 更改句子

更改句子 = 入出指定的更改句子
| 输入指定更改句子
| 输出指定更改句子; (* 见 8.3.3.1 *)
入出指定 = (过程标识符, " 更改", 入出对象更改清单)
| 输入输出指定更改选择句子; (* 见 8.3.3.2 *)
入出对象更改清单 = 输入输出对象更改短句
| 入出对象更改短句, [{", ", 入出对象更改短句}],
" and", 入出对象更改短句;
入出对象更改短句 = 对象标识符, " from", 输入状态, " to", 输出状态;
输入出指定更改选择句子 = 入出指定更改 Or 句子
| 入出指定更改 Xor 句子;
输入输出入出指定更改 Or 句子 = (过程标识符, " 更改", Or 入出对象更改清单)
| (过程 Or 清单, " 更改", 入出对象更改短句)
| 入出指定更改状态 Or 句子;
Or 入出对象更改清单 = 入出对象更改短句, [{", ", 入出对象更改短句}],
"Or", i 入出对象更改短句;
输入输出入出指定更改状态 Or 句子 = (过程标识符, " 更改 ", 对象标识符,
"从", 状态 Or 清单, " 到", 状态标识符)
| (过程标识符, " 更改 ", 对象标识符,
"从",状态标识符, " to ", state Or list) ;
入出特定变化 Xor 句子 = in out specified change object Xor sentence
| 入出指定变化过程 Xor 句子
| 入出指定变化状态 Xor 句子 ;
入出指定变化对象 Xor 句子 = process identifier, " changes one of ",
Or In out object change list ;

入出指定变化过程 Xor 句子 = process Xor list at beginning, " changes ",
in out object change phrase ;
入出指定变化状态 Xor 句子 = (process identifier, " changes ", 对象标识符,
" from ", state Xor list at end, " to ", 状态标识符)
| (过程标识符, " 更改 ", 对象标识符, "从", 状态标识符, "到", 结尾处状态异或清单) ;

输入指定更改句子 = (过程标识符, " 更改", 输入对象更改清单)
| 输入指定变化选择句子 ; (* 见 8.3.3.3 *)
输入对象变化短语 = 对象标识符, " from ", input state ;

input object change list = input object change phrase
| 输入对象变化短语, [{", ", input object change phrase }], " and ",
输入对象变化短语 ;
输入指定变化选择句子 = 输入指定变化 Or 句子
| 输入指定变化 Xor 句子;
输入指定变化 Or 句子 = (process identifier, " changes ", Or input object change list)
| (process Or list, " changes ", input object change phrase)
| (process identifier, " changes ", object identifier, " from ", state Or list) ;
Or 输入对象变化列表 = 输入对象变化短语, [{", ", input object change phrase }], " or ",
input object change phrase ;
输入指定变化 Xor 句子 = (process identifier, " changes one of ", Or input object change list)
| (process Xor list at beginning, " changes ", input object change phrase)
| (process identifier, " changes ", object identifier, " from ", state Xor list at end) ;

输入指定变化句子 = (过程标识符, " changes ", output object change list)
| output specified change select sentence ; (* 见 8.3.3.4 *)
输出对象变化列表 = output object change phrase
| output object change phrase, [{", " output object change phrase }], " and ",
output object change phrase ;
输出对象变化短语 = object identifier, " to ", output state ;
输出指定变化选择句子 = output specified change Or sentence
| output specified change Xor sentence ;
输出指定变化 Or 句子 = (process identifier, " changes ", Or output object change list)
| (process Or list, " changes ", output object change list)
| (process identifier, " changes ", object identifier, " to ", state Or list) ;
Or 输出对象变化列表 = output object change phrase, [{", ", output object change phrase }], " or ",
output object change phrase ;
输出指定变化 Xor 句子 = (process identifier, " changes one of ", Or output object change list)
| (process Xor list at beginning, " changes ", output object change phrase)
| process identifier, " changes ", object identifier, " to ", state Xor list at end ;
(* EndRegion: Transforming sentences *)

A.4.5.3 OPL 使能器

A.4.5.3.1 使能句子

(* 区域: 使能句子 – 该区域定义了代理和仪器句子以及其可能的变量*)

启用句子使能句子 = 代理句子
| 仪器句子; (* 见 8.2.1 *)

A.4.5.3.2 代理句子

代理句子 = (具有可选状态列表的对象, " 处理", 过程标识符)
| 代理选择句子; (* 见 8.2.2 和 11.3 *)

代理选择句子 = 代理 Or 句子
| 代理 Xor 句子;
代理 Or 句子 = 代理源 Or 句子
| 代理目标 Or 句子;
代理源 Or 句子 = "At least one of ", object Or list, "handles", process identifier ;
agent destination Or sentence = object with optional state, "handles at least one of ", process Or list ;
agent Xor sentence = agent source Xor sentence
| agent destination Xor sentence ;
agent source Xor sentence = "Exactly ", object Xor list at end, " handles ", process identifier ;
agent destination Xor sentence = object with optional state, " handles exactly ", process Xor list at end ;

A.4.5.3.3 仪器句子

仪器句子 = (process identifier, " requires ", object with optional state list)
| 仪器选择句子; (* 见 8.2.3 和 11.3 *)

仪器选择句子 = 仪器 Or 句子
| 仪器 Xor 句子 ;
仪器 Or 句子 = 仪器源 Or 句子
| 仪器目标 Or 句子;
仪器源 Or 句子 = process identifier, " requires at least one of ", object Or list ;
instrument destination Or sentence = "At least one of ", process Or list, " requires ",
object with optional state ;
仪器 Xor 句子 = instrument source Xor sentence
| instrument destination Xor sentence ;
仪器源 Xor 句子 = process identifier, " requires exactly ", object Xor list at end ;
instrument destination Xor sentence = "Exactly ", process Xor list at end, " requires ",
object with optional state ;

(* EndRegion: Enabling sentences *)

A.4.5.4 OPL 控制流

A.4.5.4.1 控制句子

(* Region : 控制句子 – 该区域定义了所有与系统中控制流相关的句子*) 控制句子 = 时间句子
| 条件句子
| 调用句子
| 异常句子; (* 见 8.5.1 *)

A.4.5.4.2 事件句子

事件句子 = 消耗事件句子
| 效果事件句子
| 代理事件句子
|仪器事件句子; (* 见 8.5.2 *)

消耗事件句子 = 具有可选状态的对象, " initiates ", process identifier,
", which consumes ", object identifier ;
(* 见 11.5 和 11.6 用于连接扇的额外语法*)

效果事件句子 = 简单效果事件句子
| 入出指定效果事件句子
| 输入指定效果事件句子
| 输出指定效果事件句子;
简单效果事件句子 = object identifier, " initiates ", process identifier, ", which affects ",
object identifier ;
入出指定效果事件句子= input state, object identifier, " initiates ", process identifier, ",
which changes ", in out object change phrase ;
输入指定效果事件句子 = input state, object identifier, " initiates ", process identifier, ",
which changes ", object identifier, " from ", input state ;
输出指定效果事件句子= object identifier, " in any state initiates ", process identifier, ",
which changes ", object identifier, " to ", output state ;

代理事件句子 = object with optional state, " initiates and handles ", process identifier ;
仪器事件句子 = object with optional state, " initiates ", process identifier, ",
which requires " object with optional state ;

A.4.5.4.3 条件句子

条件句子 = 条件转换句子
| 条件使能句子;
条件转换句子 = 条件消耗句子
| 条件状态指定消耗句子
| 条件效果句子 ; (* 见 8.5.3.1 和 8.5.3.3 *)

条件消耗句子 = (process identifier, " occurs if ", object identifier,
" exists, in which case ", object identifier, " is consumed, otherwise
", process identifier, " is skipped ")
| ("If ", object identifier, " exists then ", process identifier, " occurs and consumes ", object
identifier, ", otherwise bypass ", process identifier) ;

条件状态指定消耗句子 = (process identifier, " occurs if ", object identifier,
" is ", input state, ", in which case ", object identifier, " is consumed, otherwise

", process identifier, " is skipped ")
| ("If ", input state, object identifier, " exists then ", process identifier,
" occurs and consumes ", object identifier, ", otherwise bypass ", process identifier) ;

条件效果句子 = 简单条件效果句子
| 入出指定条件效果句子
| 输入指定条件效果句子
| 输出指定条件效果句子;
简单条件效果句子 = (process identifier, "occurs if ", object identifier,
" exists, in which case ", process identifier, " affects ", object identifier,
", otherwise ", process identifier, " is skipped ")
| ("If ", object identifier, " exists then ", process identifier, "occurs and affects ",
object identifier, ", otherwise bypass ", process identifier) ;
入出指定条件效果句子 = (process identifier, " occurs if there is ",
input state, object identifier, ", in which case ", process identifier, " changes ",
in out object change phrase, ", else ", process identifier, " is skipped ")
| (process identifier, " occurs if there is ",
input state, object identifier, ", in which case ", process identifier, " changes ",
in out object change phrase, ", otherwise bypass ", process identifier) ; 输
入指定条件效果句子 = (process identifier, " occurs if there is ",
input state, object identifier, " in which case ", process identifier, " changes ",
object identifier, " from ", Input state, ", else ", process identifier, " is skipped ")
| (process identifier, " occurs if there is ", input state, object identifier, 0
" in which case ", process identifier, " changes ", object identifier, " from ",
Input state, ", otherwise bypass ", process identifier) ;
输出指定条件效果句子 = (process identifier, " occurs if ",
object identifier, " exists, in which case ", process identifier, " changes ",
object identifier, " to ", output state, ", otherwise ", process identifier, " is skipped ")
| (process identifier, " occurs if ", object identifier, " exists, in which case ", process
identifier, " changes ", object identifier, " to ",
output state, ", otherwise bypass ", process identifier) ;
条件使能句子 = conditional agent sentence
| 条件仪器句子; (* 见 8.5.3.2 *)
条件代理句子 = (process identifier, " occurs if ", object with optional state,
" exists, else ", process identifier, " is skipped")
| (process identifier, " occurs if ", object with optional state,
" exists, else bypass ", process identifier) ;
条件仪器句子 = (process identifier, " occurs if ", object with optional state,
" exists, else ", process identifier, " is skipped")
| (process identifier, " occurs if ", object with optional state,
" exists, else bypass ", process identifier) ;

A.4.5.4.4 调用句子

调用句子 = (process identifier, " invokes ", process list)
| (process identifier, " invokes itself ")
| 调用选择句子; (* 见 8.5.2.5 和 11.3 *)

调用选择句子 = invocation Or sentence
| 调用 Xor 句子;

调用 Or 句子 = ("At least one of ", process Or list, " invokes ", process identifier)
| (process identifier, " invokes at least one of", process Or list) ;
调用 Xor 句子 = ("Exactly one of ", process Or list, " invokes ", process identifier)
| (process identifier, " invokes exactly ", process Xor list at end);

A.4.5.4.5 异常句子

异常句子 = 超时异常句子
| 时间不足异常句子 ; (* 见 8.5.4 *)
超时异常句子= active process identifier, " occurs if duration of ", process identifier, "exceeds ", max duration time units ;
时间不足异常句子 = active process identifier, " occurs if duration of ", process identifier, " falls short of ", min duration time units ;

(* EndRegion: Control sentences *)

(* EndRegion: Procedural sentences *)

A.4.6 OPL结构句子

A.4.6.1 结构句子

(* 区域: 结构句子 - 该区域定义了所有连接静态、不依赖时间和持久关系中的事物 *)

结构句子 = 标签结构句子
| 聚合句子
| 表征句子
| 展示句子
| 特化句子
| 实例化句子; (* 见 9.1 *)

A.4.6.2 OPL 带标签结构

A.4.6.2.1 带标签结构句子

标签结构带标签结构句子 = 单向带标签结构句子
| 双向带标签结构句子 ;

A.4.6.2.2 单向带标签结构句子

单向带标签结构句子=单个连接单向标签句子
| forked tagged structural sentence ; (* 见 9.2.1 和 10.2 *)
单个关联单向标签句子 = nullTag unidirectional object tagged structural sentence
| nullTag unidirectional process tagged structural sentence
| non nullTag unidirectional object tagged structural sentence
| non nullTag unidirectional process tagged structural sentence ; (* 见 9.2.2 和 10.2 *)

零标签空标签单向对象带标签结构句子 = [participation constraint, " "],
source object, uniDirNullTag, [participation constraint, " "], destination object ;
nullTag unidirectional process tagged structural sentence = [participation constraint, " "],
source process, uniDirNullTag, [participation constraint, " "], destination process ;
non nullTag unidirectional object tagged structural sentence = [participation constraint, " "],
source object, " ", forward tag, " ", [participation constraint, " "], destination object,
[expression constraint] ;
非零单向标签非空单向标签过程带标签结构句子 = [participation constraint, " "],
source process, " ", forward tag, " ", [participation constraint, " "], destination process ;
分叉带标签结构句子 = forked nullTag object tagged structural sentence
| forked nullTag process tagged structural sentence
| forked non nullTag object tagged structural sentence
| forked non nullTag process tagged structural sentence ;
分叉空标签对象带标签结构句子 = [participation constraint, " "], source object,
uniDirNullTag, object tine set ;
分叉空标签过程带标签结构句子 = [participation constraint, " "], source process,
uniDirNullTag, process tine set ;
分叉非空标签对象带标签结构句子 = [participation constraint, " "], source object, " ",
forward tag, " ", object tine set ;
分叉非空标签过程带标签结构句子 = [participation constraint, " "], source process,
" ", forward tag, " ", process tine set ;
object tine set = tine object | ((tine object, [{", ", tine object }], " and ", (tine object | "more")),
[(", ordered by ", order criteria) | (", in that sequence")]) ;
process tine set = tine process | ((tine process, [{", ", tine process }], " and ", (tine process | "more")),
[(", ordered by ", order criteria) | (", in that sequence")]) ;
顺序准则 = name ;
tine object = [participation constraint, " "], object with optional state ;
源对象 = object with optional state ;

目标对象 = 具有可选状态的对象;
tine process = [participation constraint, " "], process identifier ;

源过程 = process identifier ;
目标过程 = process identifier ;
uniDirNullTag = " relates to "
| " relate to "
| user defined uniDirNullTag ;
forward tag = tag expression ; user
defined uniDirNullTag = tag expression ;

A.4.6.2.3 双向带标签结构句子

双向带标签结构句子 = 非对称双向对象带标签结构句子
| 非对称双向过程带标签结构句子
| 对称双向对象带标签结构句子
| 对称双向过程带标签结构句子; (* 见 9.2.3 和 10.2 *)

非对称双向对象带标签结构句子 = ([participation constraint, " "],
source object, bidir forward tag, [participation constraint, " "],
destination object, [expression constraint])
| ([participation constraint, " "], destination object, bidir backward tag, [participation
constraint, " "], source object, [expression constraint]) ;
非对称双向过程带标签结构句子 = ([participation constraint, " "],
source process, bidir forward tag, [participation constraint, " "], destination process)
| ([participation constraint, " "], destination process, bidir backward tag, [participation
constraint, " "], source process) ;
非对称双向对象带标签结构句子 = ([participation constraint, " "],
source object, "and", [participation constraint, " "], destination object,
" are ", biDirNullTag)
| ([participation constraint, " "], source object,
" and ", [participation constraint, " "], destination object), " are ", symmetric tag ;

symmetric 双向过程带标签结构句子 = ([participation constraint, " "],
source process, " and ", [participation constraint, " "], destination process,
" are ", biDirNullTag)
| ([participation constraint, " "], source process,
" and ", [participation constraint, " "], destination process), " are ", symmetric tag ;

非对称标签 = 标签表达;
双向前推标签 = 标签表达 ;
双向倒退标签 = 标签表达 ; 双向
空标签 = " related"
| user defined biDirNullTag ;

user defined biDirNullTag = tag expression ;

A.4.6.3 OPL基本结构

A.4.6.3.1 聚合句子

聚合句子 = 对象分叉聚合句子 object forked aggregation sentence
| 过程分叉聚合句子; (* 见 9.3.2 *)
对象对象分叉聚合句子= 整个对象, " consists of ", object parts list ;
process forked aggregation sentence = whole process, " consists of ", process parts list ;
object parts list = part object
| (part object, [{ ", ", part object } , " and ", (part object | " at least one other part")]) ;
process parts list = part process
| (part process, [{ ", ", part process }, " and ", (part process | " at least one other part")]) ;
whole object = object identifier ;
part object = [participation constraint, " "], object identifier ;

整个过程 = process identifier ;
part process = [participation constraint, " "], process identifier ;

A.4.6.3.2 特化句子

特化句子 = 对象分叉特化句子
| 过程分叉特化句子 ; (* 见 9.3.3 *)

对象分叉特化句子 = 基本对象分叉特化句子
| 局部对象分叉特化句子
| AsWellAs 对象分叉特化句子
| 局部 AsWellAs 对象分叉特化句子 ;
基本对象分叉特化句子 = object identifier, " exhibits ",
(attribute list | operator list) ;
局部对象分叉特化句子 = object identifier, " exhibits ",
((attribute list, ", and at least one other attribute ")
| (operator list, ", and at least one other operator")) ;
AsWellAs 对象分叉特性化句子 = object identifier,
" exhibits ", attribute list, ", as well as ", operator list ;
partial AsWellAs 基本对象分叉特化句子 = object identifier,
" exhibits ", attribute list, ", and at least one other attribute", ", as well as ", operator list,
", and at least one other operator" ;

属性 = object identifier ;
操作员 = process identifier ;
属性列表 = object list ;
操作员列表 = process list ;

过程分叉特化句子 = 基本过程分叉特化句子 basic process forked characterization sentence
| 局部过程分叉特性化句子 partial process forked characterization sentence
| 局部 AsWellAs 过程分叉特性化句子
| AsWellAs 过程分叉特性化句子 ;
基本过程分叉特化句子 = process identifier, " exhibits ",
(operator list | attribute list) ;
局部过程分叉特化句子 = process identifier, " exhibits ",
((operator list, ", and at least one other operator ")
| (attribute list, ", and at least one other attribute")) ;

AsWellAs 过程分叉特化句子 = process identifier, " exhibits ", operator list, ",
as well as ", attribute list ;
局部 AsWellAs 过程分叉特化句子 = process identifier, " exhibits ",
operator list, ", and at least one other operator", ", as well as ", attribute list,
", and at least one other attribute" ;

A.4.6.4 展示句子

展示句子 = 对象展示句子
| 对象展示句子 ; (* 见 9.3.3.2.2 and 10.3 *)
对象展示句子 = feature, " of ", object identifier, (range clause | " is ",
((attribute list | operator list) | (attribute list, " as well as ", operator list))) ;
对象展示句子 = feature, " of " , process identifier, " is ", ((operator list | object list)
| (operator list, " as well as ", attribute list)) ;

特性 = attribute | operator ;

A.4.6.5 特化句子

特化句子 = 对象特化句子
| 过程特化句子
| 状态特化句子; (* 见 9.3.4 *)

对象特化句子 = 基本对象特化句子
| 多个对象特化句子
| 局部对象特化句子
| Xor 对象特化句子
| 多个对象继承特化句子 ;

基本对象特化句子 = special object, " is a ", general object ;
multiple object specialization sentence = special object list, " are ", general object ;
partial object specialization sentence = special object list, " and other specializations are ", general object ;
Xor 对象特化句子 = basic Xor object specialization sentence

| comma separated Xor object specialization sentence ;
基本 Xor 对象特化句子 = special object, " can be either ", general object, " or ", general object ;
逗号分隔的 Xor 对象特化句子 = special object, " can be one of ", general object,
{ ", ", general object }, " or ", general object ;
multiple object inheritance specialization sentence = special object, " is ", general object list ;

一般对象 = object identifier ;
特定对象 = object identifier ;
一般对象列表 = " a ", object identifier, [{ " a ", object identifier }], " and a ", object identifier ;
special object list = object list ;

过程特化句子 =基本过程特化句子
| 多个过程特化句子
| 局部过程特化句子
| Xor 过程特化句子
| 多个过程继承特化句子 ;
基本过程特化句子 = special process, " is ", general process ;
multiple process specialization sentence = special process list, " are ", general process ;
partial process specialization sentence = special process list, " and other specializations are ",
general process ;
Xor 过程特化句子 = basic Xor process specialization sentence
| comma separated Xor process specialization sentence ;
基本 Xor 过程特化句子 = special process, " can be either ", general process, " or ",
general process ;
逗号分隔的 Xor 过程特化句子 = special process, " can be one of ",
general process, { ", ", general process }, " or ", general process ;
多个过程继承特化句子 = special process, " is ", general process list ;

一般过程 = process identifier ;
特殊过程 = process identifier ;
一般过程列表 = " a", process identifier, [{ " a ", process identifier }] " and a ", process identifier ;
特殊过程列表 = process list ;

状态特化句子 = 基本状态特化句子
| 多 种 状 态 特 化 句 子
| 局部状态特化句子 ;
基本状态特化句子 = state specified object, " is a ", state specified object ;
多种状态特化句子 = state specified object list, " are ", state specified object ;
局部状态特化句子 = state specified object list, " and other specializations are ",
state specified object ;
状态指定句子= state identifier, " ", object identifier ;
状态指定对象列表 = state specified object
| state specified object, [{ ", ", state specified object }], " and ", state specified object ;

A.4.6.6 实例化句子

实例化句子 = 对象实例化句子
| 过程实例化句子; (* 见 9.3.5 *)

对象实例化句子 = 基本对象实例化句子
| multiple object instantiation sentence ;

basic object instantiation sentence= instance object, " is an instance of ", object class ;
多重对象实例化句子 = instance object list, " are instances of ", object class ;

过程实例化句子 = 基本过程实例化句子
| 多个过程实例化句子;
基本过程实例化句子 = instance process, " is an instance of ", process class ;
multiple process instantiation sentence = instance process list, " are an instance of ", process class ;

实例化对象 = 对象标识符;
实例化过程 = 过程标识符 ;
对象类 = 对象标识符 ;
过程类 = 过程标识符 ;
实例对象列表 = 对象列表 ;
实例过程列表 = 过程列表;
(* EndRegion: 结构句子 *)

A.4.7 OPL 上下文管理

A.4.7.1 上下文管理句子

(* 区域: 上下文管理句子 - 该区域定义了所有管理 ODP 上下文变化的句子*)

上下文管理句子 = 展开句子
| 展开句子
| 放大句子
| 缩小句子; (* 见 13.2.1 *)

(* 图表中对象和过程的展开等同于相应的结构句子*)

A.4.7.2 展开句子

展开句子 = 对象展开句子
| 过程展开句子 ;
对象展开句子 = 未指定对象展开句子
| 整个对象展开句子
| 一般对象展开句子
| 类对象展开句子

| 展示者对象展开句子 ;

未指定对象展开句子 = object identifier, " unfolds into ", attribute list,
[" as well as ", operator list] ;
整个对象展开句子 = whole object, " from ", parent OPD, " part-unfolds in ",
child OPD, " into ", object parts list ;
一般对象展开句子 = general object, " from ", parent OPD, " specialization-unfolds in ",
child OPD, " into ", special object list ;
类对象展开句子 = object class, " from ", parent OPD, " instance-unfolds in ",
child OPD, " into ", instance object list ;
展示者对象展开句子 = object identifier, " from ", parent OPD, " feature-unfolds in ",
child OPD, " into ", attribute list, [" as well as ", operator list] ;
过程展开句子 = 未指定过程展开句子
| 整个过程展开句子
| 一般过程展开句子
| 类过程展开句子
| 展示者过程展开句子 ;
未指定过程展开句子 = process identifier, " unfolds into ", operator list,
[", as well as ", attribute list] ;
整个过程展开句子 = whole process, " from ", parent OPD, " part-unfolds in ",
child OPD, " into ", process parts list ;
一般过程展开句子 = general process, " from ", parent OPD, " specialization-unfolds in ",
child OPD, " into ", special process list ;
class process unfolding sentence = process class, " from ", parent OPD, "
instance-unfolds in ", child OPD, " into ", instance process list ;
展示者过程展开句子 = process identifier, " from ", parent OPD,
" feature-unfolds in ", child OPD, " into ", operator list, [" as well as ", attribute list] ;

A.4.7.3 折叠句子

折叠句子 = 对象折叠句子

|过程折叠句子 ;

(* 一个折叠句子仅对一个 OPD 对象或过程相关，展开会为此而产生一个子 OPD，并且是相当于图形粗体轮廓指派的 OPL *)

对象折叠句子 = object identifier, " is folding of ", child OPD ;
过程折叠句子 = process identifier, " is folding of ", child OPD;

A.4.7.4 放大句子

放大句子 = 过程放大句子
| 对象放大句子 ;

过程放大句子 = 图中过程放大句子
| 新图过程放大句子 ;

图中过程放大句子 = (process identifier, " zooms into ", process list,
" in that sequence", [", as well as ", object in zoom list])
| (process identifier, " zooms into parallel ", process list, [", as well as ", object in zoom list])
| (process identifier, " zooms into ", process list, " and parallel ", process list,
", in that sequence", [", as well as ", object in zoom list]) ;
新图过程放大句子 = (process identifier, " from ", parent OPD, " zooms in ",
child OPD, " into ", process list, " in that sequence", [", as well as ", object in zoom list])
| (process identifier, " from ", parent OPD, " zooms in ", child OPD, " into parallel ",
process list, [", as well as ", object in zoom list])
| (process identifier, " from ", parent OPD, " zooms in ", child OPD, " into ", process list,
" and parallel ", process list, ", in that sequence", [", as well as ", object in zoom list]) ;

对象放大句子 = in diagram object in zoom sentence
| 新图对象放大句子 ;

图中对象放大句子 = (object identifier, " zooms into ", object list, " in that sequence",
[", as well as ", process in zoom list]) ;
新图对象放大句子 = (object identifier, " from ", parent OPD, " zooms in ",
child OPD, " into ", object list, " in that sequence", [", as well as ", process in zoom list]) ;
对象放大列表 = object identifier, [{ ", ", object identifier }, " and ", object identifier, ",
in that sequence"] ;

过程放大列表 = process identifier, [{", ", process identifier }, " and ", process identifier,
", in that sequence"] ;

A.4.7.5 缩小句子

缩小句子 = 过程缩小句子
| 对象缩小句子;

(* 一个缩小句子仅与一个 OPD 过程或对象有关，为此，缩小产生了一个子 OPD，并是与图形粗体轮廓称号等同的 OPL *)

过程缩小句子 = process identifier , " is out zoom from ", child OPD ; object
out Zoom sentence = object identifier, " is out zoom from ", child OPD ;

(* EndRegion: Context management sentences *)

(* EndRegion: OPL document *)

(* EndRegion: OPL EBNF *)

BA

附 录 B
（资料性附录）
OPM 运用指南

B.1 通则

鉴于结构及组成复杂的系统的快速发展，人们对采用直观而形式化的对新系统的设计或现有系统的知识进行记录的标准需求变得越来越明显。这种需求则反过来需要具有一个坚实的基础设施来记录、存储、整理和呈现在此知识上建立的知识积累和创意。

概念建模指的是提供系统相关知识的实践活动。这项活动所产生的结果是一个概念模型。概念建模通常先于数学和物理建模，其首要任务不仅是要了解工程、设计和管理系统，而且还要尽可能地制定完整和条理清晰的标准。建模的必要性最终体现在基于模型的系统工程（MBSE）。

了解物理、生物、人工和社会制度并制定与其相关的标准需要一种能够将这些复杂性以一种连贯和直截了当的方式建模的有理有据、形式化但直观的方法论和语言。相同的建模范式，也就是方法论的核心，应为设计新系统和研究及改进现有系统而服务。该范式应该应用于人工和自然系统，并忠实地表达建模域的物理和信息化事物。对象过程方法（OPM）提供了实现这些愿望的手段。

B.2 OPM原理的事物重要性

主要的系统级过程可与系统模型中的对象同样重要或更为重要。OPM特别明确地指出，一个系统的对象过程方法（OPM）模型的最高级别的过程是系统功能，即体现该系统目的和用途的价值提供过程。因此，一个适合建模的过程需独立于在其发生过程中所涉及到的任何对象特定集。

一个OPM系统模型中的事物T的相对重要性通常与T所处对象过程图（OPD）层次中的最高对象过程图（OPD）成正比。

B.3 新对象过程图（OPD）内涵

一个好的OPD集是可读、易懂及易理解的。以下的经验法则有助于决定何时创建一个新的对象过程图及如何尽可能地保持对象过程图的易读易懂性：

——对象过程图（OPD）延伸不应超过一个页面或一个平均大小的显示屏；

——对象过程图（OPD）含有的事物不应超过 20-25 个事物 ；

——事物不应相互遮挡，即，它们或者完全包含在更高级别的事物中，例如在缩放的情况下或没有重叠区域；

——图中不应包含太多关联 - 大致与事物的数量相同；

——一个关联不得跨越已被一个事物占用的区域； 以及，

——跨越彼此的关联数量必须被最小化。

B.4 对象过程方法原理要素表达

出现在一个对象过程图（OPD）的OPM模型要素可作为相同的要素出现在其它任何对象过程图（OPD）中。这个原理可使建模者在OPD中充分发现有助于表达任何数量的任何模型元素（事物或关联）的可能

性。由于一个关联无法在没有其所连接的事物的情况下存在，所以对于一个出现在对象过程图（OPD）中的关联而言，其连接的两个事物也都需要出现。

尽管一个建模者可将任何数量的事物包含在任何对象过程图（OPD）中，但为了清晰度和避免杂乱无章，则往往更需要在一个对象过程图（OPD）中仅包含那些有必要把握系统某一方面或观点的要素。

B.5 多重事物副本约定

为了避免从对象过程图（OPD）的一侧跨越到另一测而引起混乱的这种漫长而曲折的环节，一个对象过程图（OPD）可包含相同事物的多个副本。这种多重事物副本约定补充了对象过程图（OPD）原理的要素表示。正如一个出现在对象过程图（OPD）中的对象过程方法（OPM）模型要素可出现在任何一个对象过程图（OPD）中一样，一个对象过程方法（OPM）要素也可再任何对象过程图（OPD）中多次出现。因此，为避免对象过程图（OPD）由于漫长及纵横交错而引起的混乱，一个事物可使用一个较短连接出现于相同对象过程图（OPD）中的另外一个地方。为便于辨别重复的出现，建模者可通过一个相应的复制事物符号来替代事物符号 - 一个如图B.1所示的略微显示在重复事物背后的小型对象或过程。然而，建模者必须慎用此替代法，因为它需要引起模型读者的注意并要记住这种较长的连接并没有清晰地出现在目前的对象过程图（OPD）的上下文中。

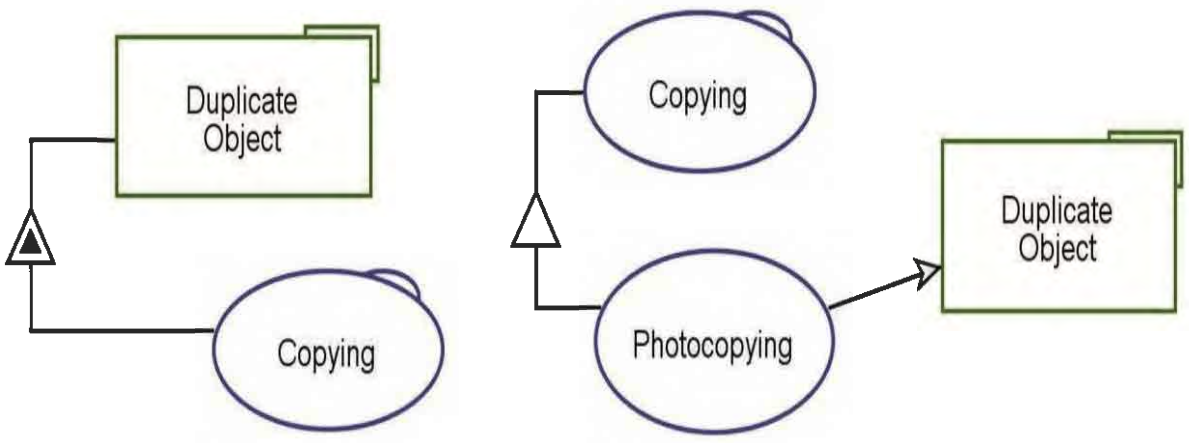


图 B.1 重复对象和重复过程符号

B.6 命名指南

B.6.1 选择名称的重要性

为对象过程方法的模型要素，即对象、过程和关联来选择恰当的标签名称是尤为重要的，因为标签会影响到目标观众对于模型和相应对象过程语言（OPL）的逻辑连贯性和意义领会的交流和理解的难易程度。

B.6.2 对象命名

一个用于对象的名称必须是单数，要将复数名称转换为单数形式。转换一个带有几个成员的对象的推荐方法是在单数形式后加上“集”（通常是无生命物体）或 “组”（通常用于人类）这个单词。

示例1：“配料”（比方说，一个蛋糕的）变为“配料集”，而 “客户” 变成“客户组”。

由于对象名称需要在系统模型中保持独特性，建模者可将一个可细化物的名称用作一个前缀用于其细化名称，或可将可细化物的名称作为一个细化物名称之后以 “的”打头的后缀来使用。这两种命名方案的任何一种在涉及到具有相同语义的细化物时都允许背景识别。

对象名称可以是拥有多个单词的词组，如苹果蛋糕或汽车碰撞。

示例2：如果一个建模者希望将尺寸大小既作为时钟集又作为手表集的一个属性，那么为了区分这两种尺寸，前者可以是时钟集尺寸 （Clock Set Size），后者为手表集尺寸（Watch Set Size），或者，前者可以是时钟集尺寸（Size of Clock Set），后者为手表集尺寸（Size of Watch Set） 。

注1：实施一个 OPM 可以在尝试将作为一个细化物的对象包含到多个上下文中时通知建模者， 这样建模者可以决定内涵的恰当性。

注2：一项实施可以建立一个默认语法以解决细化物的名称问题。

B.6.3 过程命名

一个过程名称是一个短句，最后一个字必须是一个动词的动名词形式，即带有 “ing” 的后缀动词。如果有多项选择的话，如建筑（Construction ）与施工 （Constructing），则可选择后者。

过程命名存在以下的变量：

—— 动词版本，就是动词的动名词形式，即动词+ ing，如制作（Making）或响应 (Responding)；

——名词-动词版本，是一个名词（一个 OPM 对象）与动名词的级联，即名词+动词+ ing，如蛋糕制作 Cake Making 或碰撞响应 ；

—— 形容词-动词版本，是一个带有动词的动名词形式的一个形容词的级联 ，即形容词+动词+ ing，如快速制作（Quick Making）或自动响应 Automated Responding ； 以及

—— 形容词-名词-动词版本，是一个与带有动名词的名词的形容词的级联，也就是形容词+名词+动词+ ing，如快速蛋糕制作（Quick Cake Making ）或自动碰撞响应（Automatic Crash Responding）。

在后一种情况下，该形容词修饰了过程（动名词，是一个名词）。但是，形容词也可修饰对象（名词），如甜蛋糕制作（Sweet Cake Making）或致命碰撞响应（Fatal Crash Responding）。

功能的名称以及所有 OPM 过程的名称应包括以一个带有动名词的动词形式结尾的不超过四个的大写子母，例如：大城市人口保障（Large City Population Securing）。

由于过程名称必须具有独特性，建模者可将一个细化物的名称用作一个细化物名称之后以“的”开头的后缀。命名方案允许在涉及到具有类似语义的细化物时可对上下文进行区分。

B.6.4 状态命名

状态名称必须反映不同的相关情况，它们其中所 “拥有”的对象可在任何特定时间点发生。比较可取的状态名称是所拥有对象的被动式，而非动名词形式。

示例：如果一个产品被涂漆，然后被检查，那么它的状态就是被喷涂（painted）和被检查（inspected），而非涂漆（painting）和检查（inspecting）。喷涂是将产品从其未被喷涂变为被喷涂状态的过程，而检查（inspecting）是

将产品从被喷涂状态变为其被检查（inspected）状态。尽管产品的喷涂出现，只要喷涂一发生，它就已经离开了其未被喷涂的状态，并处于状态之间的过渡中，尚未进入其被喷涂状态，直至喷涂完成。

B.6.5 大写约定

在OPM中，一个事物（对象或过程）名称中的每个单词的第一个字母大写，而一个对象状态或关联名称不大写。这项约定有助于产生更加易读的对象过程语言（OPL）句子。

CB

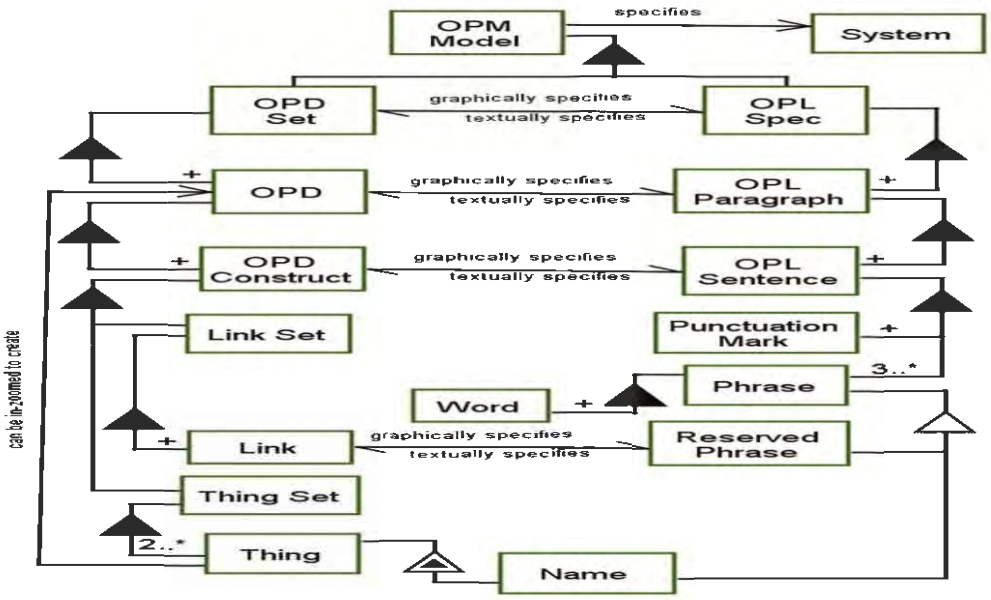
附 录 C
（资料性附录）
使用对象过程方法（OPM）建立 OPM 模型

C.1 对象过程方法（OPM）模型

图C.1中的OPD代表了作为OPM模型的 OPM的各个方面。第C.4条阐述了特定要素。第C.5条展示了在展开和放大过程中有关关联处理的一个模型。 第C.6条呈现了一个用于评估过程调用、性能和完成的模型。

这一组条款将OPM表达为一组拥有相应OPL的OPD。建模者为这个展示选择了将模型内容限制为相对简单的OPM用法，即最小复合性关联，且并未尝试将单独OPD统一到一个单一的OPM模式中去。然而，的确出现了一些减少文本冗余性并有助于澄清确实发生的相关模型事实的高级表达方式。

C.2 OPM模型架构



OPM 模型指定了系统。

OPM 模型包含 OPD 集和 OPL 规范.

OPL 规范 包含至少一个 OPL 段落.

OPD 集包含至少一个 OPD.

OPD 集以图形形式指定了 OPL 规范.

OPL 规范以文本形式指定了 OPD 集.

OPD 包含至少一个 OPD 构造。

OPL 段落 包含至少一个 OPL 句子。

OPD 以图形形式指定了 OPL 段落.

OPL 段落以文本形式指定了 OPD.

OPD 以图形形式构造了 OPL 句子.

OPL 句子以文本形式指定了 OPD 构造。

OPD 构造由事物集和关联集组成。

事物集包含两个到多个事物。

连接集包含至少一个连接连接。

事物展现了名称。

OPL 句子 c 包含三个到多个词语和至少一个标点符号。

词句包含至少一个单词。

OPL 保留词句事物名称是词句。

连接已图形形式指定了保留的词句。

保留的词句以文本形式指定了连接。

事物可被放大以创立 OPD。

图 C.1 OPM 模型架构

图C.1是 一个OPD模型的架构模型，其将 OPM的概念方面描述为图形式和文本式 OPM模式的并行层次以及它们产生等效模型表示的相对应性。一个OPD构造是相对应文本OPL句子的图形表达式，其表达了相同模型的事实。一个OPD及其相应的 OPL段落是一个建模者放入到同样模型情景中模型事实的集合体。

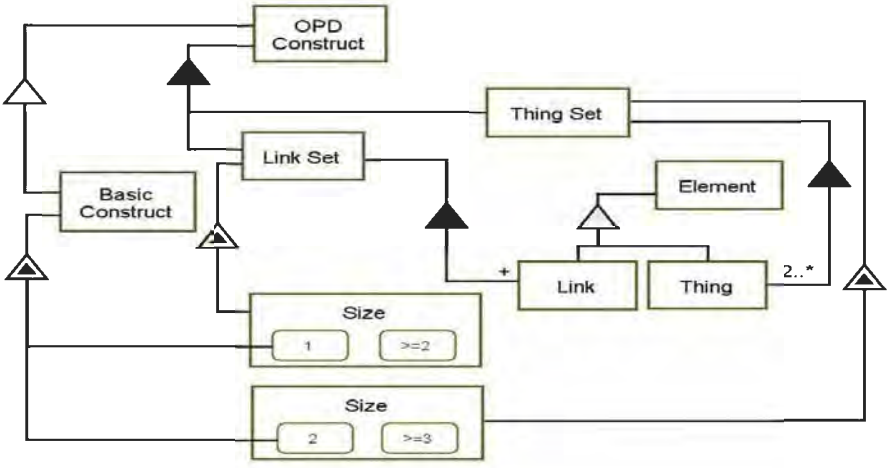
C.3 OPD构造模型

图C.2阐述了OPD的构造概念。该模型的目的在于区分来自于另一种可能的OPD构造的基本构造。一个基本构造是OPD构造的一个特化，其包含仅由一个关联所连接的恰好两个事物。非基本构造除其它外还包括那些具有关联扇出或多于两个的细化物的构造。

示例1：图 C.1 中，两个对象 OPM 模式和 OPD 集一同与从前者到后者的集合-分散关联构成一个基本结构。相当于该基本构造的 OPL 句子是：OPM 模型由 OPD 集 组成 。

示例2：图 C.1 中的三个对象 OPM 模型和 OPD 集和 OPL 规范连同来自于 OPM 模型到 OPD 集和 OPL 规范的集合-分散关联共同构成了一个复合结构。相当于该基本构造的 OPL 句子是：OPM 模型由 OPD 集和 OPL 规范组成。

注：一个对象-状态关联在一个对象与其状态的每一个状态之间是隐式的。从图形上看，该关联表达法通过将状态放入对象矩形内发生，有效地将对象与状态连接在一起。因此，一个带有两个或更多状态的对象是一个OPD构造，而带有一个状态的对象是基本构造。一个无状态对象根本不是一个构造，因为它连一个隐式关联都不具备。



OPD 构造包含事物集和关联集。

事物和关联是元素。

事物集包含两到多个事物。

关联集包含至少一个关联。

事物集展示了事物集的大小。

关联集展示了关联集的大小。

事物集的大小可以是 2 或 >=3。

关联集的大小可以是 1 或 >=2。

基本构建是一个 OPD 构建。

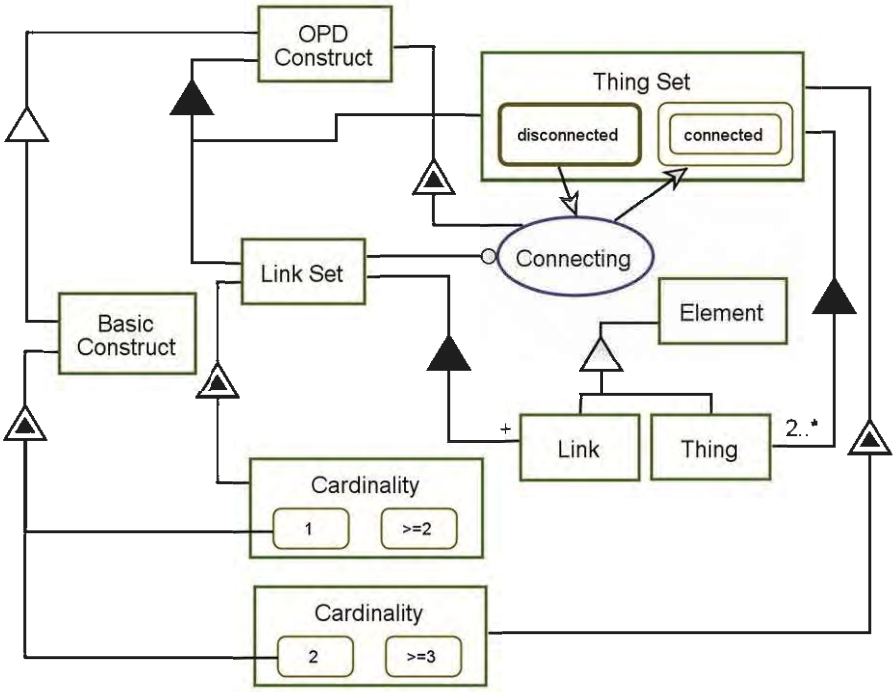
基本构建展示了关联集的 1 个大小。

基本构建展示了关联集的 2 个大小。

图 C.2 OPD 构造和基本构造模型

在某些情况下，两个构造的语法很容易组合到一个如下列OPD构造模型变量中所示的减少了文本内冗余的复合OPL句子中去。

建模者可将一个过程添加到图C.2的模型中去，以表明OPD构造展示了如图C.3中所示的连接性。通过添加事物集的断开和被连接状态，模型的目标行为因此会包括使用关联集作为一个连接手段去将一个被断开的事物集转化到一个已连接的事物集。



OPD 构造包含关联集和事物集。

OPD 构造展示了关联。

关联集包含至少一个关联。

关联集展示了基数。

关联集的基数可为 1 或 >=2。

事物集展示了基数。

事物集包含了 2 个到多个事物。

事物集的基数可为 2 或>=3。

关联和事物 是元素。

连接要求关联集。

连接将事物集从被断开改变为被连接。

事物集被断开的状态为初始。

事物集被连接的状态为最终。

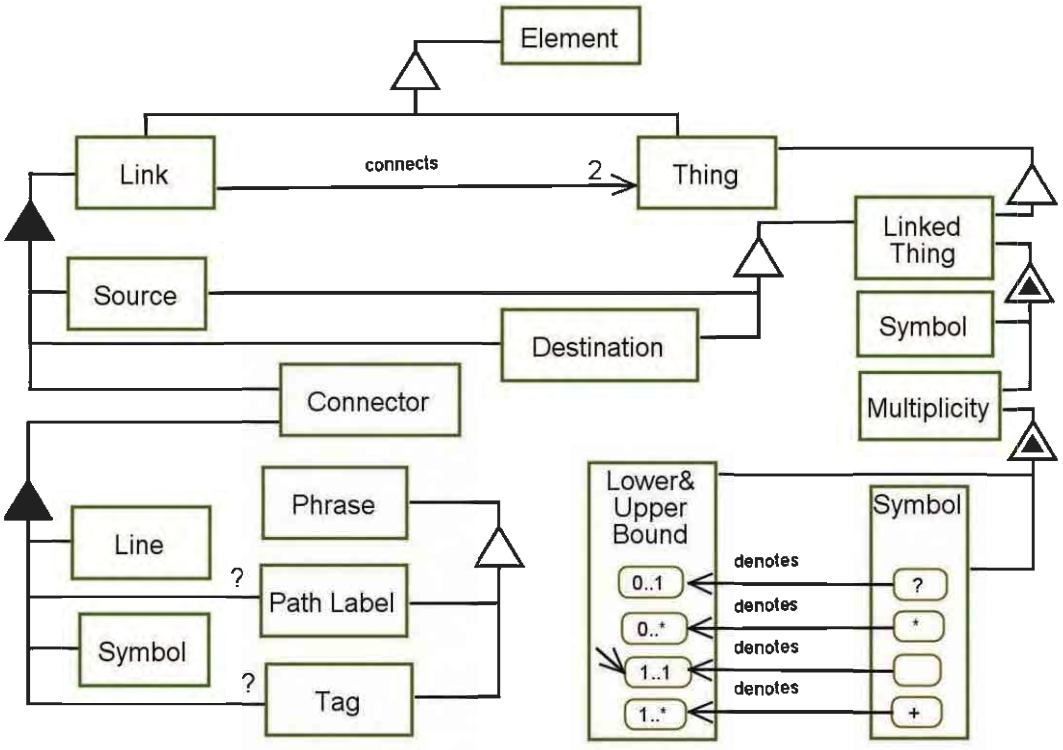
基本构造是一个 OPD 构造。

基本构造展示了连接集 的 1 个基数和事物集的 2 个基数。

图 C.3 OPD 构造和基本构造的构建

C.4 OPM 要素模型

图 C.4中的模型仅适用于基本构造，因为关联连接了2个事物且没有超过两个事物。



事物和关联是要素。

关联将两个事物连接起来。

关联包含源、目的地和连接器。

连接器包含线、符号、一个可选标签和一个可选路径标签。

标签和路径标签是短句。

源和目的地是相关联的事物。

相关联的事物是一个事物。

相关联的事物展示了符号和多重性。

多重性展示了符号和上下界。

上下界可以是 0..1, 0..*, 1..1, 或 1..*。

上下界的默认值为 1..1。

多重性的符号 可以是 ?, *, NONE, 或 +。

多重性的? 符号代表的是 0..1 上下界。

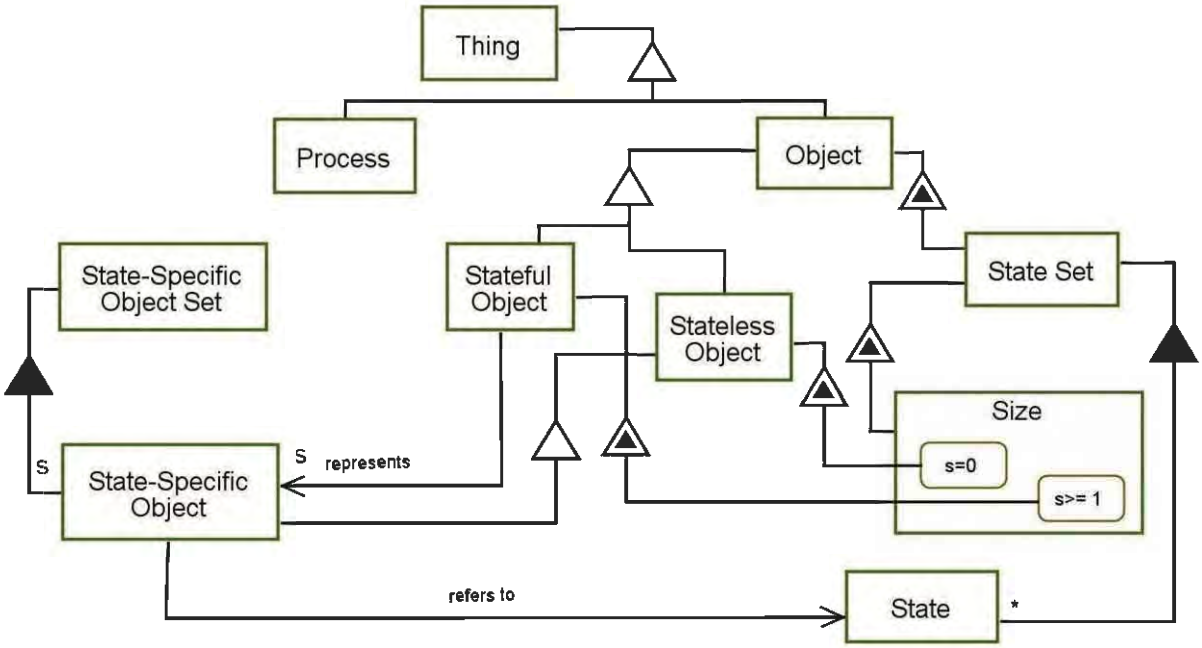
多重性的* 符号代表了 0..* 上下界。

多重性的 NONE 符号代表了 1..1 上下界。

多重性的+ 符号代表了 1..*上下界。

图 C.4 OPM 要素的 OPM 模型

图C.5 是一个OPM事物模型，将其特化显示到对象和过程当中。一组状态表现了对象的特征，它可为空状态。即处在一个无状态对象中，而在一个有状态对象的情况下则不为空。一个带有s状态的状态对象会产生一组的s无状态的状态-指定对象，每个对象针对一个状态。一个特定状态-指定对象指的是在一个特定状态中的对象。将状态-指定对象的概念即作为一个对象又作为一个状态来进行建模能使我们通过引用一个对象和其任何状态仅仅是以指定对象的方式来简化概念模型。



过程和对象是事物。

对象展示了状态集。

状态集展示了尺寸。

状态集基数 A 可以是 s=0 或 s>= 1。

状态集包含可选择 S 状态。

目前状态是一种状态。

无状态对象和状态对象是对象。

无状态对象展示了状态集的 s= 0 Size 。

状态对象展示了状态集的 s>= 1 Size 。

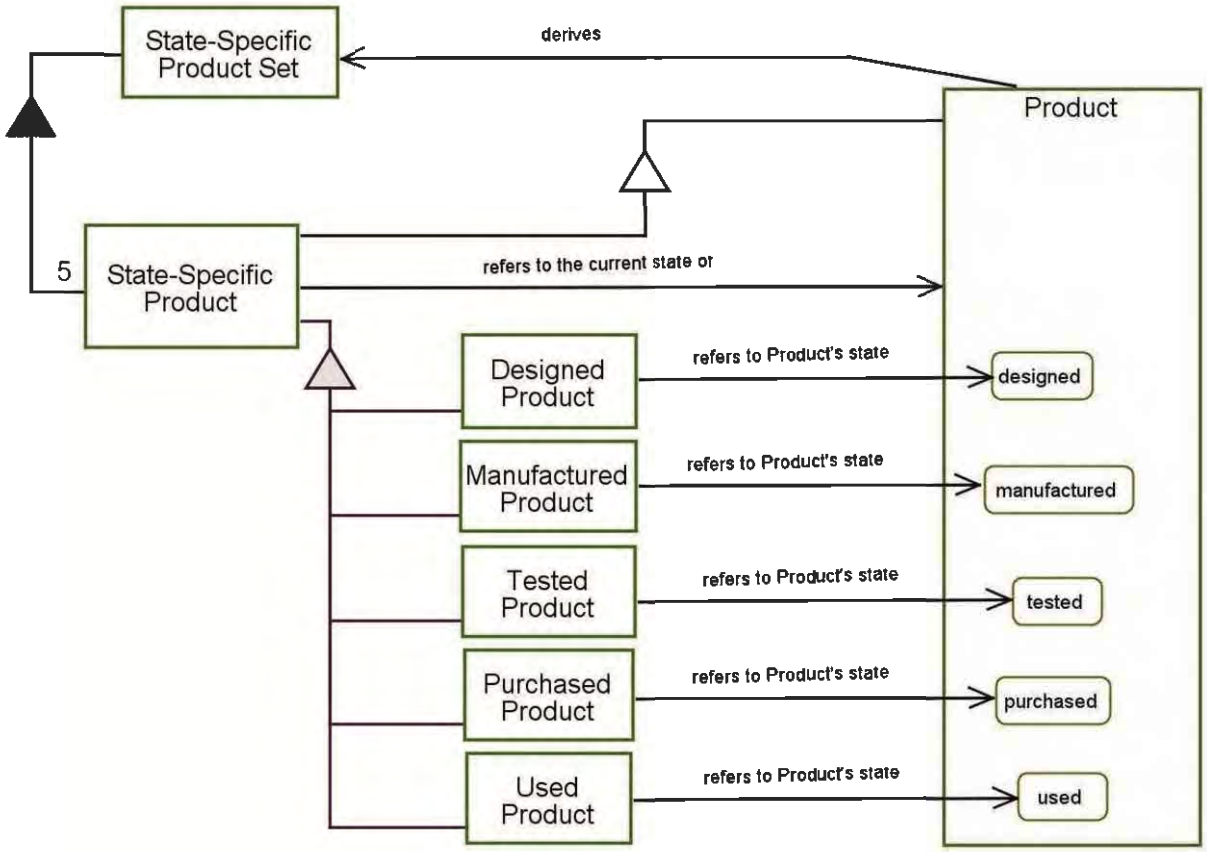
状态对象代表 s 状态-指定对象。

状态-特定对象包含 s 状态-指定对象。

状态-特定对象指的是状态。

图 C.5 事物的 OPM 模型

示例：图 C.6 中的产品是一个具有 5 种状态的状态对象，产品的五个独特的特化就是从中导出的，每一个特化指的是产品的一个独特状态。因此，状态-指定产品也被称为测试产品，指的是产品测试这种状态。当然，相同的对象，测试产品，指的也是产品本身，因为是一种状态； 如果不提及所处状态中的对象的话，“测试”则没有任何意义。这样来说，就有五个状态-指定产品，每一个产品都是产品的一个特化，且捕捉产品的一个特定状态。



产品 可以是设计的、 制造的、测试的、购买的或使用的。

产品衍生了状态-指定产品集。

状态-指定产品集 包含 5 个状态-特定产品。

状态-指定产品是一个产品。

状态-指定产品指的是产品的目前状态。

指定设计产品、制造产品、测试产品、购买产品和使用产品是状态-指定产品。

指定设计产品指的是被设计的产品状态。

制造产品指的是被制造的产品状态。

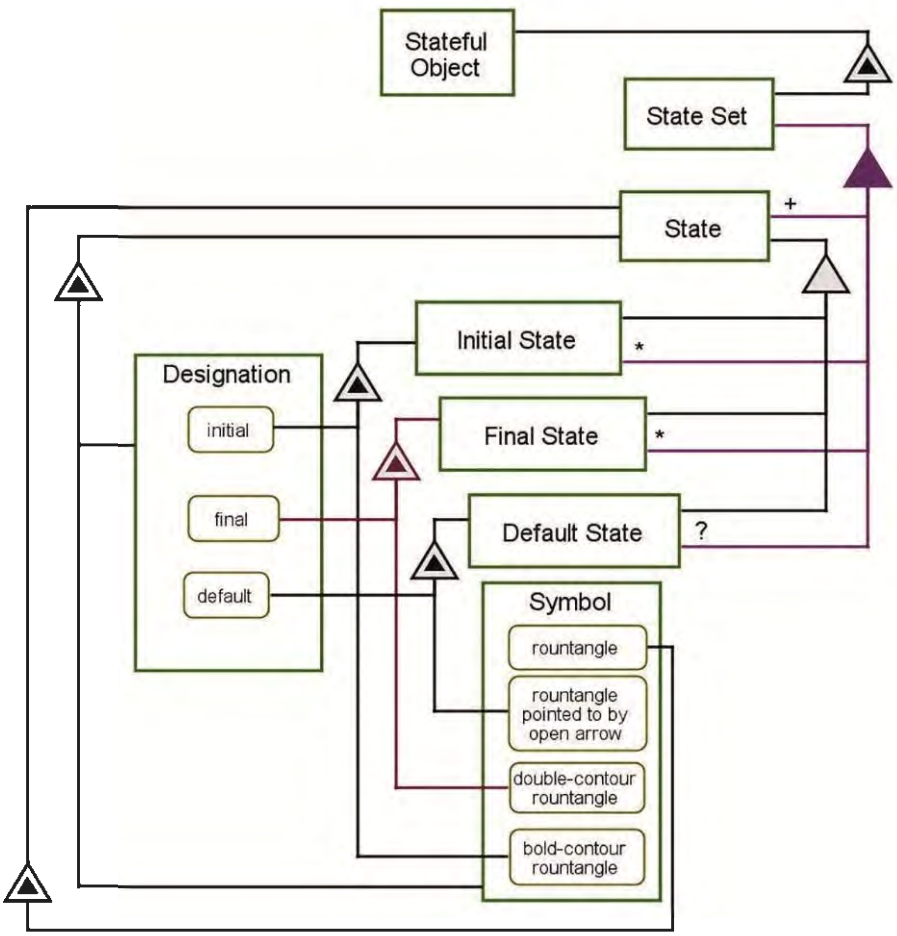
测试产品指的是被测试的产品状态。

购买产品指的是被购买的产品状态。

使用产品指的是被使用的产品状态。

图 C.6 状态-特定对象例子

图C.7 是一个状态对象和状态的OPM模型。



状态对象展示了状态集。

状态集包含至少一个状态，可选择初始状态，可选择 最终状态和可选择默认状态。

状态展示了终点和符号。

指定可以是初始、最终或默认是状态。

初始状态、最终状态和默认状态是状态。

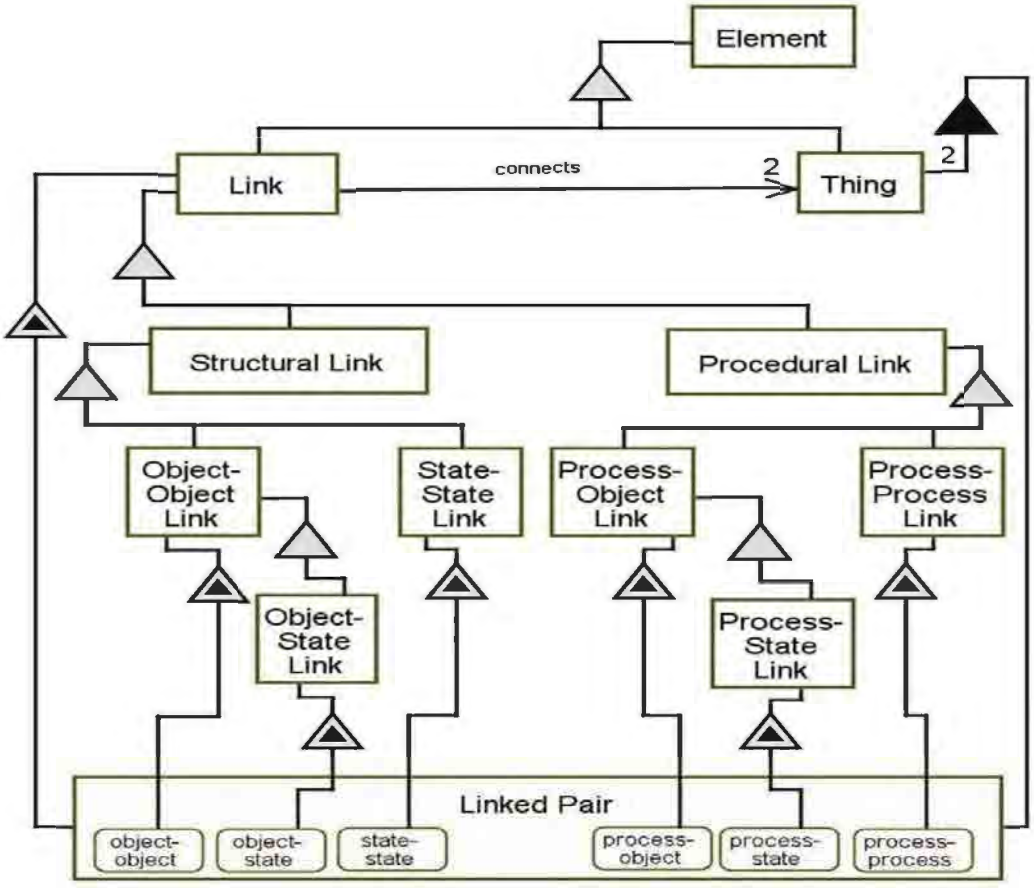
初始状态展示了初始指定和状态的粗体轮廓圆角符号。

最终状态展示了最终指定和状态的双轮廓圆角符号。

默认状态展示了默认指定和状态的开放箭头符号所指向的圆角。

图 C.7 状态对象和状态的 OPM 模型

图C.8中的模型仅对基本构造有效，因为连接将2个事物进行连接 且没有超过两个事物。



事物和关联是元素。

关联将两个事物进行连接。

关联展示了关联对。

关联对包含两个事物。

关联对可以是对象-对象、对象-状态、状态-状态、过程-对象、过程-状态或过程-过程。

结构关联和程序关联是关联。

对象-对象关联和状态-状态关联是结构关联。

对象-状态关联是一个对象-对象关联。

对象-对象关联展示了对象-对象关联对。

对象-状态关联展示了对象-状态关联对。

状态-状态关联展示了状态-状态关联对。

过程-对象关联和过程-过程关联是写程序关联。

过程-状态关联是一个过程-对象关联。

过程-对象关联展示了过程-对象关联对。

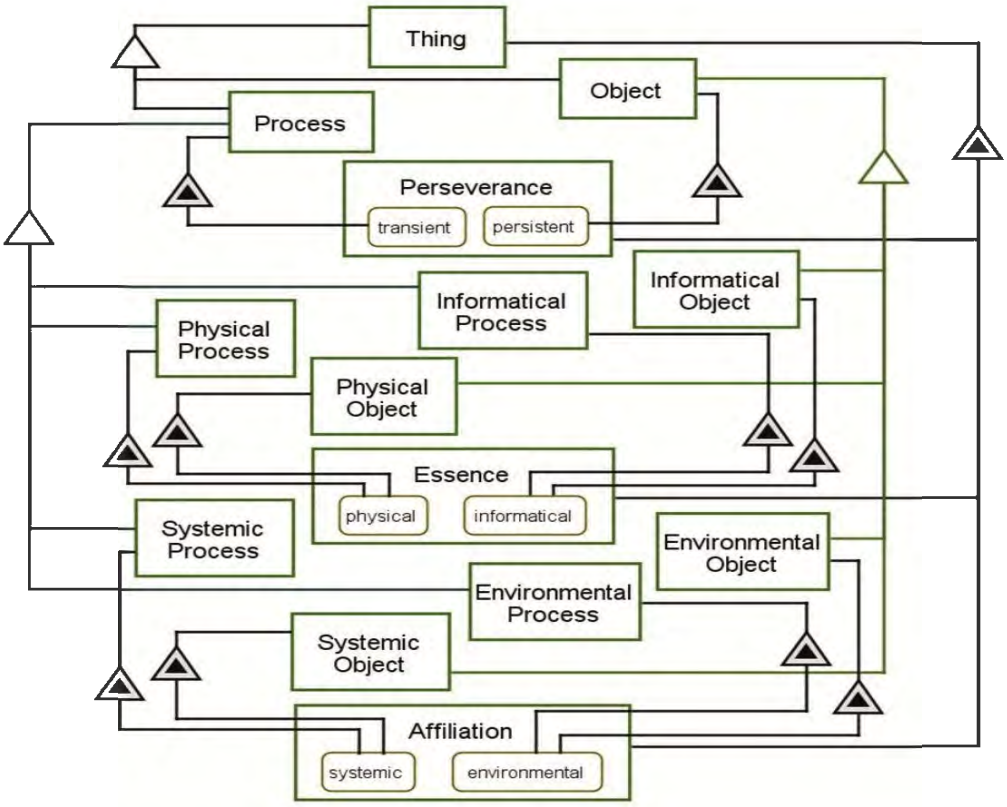
过程-状态关联展示了过程-状态关联对。

过程-过程关联展示了过程-过程关联对。

图 C.8 关联的 OPM 模型

图C.9描绘了事物和其韧性、本质和隶属关系的类属属性被用作为一个展示-表征关联的属性细化物而被建模，韧性是对象与过程之间的鉴别属性。本质一方面是物理对象与物理过程的鉴别属性，而另一方面则是信息对象与信息过程之间的鉴别属性。

隶属关系一方面是系统对象与系统过程之间的鉴别属性，而另一方面则是环境对象和环境过程之间的鉴别属性。



事物展示了韧性、本质和隶属关系。

韧性可以是暂时性的或持久性的。

本质可以是物理的或信息的。

隶属可以是系统性或环境性的。

对象和过程是事物。

过程展示了暂时性的韧性。

对象展示了持久性的韧性。

物理过程、系统过程和环境过程是过程。

物理对象、信息对象、系统对象和环境对象是对象。

物理过程和物理对象展示了物理本质。

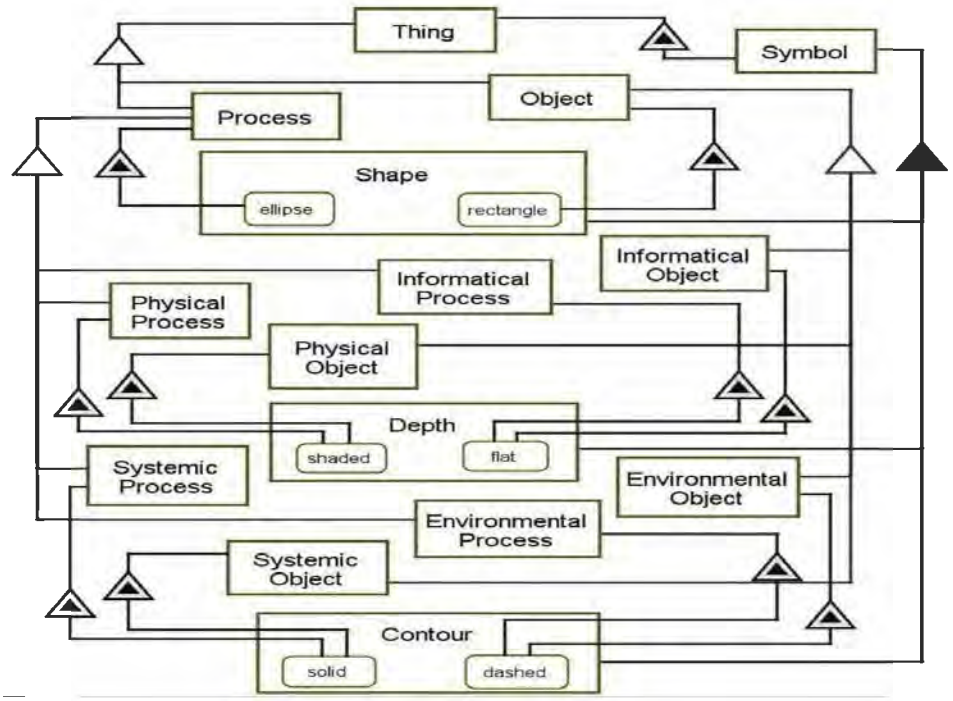
信息过程和信息对象展示了信息本质。

系统过程和系统对象展示了系统隶属关系。

环境过程和环境对象展示了环境隶属关系。

图 C.9 事物类属属性的 OPM 模型

图C.10 描绘了一个OPM事物图形表达的OPM模型，显示了一个符号细化物属性和符号的三个部分：形状、深度和轮廓。形状是使对象与过程之间进行区分的部分。深度一方面是使物理对象与物理过程之间进行区别的部分，而另一方面又是信息对象与信息过程之间进行区分的部分。轮廓一方面是使系统对象与系统过程之间进行区别的部分，而另一方面又是环境对象与环境过程之间进行区分的部分。由于一个对象的状态是与对象绑定的，所以与一个特定状态对象相关联的本质和隶属关系与对象的本质和隶属关系是相同的。



事物展示了符号。

事物符号包含形状、深度和轮廓。

形状可以是椭圆形或长方形。

深度可以是有阴影或非阴影的。

轮廓可以是实线或虚线的。

过程和对象是事物。

过程展示了椭圆形。

对象展示了长方形。

物理过程、信息过程、系统过程和环境过程都是过程。

物理对象、信息对象、系统对象和环境对象都是对象。

物理过程和物理对象展示了有阴影的深度。

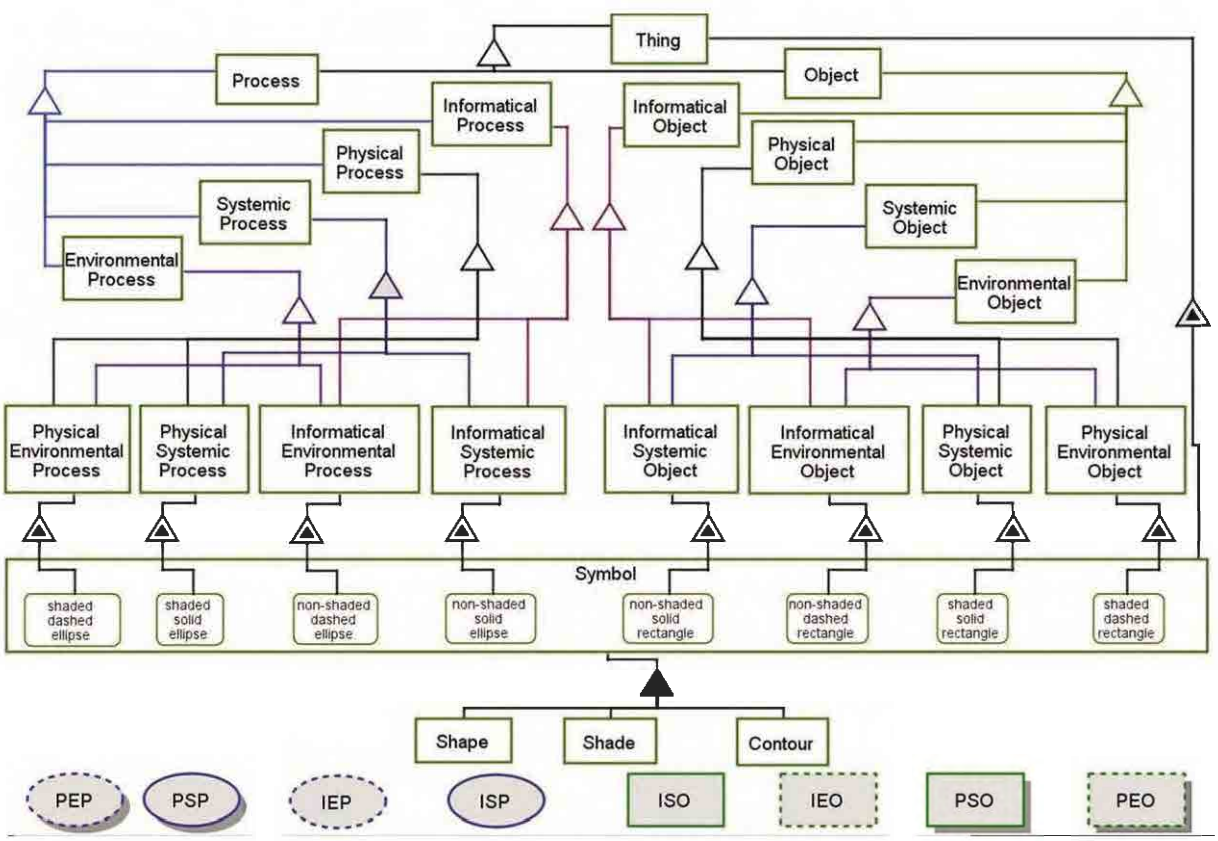
信息过程和信息对象展示了平面深度。

系统过程和系统对象展示了实体轮廓。

环境过程和环境对象展示了虚线轮廓。

图 C.10 事物符号表达式的 OPM 模型

图C.11　是图C.10中模型的变体，其中事物符号属性的三个部分显示为八个值，每个值用于每个可能事物的配置。这里和本附录的其它几个模型图表中的实际符号出现在OPD的底部。在此情况下，该符号处于其各自模型对象和事物的符号值下方。这八个位于OPD底部的符号是用来进行说明的，所以与OPD本身不同。图C.11通过枚举符号的八个状态，也就是符号的深度、轮廓和形状细化物属性值2x2x2的笛卡儿积来增强图C.10的符号细化物。



事物展示了符号。

事物的符号包含深度、轮廓和形状。

事物的符号 可为阴影虚线矩形、阴影实心椭圆、非阴影虚线椭圆、 非阴影实心椭圆、非阴影实体矩形、非阴影虚线矩形、阴影实体矩形或阴影虚线矩形。

对象和过程是事物。

物理过程、信息过程、系统过程和环境过程是过程。

物理对象、信息对象系统对象和环境对象是对象。

物理系统过程是一个物理过程和一个系统过程。

物理系统过程展示了事物的阴影实体椭圆符号。

物理环境过程是一个过程和一个环境过程。

物理环境过程展示了事物的阴影虚线椭圆符号。

信息换进过程是一个信息过程和一个环境过程。

信息环境过程展示了事物的非阴影虚线椭圆符号。

信息系统过程是一个信息过程 和一个系统过程。

信息系统过程展示了事物的非阴影实体椭圆符号。

物理环境对象是一个环境对象。

物理环境对象展示了事物的阴影虚线矩形符号。

物理系统对象是一个物理对象和一个系统对象。

物理系统对象展示了事物的阴影实体矩形符号。

系统环境对象是一个环境对象。

信息环境对象展示了事物的非阴影虚线矩形符号。

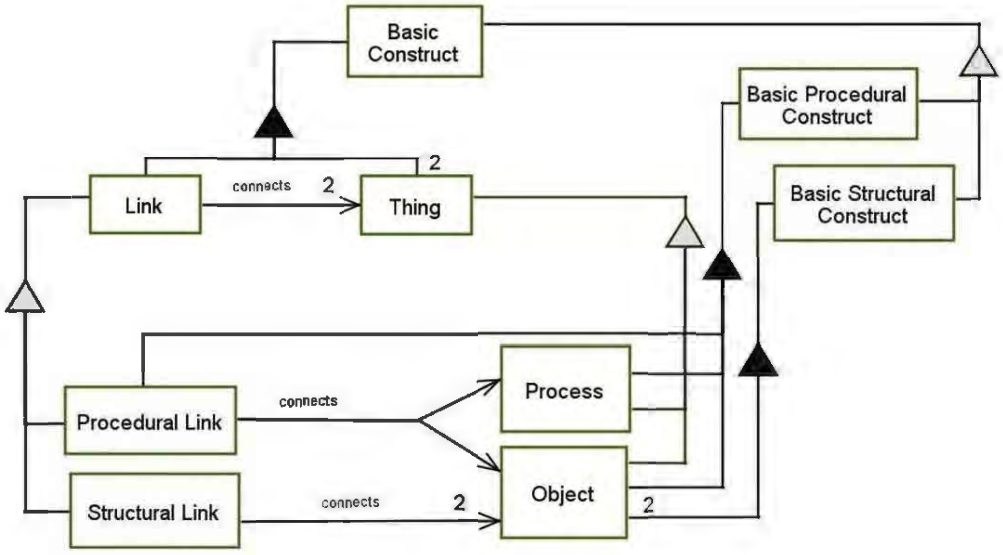
信息系统对象是一个信息对象和一个系统对象。

信息系统对象展示了事物的非阴影实体矩形符号。

事物的系统包含深度、轮廓和形状。

图 C.11 八个事物符号表达式的 OPM 模型

图C.12的模型仅对基本构造有效，因为关联连接了两个事物且没有超过两个事物。



基本构造包含连接和 2 个事物。

连接将两个事物连接起来。

结构连接和程序连接是连接。

基本结构构造和基本程序构造是基本构造。

基本结构构造包含结构连接和 2 个事物。

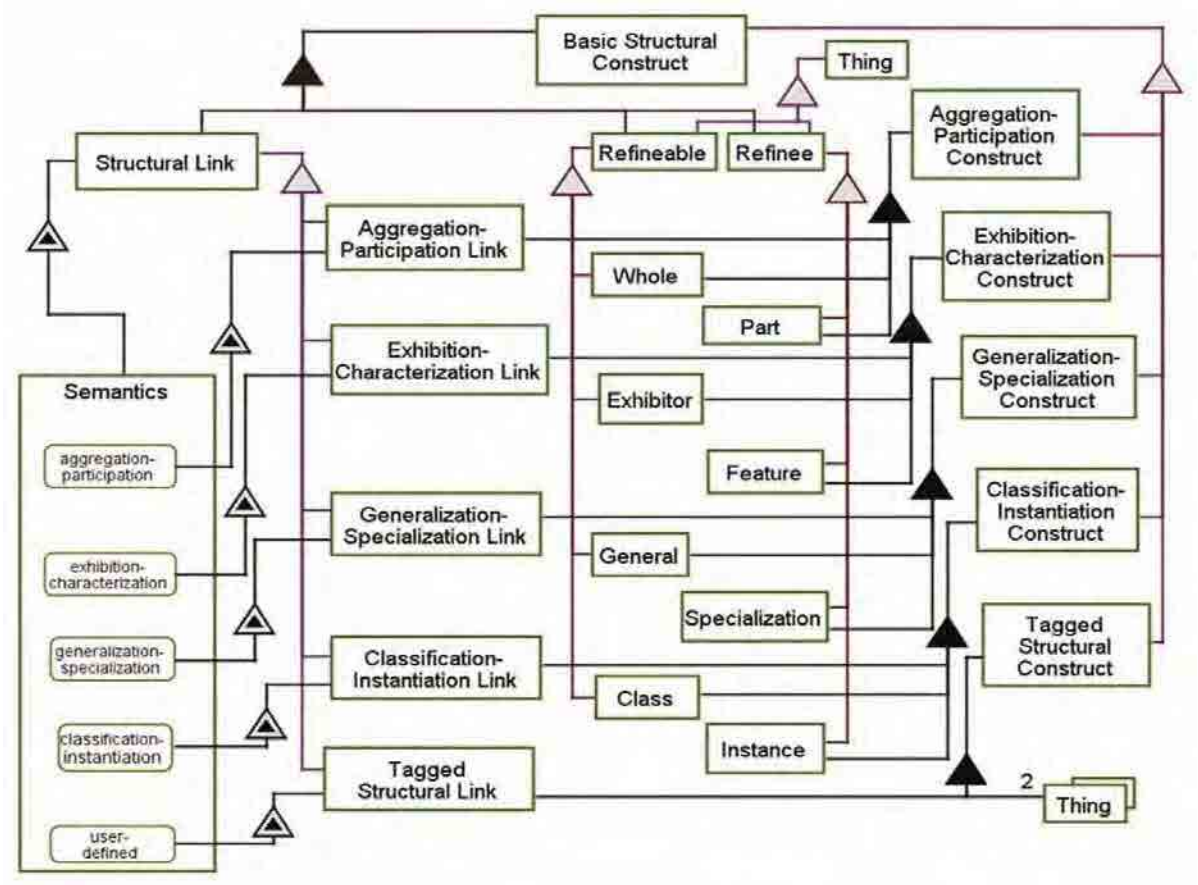
基本程序构造包含程序连接、对象和过程。

结构连接包含 2 个对象。

结构连接将一个过程和一个对象连接起来。

图 C.12 基本构造阐述

图 C.13 是一个基本结构构造的 OPM 模型。



基本结构构造包含可细化物、细化物和结构关联。

可细分物和细化物是事物。

整体、参展物、类属和类都是可细化物。

部分、性能、特化和实例都是细化物。

结构关联展示了语义。

结构关联的语义可以是集合-分散、展示-表征、泛化-特化、类化-实例化或用户定义。

集合-分散关联、展示-特性关联、泛化-特化关联和带标签结构关联是结构关联。

集合-分散关联展示了聚合-分散语义。

展示-表征关联展示了展示-表征语义。

泛化-特化关联展示了泛化-特化语义。

类化-实例化展示了类-实例语义。

标签结构带标签结构关联展示了用户定义语义。

聚合-分散构造、展示-表征构造、泛化-特化构造、类化-实例化构造和带标签结构构造是基本结构构造。

聚合-分散构造包含聚合-分散关联，整体和部分 。

展示-表征构造包含展示-表征关联、展示物和特征。

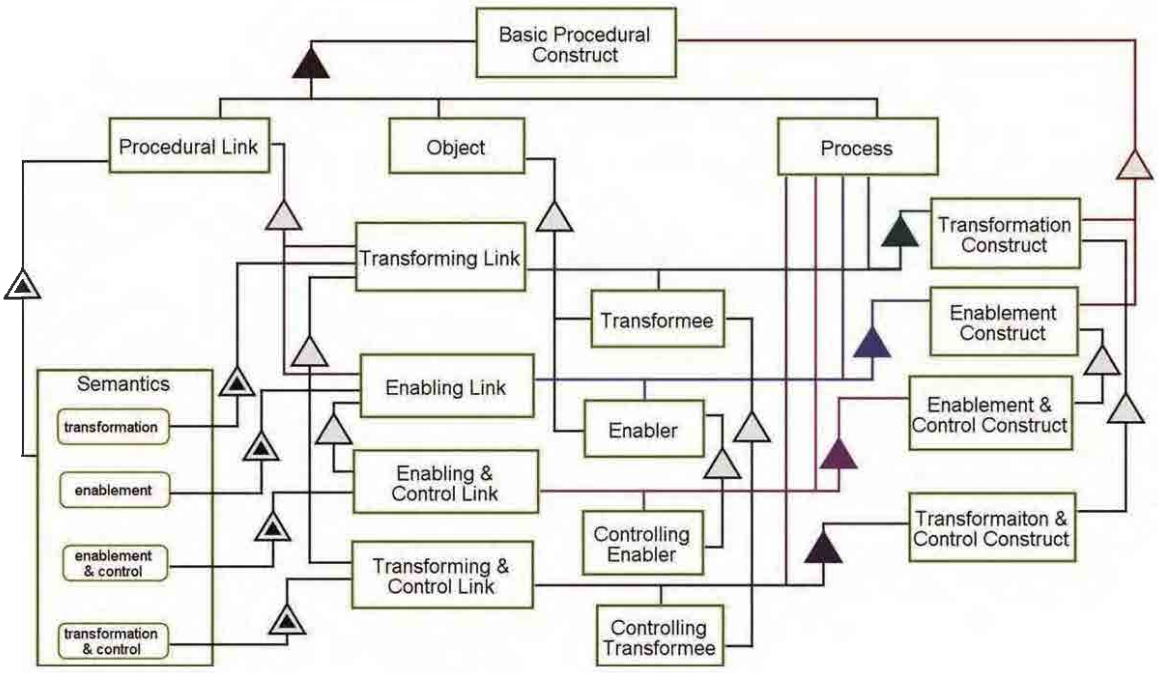
泛化-特化构造包含泛化-特化关联、一般类和特殊类。

类化-实例化构造包含类化-实例化关联、类和实例。

标签结构带标签结构构造包含带标签结构关联和两个事物。

图 C.13 基本结构构造的 OPM 模型

图C.14 是一个基本程序构造的OPM模型。



基本程序构造包含对象、过程和程序关联。

程序关联展示了语义。

程序关联的语义可以是转换、启用、转换和控制以及启用和控制。

转换物和启用者是对象。

控制被转换物是一个被转换物。

控制使能器是一个使能器。

转换关联和启用连接是程序关联。

转换和控制关联是一个转换关联。

使能和控制关联是一个使能关联。

转换关联展示了程序关联的转换语义。

启用关联使能关联展示了程序关联的使能语义。

转换和控制关联展示了程序关联的转换和控制语义。

使能和控制关联展示了程序连接的使能和控制语义。

转换构造和启用构造是基本程序构造。

转换构造包含转换关联、转换物和过程。

使能构造包含使能关联、使能器和过程。

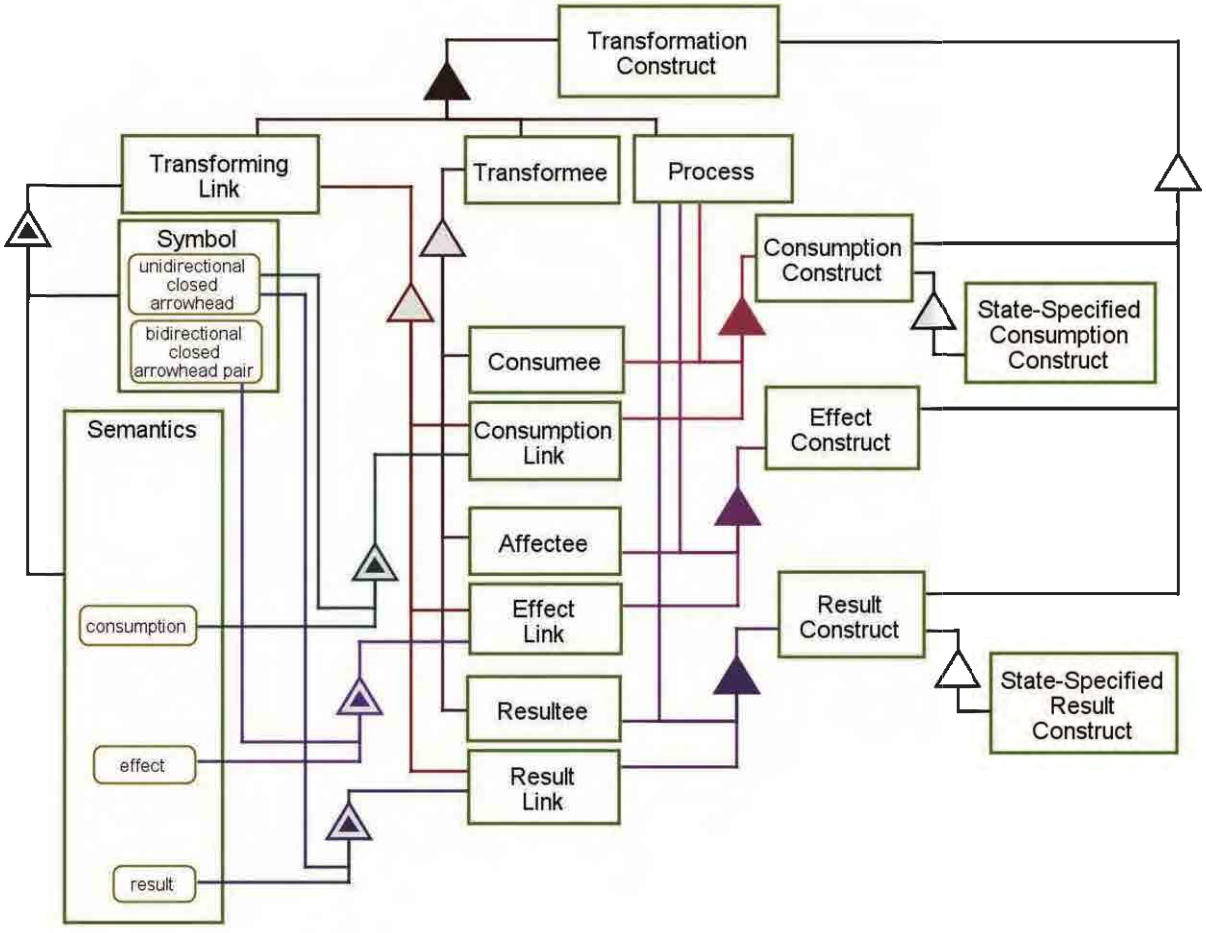
转换和控制构造是一个转换构造。

使能和控制构造是一个使能构造。

转换和控制构造包含转换和控制连接、控制转换物和过程。

使能和控制构造包含使能和控制连接、控制使能器和过程。

图 C.14 基本程序构造的 OPM 模型



转换构造包含转换物、过程和转换连接。

转换关联展示了符号和语义。

转换关联符号可以是单向封闭箭头或双向封闭箭头对。

转换关联语义可以是消耗、效果或结果。

消耗关联、效果关联和结果关联是转换关联。

被消耗物、受影响物和结果物是被转换物。

消耗构造、结果构造和效果构造是转换构造。

消耗构造包含消耗关联、过程和消耗物。

效果构造包含效果关联、过程和受影响物。

效果构造包含结果关联、过程和结果物。

消耗关联展示了转换关联的单向封闭箭头符号和转换关联的消耗语义。

效果关联展示了转换关联的双向封闭箭头消耗对和转换关联的效果语义。

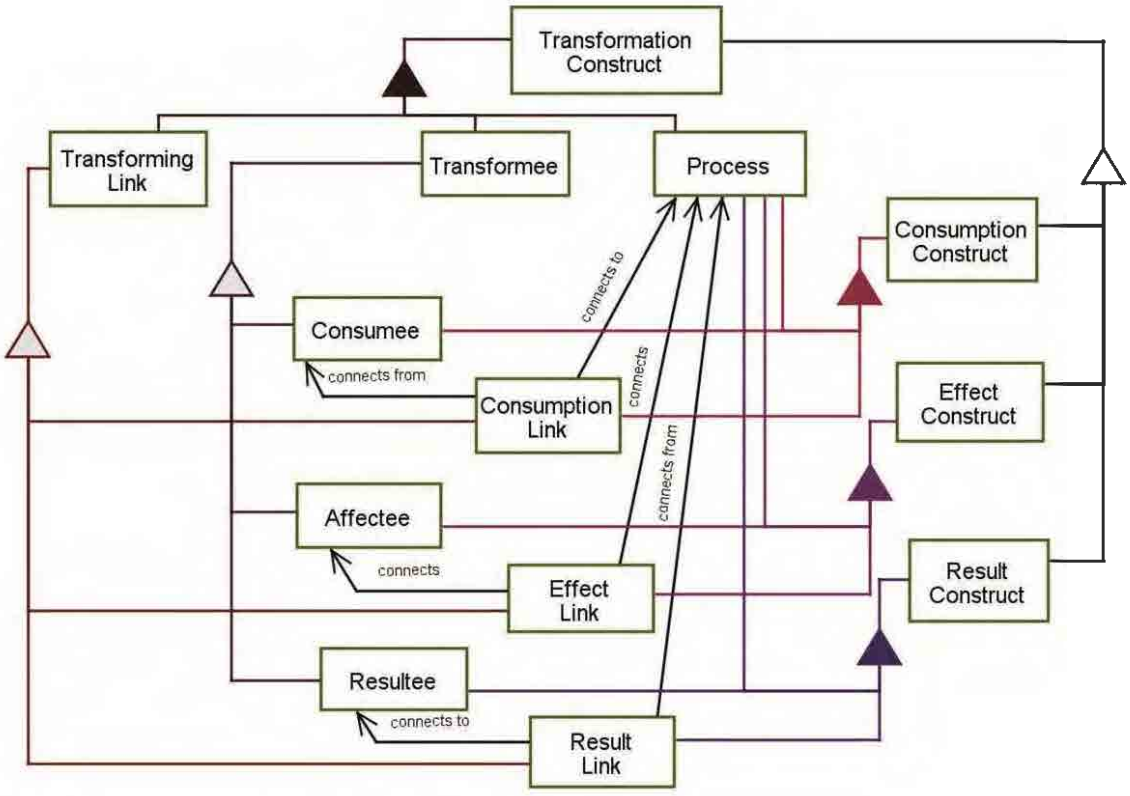
结果关联展示了转换关联的单向封闭箭头符号和转换关联的结果语义。

状态-指定消耗是消耗构造。

状态-指定结果构造是结果构造。

图 C.15 转换构造的 OPM 模型

图C.16 通过增加有关将一个对象与过程连接的箭状符号头符号的有向信息对图C.15进行了补充。将该信息添加到图C.15可能会将模型变得杂乱无章，从而使之变得更难以理解。



转换构造包含转换物、过程和转换关联。

消耗关联、效果关联和结果关联是转换关联。

消耗构造、结果构造和转换构造是转换构造。

消耗构造包含消耗关联、过程和被消耗物。

效果构造包含效果关联、过程和受影响物。

结果构造包含结构关联、过程和结果物。

消耗关联从消费物进行连接。

消耗关联连接到过程。

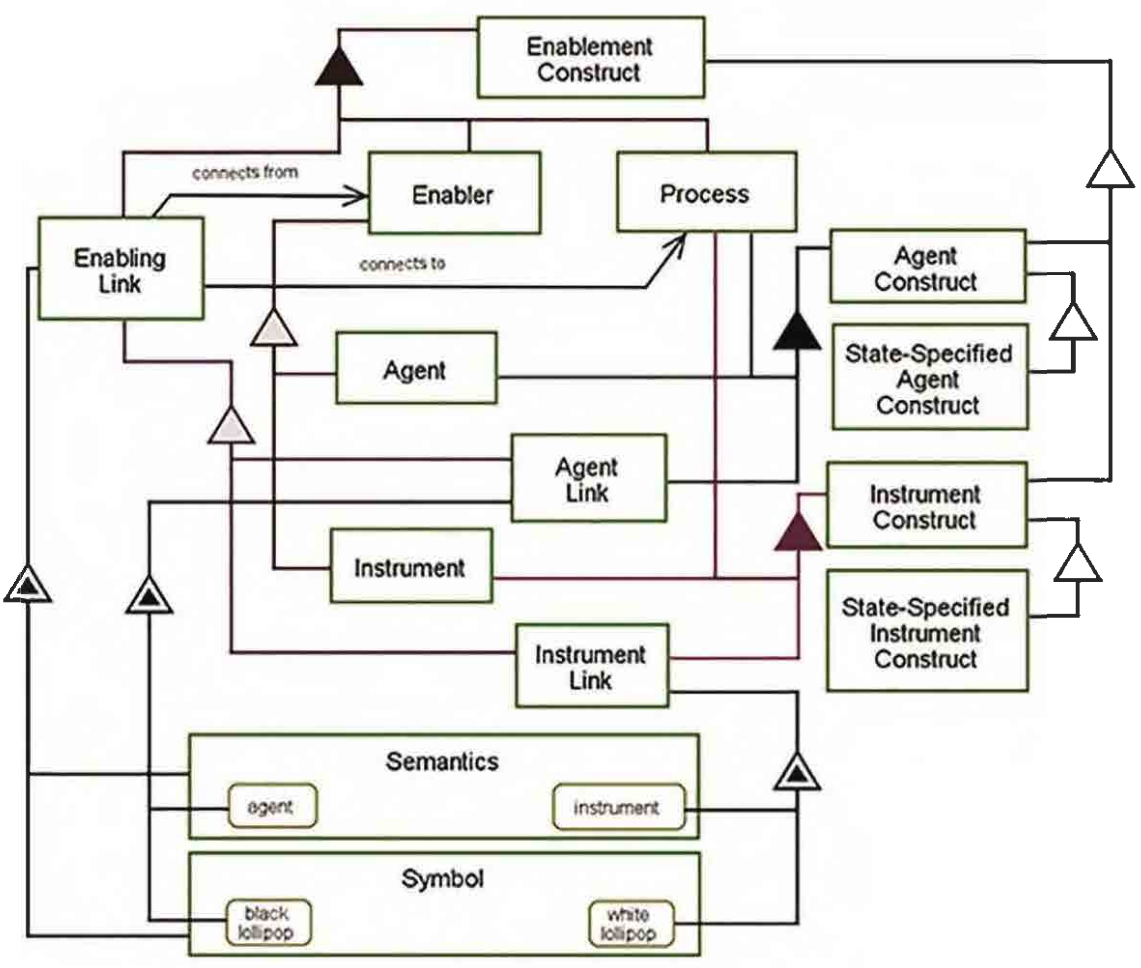
效果关联连接受影响物和过程。

结果关联连接到结果物。

结果关联从过程进行连接。

图 C.16 转换构造关联方向性的 OPM 模型

图C.17 是一个基本启用构造的OPM模型。



使能构造包含使能器、过程和使能关联。

启用关联使能关联展示了语义和符号。

启用关联使能关联从使能器连接。

启用使能关联连接到过程。

启用使能关联的语义 可以是代理或仪器。

启用使能关联的符号可以是黑色棒棒糖或白色棒棒糖。

代理员代理和仪器是使能器。

代理关联和仪器关联是使能关联 。

代理关联展示了启用连接代理语义 和启用连接黑色棒棒糖符号.

仪器关联展示了启用连接 的仪器语义和启用连接的白色棒棒糖符号。

代理构造和仪器构造是启用构造。

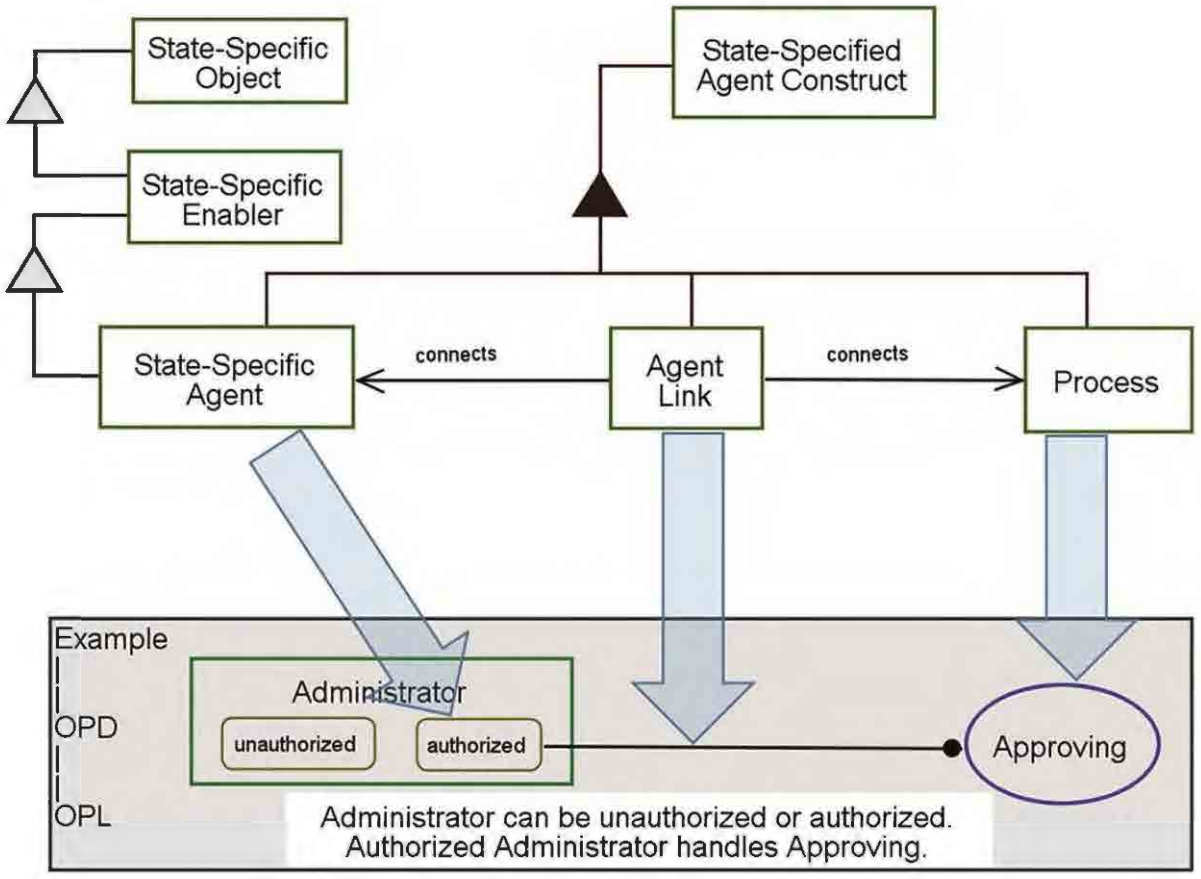
代理构造包含代理、过程和代理关联。

仪器构造包含仪器、过程和仪器关联。

状态-指定代理构造是一个代理构造。

状态-指定仪器是一个仪器构造。

图 C.17 基本启用构造的 OPM 模型



状态-指定代理构造包含状态-指定代理、过程和代理关联。

状态-指定代理是一个状态-指定使能器。

状态-指定使能器是一个 S 状态-指定对象。

代理关联连接状态-指定代理和过程。

图 C.18 带有影像例子的状态-指定代理构造的 OPM 模型

图C.18 描绘了两个OPM模型，上图表达了用于一个状态-指定代理构造的重要关联，下图表达了一个相应的模型构造。前者为后者提供了一个元模型。宽箭头将构造概念部分映像到示例的OPD符号。下面例子中的对象过程图（OPD）是相应的对象过程语言（OPL）。

为示范目的，类似影像图可以表达OPM构造概念模型与应用程序中相应OPM模型之间的对应关系。

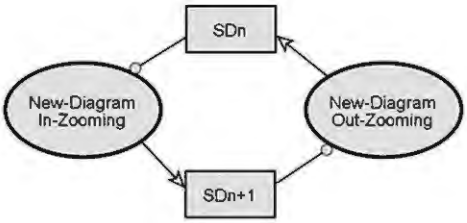
C.5 放大和缩小模型

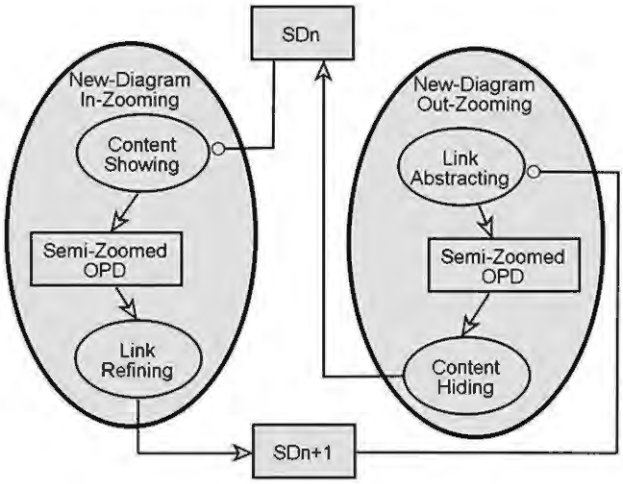
C.5.1 放大和缩小机制

新图放大和新图缩小从现有OPD情景中创建了一个新的OPD情景。新图放大起始于一个细节相对较少的OPD，并将详细阐述或细化作为一个适用于较少细节OPD中特定事物的子OPD而添加进来。新图缩小起始于一个细节相对较多的OPD，并删除了详细阐述或细化以在一个父辈情景中产生的较少细节及更抽象的事物。

新图放大通过创建一个新的OPD，即SDn+1而详细阐述了一个出现在现有OPD中的可细化物，我们可以假设是SDn，它通过增加子过程、相关对象及相关关联说明了可细化物。新图放大和新图缩小中的过程为逆运算。

图C.19 描绘了新图放大和新图缩小的过程。右侧模型使用了左侧模型的图内放大以阐述两个过程，一个用于建立新图放大的上下文，另一个用于创建新图缩小的情景。新图放大从内容显示开始，后接关联细化。新图缩小从关联抽象开始，即关联细化的逆反过程，后接内容隐藏，即内容显示的逆反过程。





新图表放大需要 SDn。

新图表放大产生 SDn+1。

新图表缩放需要 SDn+1。

新图放大拉近到内容展示和关联细化，半放大的 OPD 也按此顺序

内容展示需要 SDn。

内容展示产生半放大的 OPD。

关联细化 消耗半放大的 OPD。

关联细化产生 SDn+1。

新图表缩放拉近到关联抽象和内容隐蔽这种顺序，以及半放大 的 OPD。

关联抽象需要 SDn+1。

关联抽象生成半放大的 OPD。

内容隐藏消耗半放大的 OPD。

内容隐藏 生成 SDn。

图 C.19 新图放大和新图缩小模型

半放大OPD是一个在新图放大或新图缩小期间所创建且随后被消耗的临时对象。半放大OPD仅在新图放大和新图缩小的情景内出现。

图C.20显示了带有SDn、SDn+1展开的新图放大和新图缩小以及图C.19中的半放大OPD。新图放大和新图缩小在图C.20顶部中间所显示的一个特定实例上进行操作，其中众多的可能性之一是具有SDn的细节。在这种情况下，SDn包括可细化过程的P以及四个连接到P的具有不同类型关联的的对象：被消耗物 C、代理A、仪器D和结构物B。

半放大OPD的图内缩小明确指出，它是一个在新图放大以及新图缩小期间所创建和消耗的暂时表示。半放大OPD在这两种情况下是相同的。

内容显示是两个新图缩小的子过程中的第一个。在内容显示期间，P的边界扩展以便为显示其内容-模型子过程P1、P2和P3以及临时模型对象BP而腾出空间。内容显示的结果是对象半缩小OPD的展开。作为一个仅在新图放大情景中可识别的临时对象，第二个子过程，即关联细化在创建SDn+1使将它消耗。在关联细化期间，依附到P轮廓的程序关联迁移到由建模者所确定的适当的子进程中去。因此，由于P1消耗C，消耗关联箭头从P迁移到P1。代理A同时处理P1和P2，因此在 SDn+1中的两个代理关联，一个连到P1，另一个连到P2，替代了SDn 中从A到P的单独的一个 。P3需要D，所以仪器关联从P移动至P3。最后，由于从P1和P3的BP结果消耗了它，相应的结果和消耗关联就被添加进来，使得BP变成了一个P的内部对象，即一个仅在P的情景内可识别的对象，如P1、P2和P3。请留意，BP对P来说就如同半放大OPD对于新图放大一样。

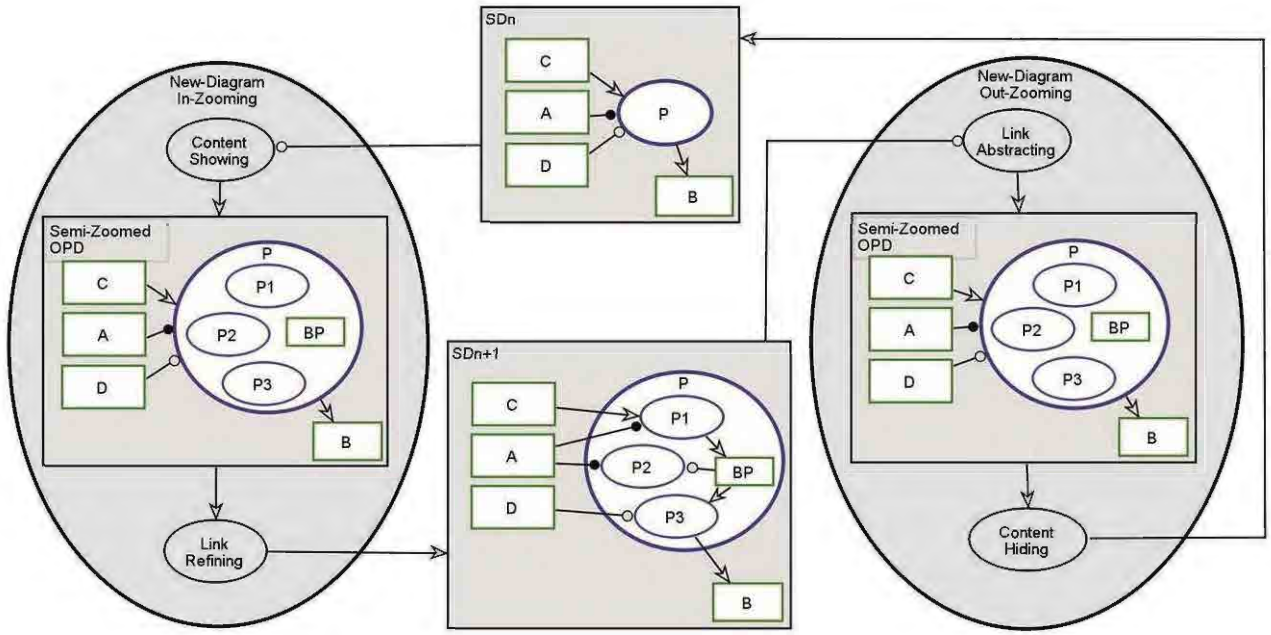


图 C.20 新图放大和新图缩小阐述

C.5.2 简化对象过程图

图内放大可与新图缩小结合起来以简化一个建模者认为过于复杂的OPD建模。图内缩小后紧跟新图放大是在建模者意识到目前的OPD细节已超载时的一种选择。图内缩小以将一个新OPD添加到OPD集为代价减少了由于后续新图放大结果而需要对错综复杂的对象过程图 (OPD) 进行理解的这种认知负担。

图C.21展示了图内缩小，紧跟着是新图缩小。左侧是拥有三个OPD的原有OPD集：SD、SD1和SD1.1。建模者认为SD1过于复杂，为了减少这种复杂性，如图中部所示，建模者选择了P1、P2和P3连同BP一起通过使用新图缩小由P123来取代。右侧是拥有四个OPD重新编号的新OPD集以反映新的层次。新SD1的复杂程度小于原有SD1，具有五个更少的要素（三个过程、一个对象和删除了两个关联 ； 添加了一个过程 - P123）。 P123在新SD1.1.1中经过了新图放大，且这个新的OPD被插入到过程层次中，推动旧SD1.1变成新SD1.1.1。

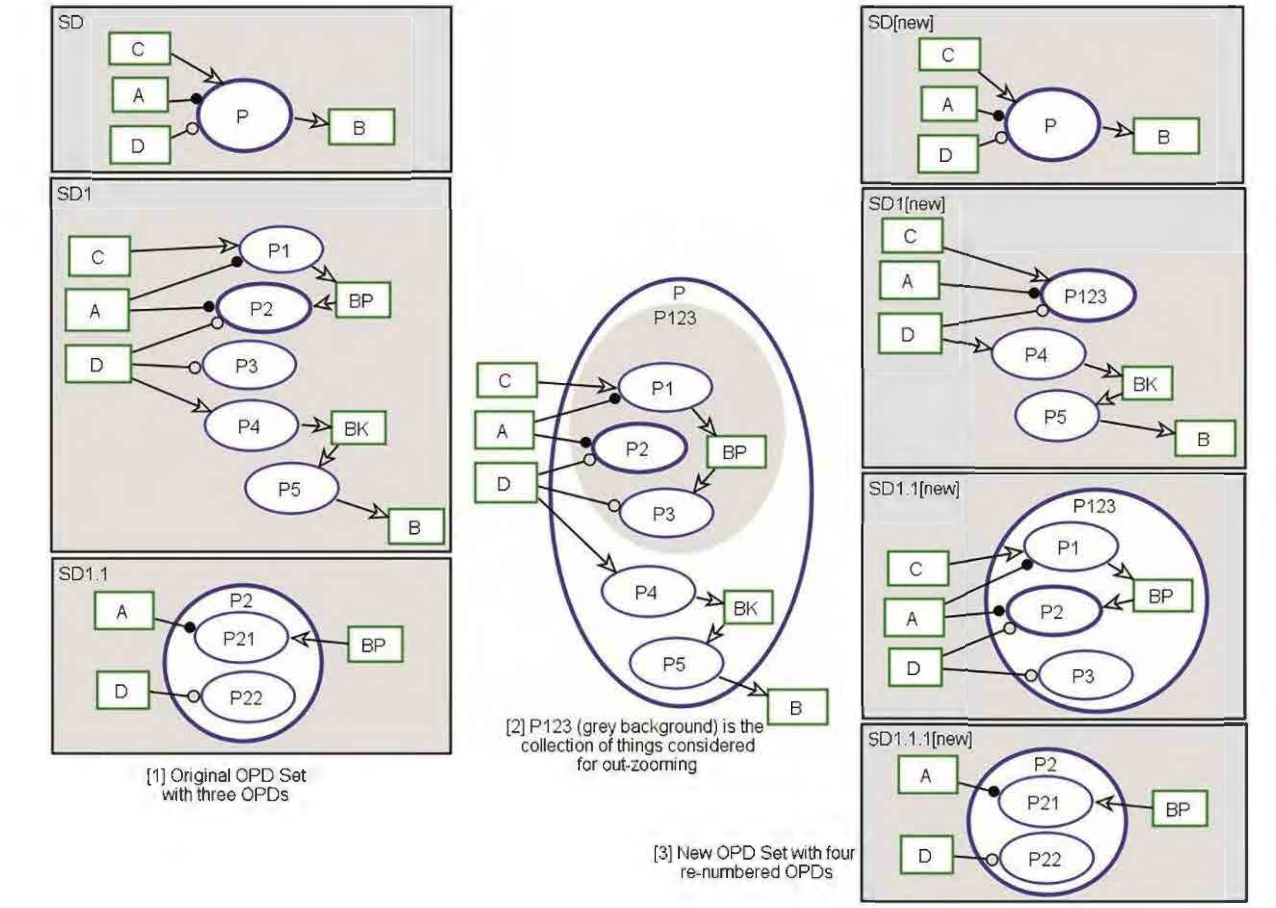


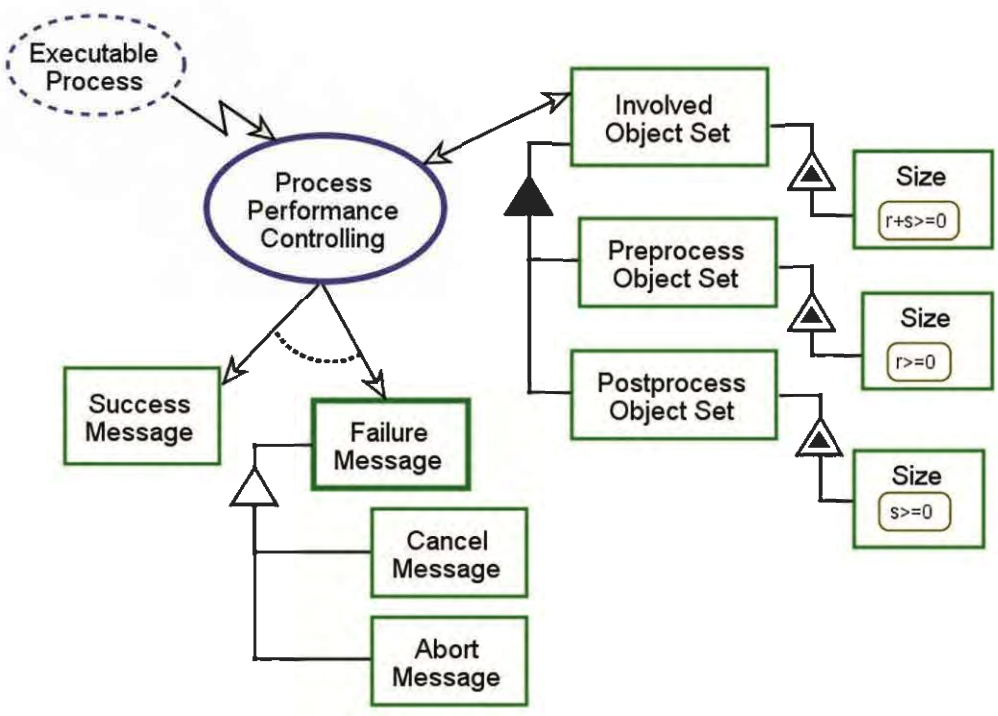
图 C.21 简化对象过程图（OPD）

图内缩小始于选择事物的TO集以将其在当前复杂的OPD中缩小，目的在于将其在一个新的OPD中进行放大。假设一个新的单一过程，PA，来替换TO集，每一个扩展到TO成员的程序关联都需要连接到新的过程PA上去并连接到一个不是TO集的成员中去。PA是一个取代TO成员的新的抽象过程，并成为一个新的模型要素。 PA在一个新OPD中被放大，OPD集标签需要反映新的OPD层次。

图C.21中间的过程P1、P2和P3与对象BP一同是TO的四个成员，其被P123所围绕。创建P123的后果是TO的这四名成员从新的SD1中消失。每一个跨越中间图形的灰-白边界的关联现在都连接到新SD1中的P123的边界上去。连接到新SD1中的P123边界的对象然后再连接到新SD1.1中适当的子进程中去。对象BK不能成为TO的成员，因为如果BK发生在P123中，其关联将会创建直接连接两个过程的两个程序关联， P4到P123和P123到P5。OPM并没有定义这些关联的语义，而且模型将会违背每一个程序关联（除了调用和时间异常关联）将一个对象连接到一个过程中去的这种规定。

C.6 OPM过程性能控制模型

C.6.1 OPM 过程性能控制系统 - 系统图



涉及到的对象集包含前处理对象集和后处理对象集。

前处理对象集展示了规模大小。

前处理对象集的规模大小是 r>=0。

后处理对象集展示了规模大小。

后处理对象集的规模大小是 s>=0。

涉及到的对象集展示了规模大小。

设计的对象集的规模大小是 r+s>=0。

过程性能控制影响涉及到的对象集。

可执行过程是环境性的。

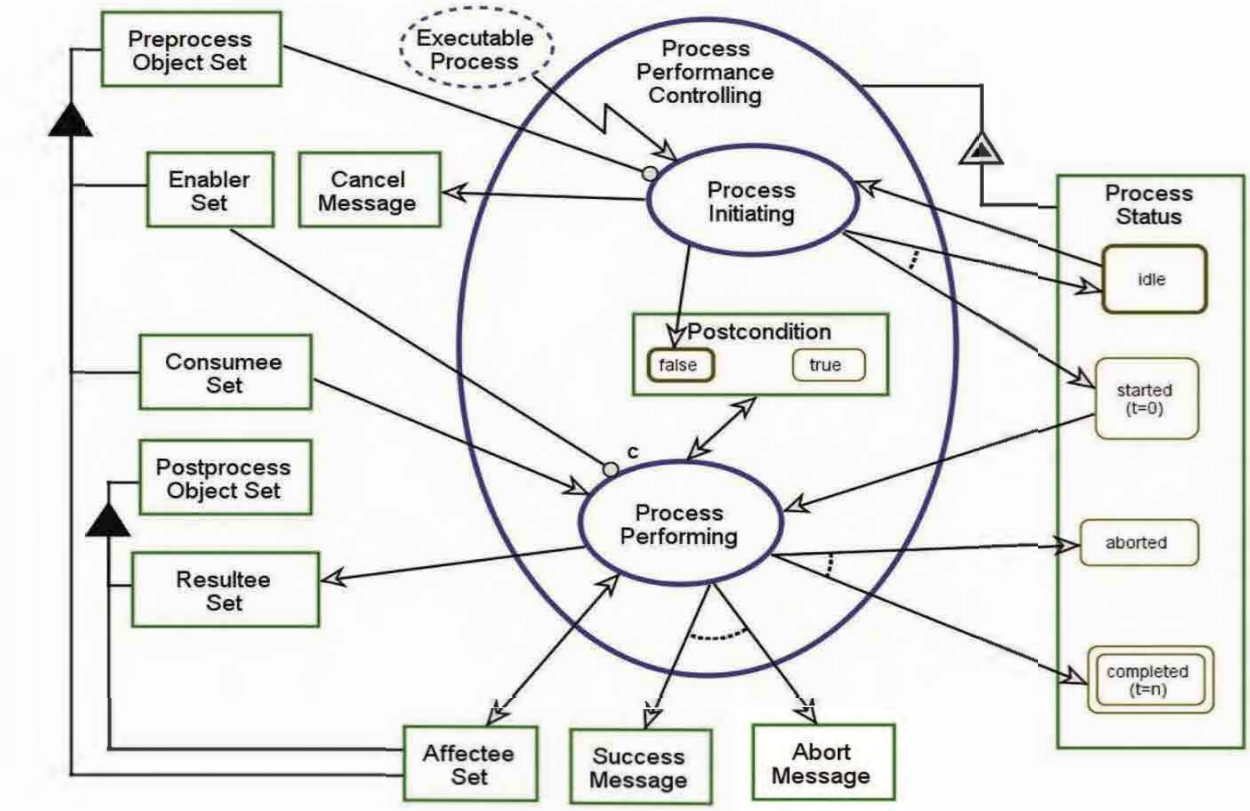
可执行过程调用了过程性能控制。

过程性能控制生成成功报文或失败报文的其中之一。

中止报文和取消报文是失败报文。

图 C.22 过程性能控制系统图 – 系统图

C.6.2 过程性能控制放大为SD1



过程性能控制放大到过程启动和过程运行，后置条件页按次顺序。

前过程对象集包含被消耗物集、受影响物集和使能器集。

后过程对象集包含结果物集和受影响物集。

可执行过程是环境性的。

可执行过程调用过程初始。

过程性能控制 展示了过程状态。

过程状态 可以是闲置、已开始 (t=0)。中止或完成(t=n)。

过程状态 最初是闲置，而最终是完成(t=n) 或中止。

后置条件可以是对或错。

后置条件最初为错。

过程启动需要前过程的对象集。

过程启动将过程状态从闲置改变为正好闲置或已启动之一的状态(t=0)。

过程启动 生成错误后置条件和取消报文。

如果使能器存在，则过程运行发生，否则过程运行将被跳过去。

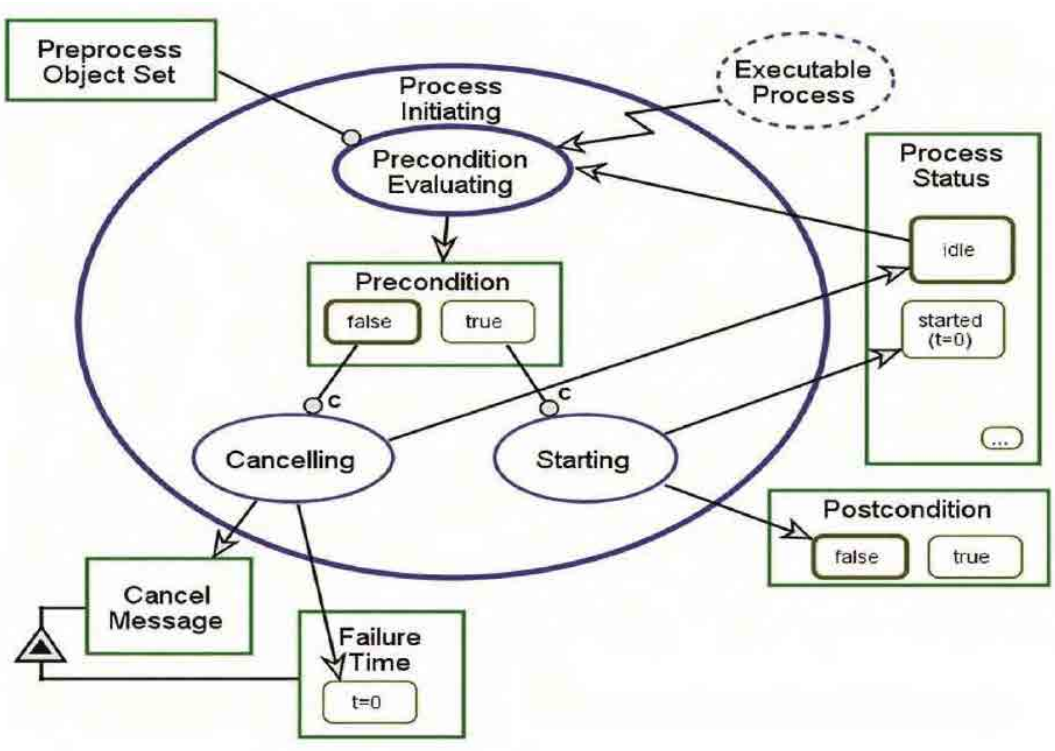
过程运行影响后置条件和受影响物集。

过程运行将过程状态从已启动(t=0) 正好改变为中止或完成 t=n)之一的状态。

过程运行产生结果物集和或成功报文或中止报文。

图 C.23 来自 SD1 中放大的 SD 的过程性能控制

C.6.3 过程启动放大为SD1.1



来自 SD1 的过程启动在 SD1.1 变焦到前置条件评估及并行取消和启动，前置条件也按此顺序。

过程状态可以是闲置、启动(t=0) 或其它状态。

过程状态最初是闲置状态。

后置条件可以是对或错。

后置条件最初为错。

可执行过程是环境性的。

可执行过程涉及到前置条件评估。

前体条件评估产生前置条件。

前提条件前置条件可以是对或错。

前提条件前置条件评估需要前过程对象集。

前提条件前置条件改变了过程状态的闲置状态。

如果前置条件为错，则取消就会发生，否则取消将被跳过去。

取消将过程状态变为闲置状态。

取消产生取消报文。

取消报文展示故障时间。

取消将故障时间值设置为 t=0. 取消报文的故障时间是 t=0。

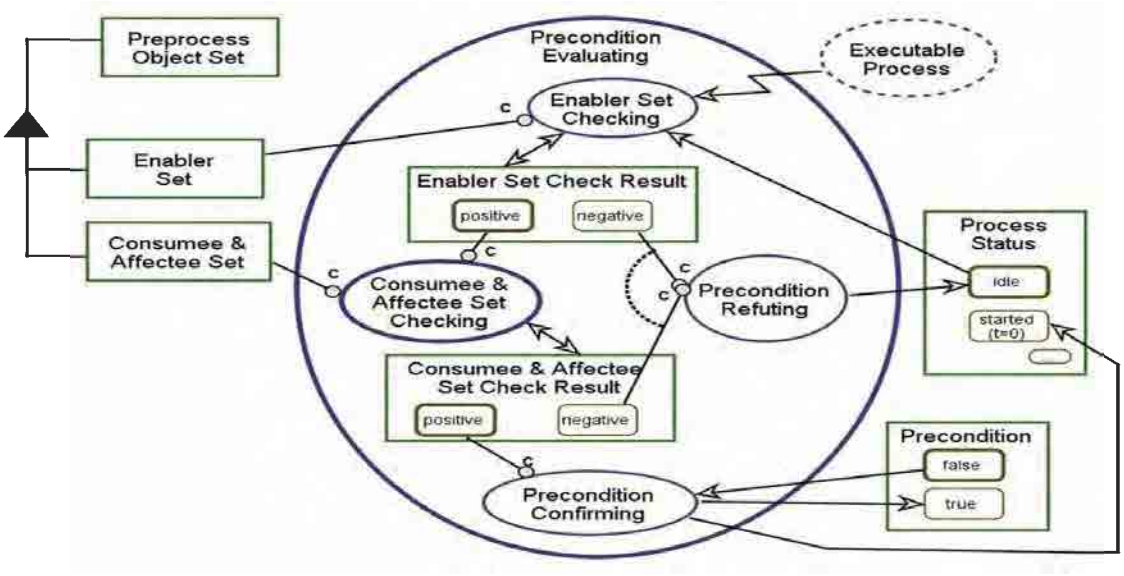
如果前置条件为对，即在前置条件被消耗的情况下，则启动发生，否则启动将被跳过去。

启动将过程状态变为启动状态 (t=0)。

启动产生错误的后置条件。

图 C.24 过程启动放大为 SD1.1

C.6.4 前置条件评估放大为SD1.1.1



来自 SD1.1 的前置条件评估在 SD1.1 中放大至使能器集检查、被消耗物和受影响物集检查、前置条件驳斥和确认这种顺序，以及使能器集检查结果和被消耗物和受影响物集检查结果。

前提过程对象集包含使能器设置和被消耗物和受影响物集。

过程状态可以是闲置、启动 (t=0)或其它状态。

过程状态初始时是闲置状态。

前提条件前置条件可以是对或错。

前提条件前置条件 初始时是错。

可执行过程调用使能器集检查。

启动器使能器集检查需要使能器集存在，否则使能器集检查将会被跳过去。

启动器使能器集检查 改变了过程状态的闲置状态。

启动器使能器集检查结果可以是正面或负面的。

启动器使能器集检查结果初始时是正面的。

启动器使能器集检查 影响使能器集检查结果。

如果使能器集检查结果为正且被消耗物和受影响物集存在时，则被消耗物和受影响物集检查发生，否则 被消耗物和受影响物集检查将被跳过去。

被消耗物和受影响物集检查结果可以是正或负。

被消耗物和受影响物集检查结果 初始时为正。

被消耗物和受影响物集检查影响被消耗物和受影响物集检查结果。

前提条件前置条件驳斥要求使能器集检查结果为负或者被消耗物和受影响物集检查结果为负，否则前置条件驳斥将被跳过去。

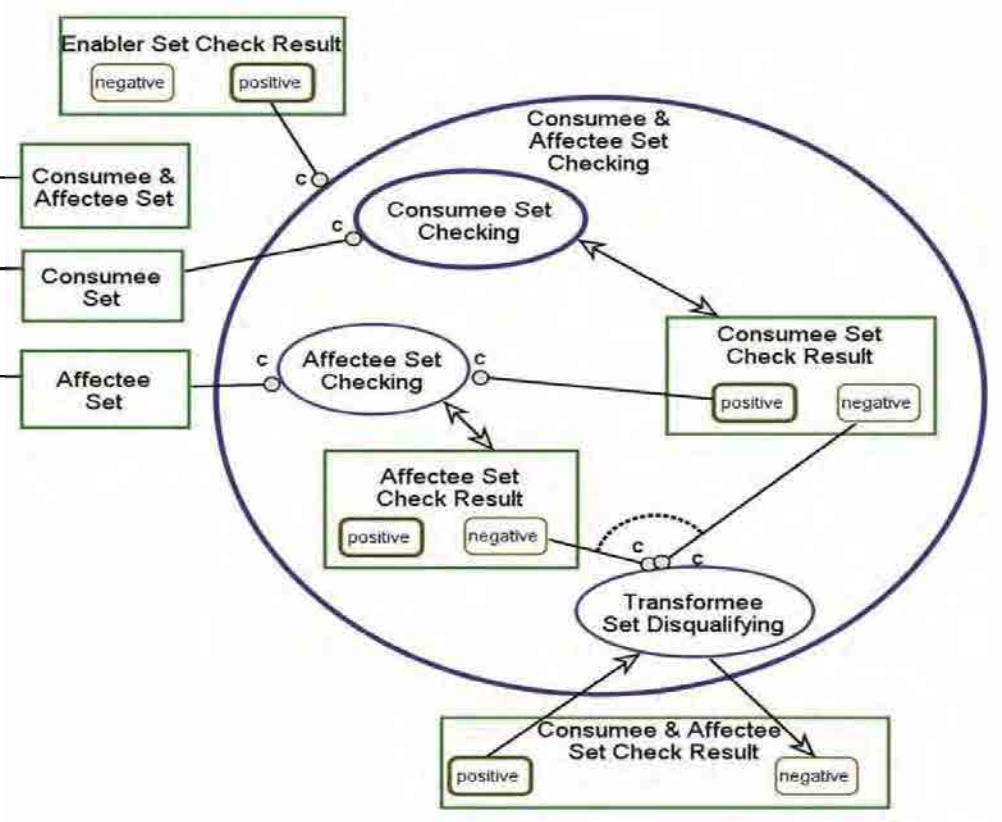
前提条件前置条件驳斥将过程状态将过程状态变为闲置状态。

如果被转换物检查结果为正，则前置条件确认发生，否则前置条件确认将会被跳过去。

前提条件前置条件确认将前置条件从错变为对，过程状态变为启动(t=0)。

图 C.25 前置条件评估被放大为 SD1.1.1

C.6.5 被转换物集检查被放大为SD1.1.1.1



被消耗物和受影响物集检查从 SD1.1.1 受影响物集检查、被消耗物集检查和被转换物集资格取消这样的顺序，以及受影响物集检查结果和被消耗物集检查结果。

启动器使能器集检查结果可为正或负。

启动器使能器集检查结果最初为正。

被消耗物和受影响物集检查结果可以是负或正。

被消耗物和受影响物集检查结果最初为正。

被消耗物和受影响物集包含有被消耗物和受影响物集。

如果使能器集检查结果可为正的化，则被消耗物和受影响物集检查发生，否则被消耗物和受影响物集检查将会被跳过去。

被消耗物集检查结果可为正或负。

被消耗物集检查结果最初为正。

如果被消耗物集存在，则被消耗物集检查检查发生，否则被消耗物集检查将会被跳过去。

被消耗物集检查影响被消耗物集检查结果。

如果被消耗物集检查结果为正，受影响者集存在的话，则受影响物集检查发生，否则受影响物集检查将会被跳过去。

受影响物集检查产生。

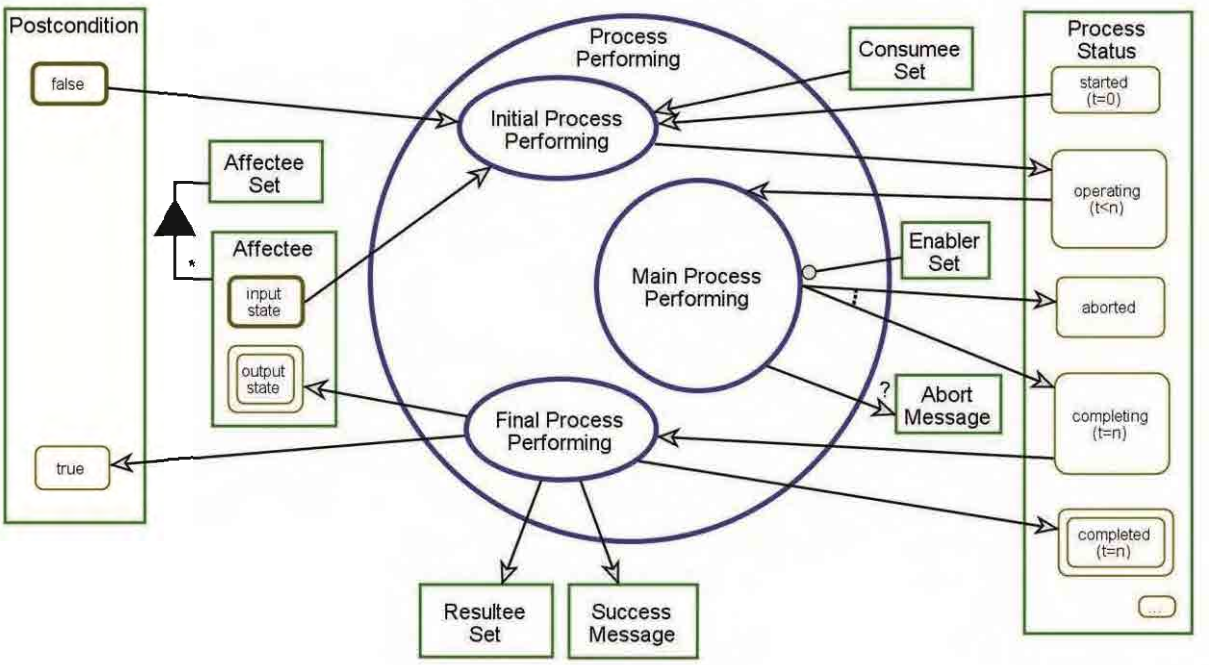
受影响物集检查结果可为正或负。

如果受影响物集检查结果为负或 被消耗物设置检查结果为负，则被转换物集资格取消发生。

被转换物集资格取消将被消耗物和受影响物集检查结果从正变为负。

图 C.26 被转换物集检查放大为 SD1.1.1.1

C.6.6 过程操作被放大为SD1.2



过程操作从 SD1 将 SD1.2 放大至最初过程操作、主要过程操作和最终过程操作，以这种顺序进行。

过程状态可以是闲置状态、启动状态 (t=0)、操作状态 (t<n)、中止状态、完成状态(t=n)、完成状态(t=n)或其它状态。

过程状态最终是完成 (t=n)。

后置条件可以是对或错。

后置条件最初为错。

受影响设置包含选择性受影响物。

受影响物可以是输入状态或输出状态。

受影响物最初为输入状态最终为输出状态。

初始过程操作将过程状态从启动(t=0) 变为操作(t<n)、改变了后置条件的错状态以及改变了受影响物的 输入状态。

初始过程操作消耗受消耗物设置。

主要过程操作要求使能器设置。

主要过程操作产生了一个中止报文。

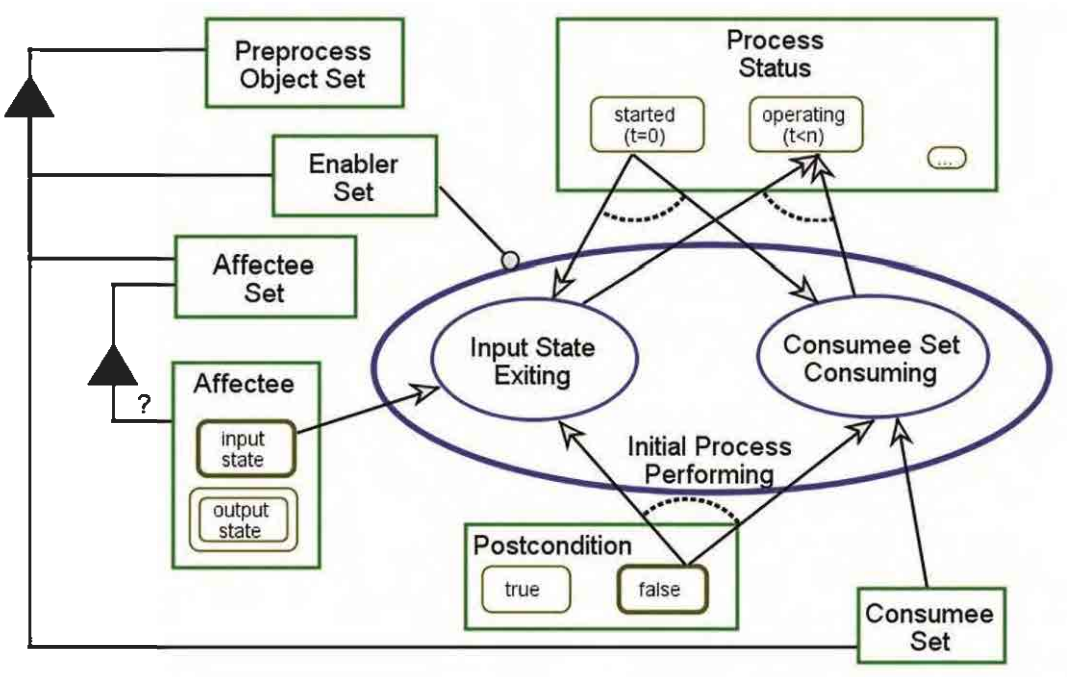
主要过程操作将过程状态从操作(t<n) 改变为完成 (t=n) 或中止之一。

最终过程操作将过程状态从正在完成(t=n) 变为已经完成(t=n)，后置条件变为对以及受影响物变成了输出状态。

最终过程操作生成成功报文和结果物设置。

图 C.27 过程操作被放大为 SD1.2

C.6.7 初始过程操作放大为SD1.2.1



来自 SD1.2 的初始过程操作在 SD1.2.1 中放大至并行输入状态退出及被消耗物集消耗。

前提条件前置条件对象设置包含使能器集、受影响物集和被消耗物集。

受影响物集包含可选择的受影响物。

受影响物可以是输入状态或输出状态。

受影响物初始为输入状态，最终为输出状态。

过程状态可以是启动(t=0)，操作(t<0)或其它状态。

后置条件可以为对或错。

后置条件初始为错。

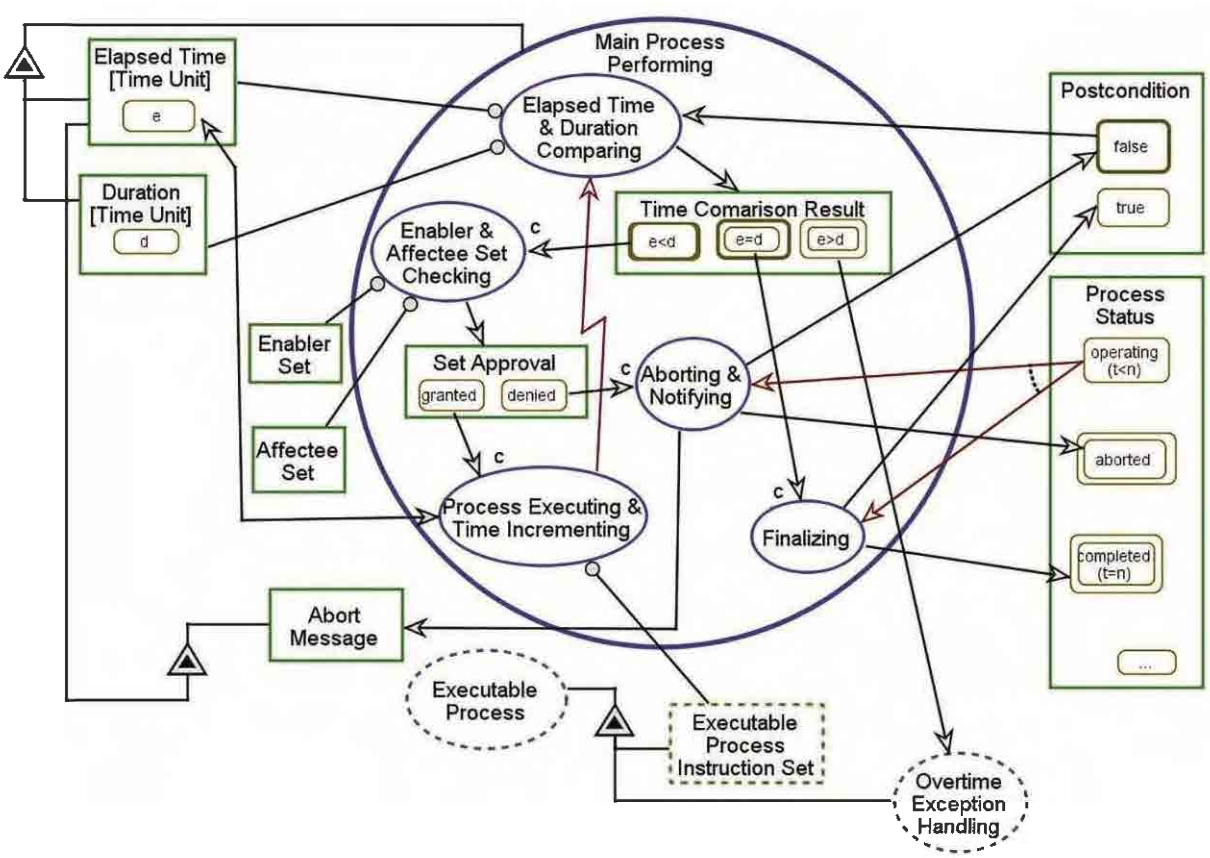
初始过程操作需要使能器集。

输入状态退出改变了受影响物的输入状态。

被消耗物集消耗或输入状态存在将过程状态从启动(t=0) 改变为操作(t<n) ，同时改变后置条件的失败状态。

图 C.28 初始过程操作被放大为 SD1.2.1

C.6.8 主要过程操作被放大为SD1.2.2



主要过程操作从 SD1.2.2 中放大至 逝去的时间和持续时间比较、使能器和受影响物集检查、中止和通知、时间递增和完成，以该序列进行以及时间比较结果和集批准。

可执行过程展示可执行过程指令集和超时异常处理。

可执行过程、可执行过程指令集以及超时异常处理是环境性的。

过程状态可以是中止、完成 (t=n)、操作 (t<0)或其它状态。

过程状态最终是中止或完成的(t=n)。

后置条件可以是对或错。

后置条件初始是错的。

主要过程操作展示了时间单位中的逝去时间和时间单位中的持续时间。

中止报文展示了时间单位中的逝去时间。

时间单位中的逝去时间是 e。 时间单位中的持续时间是 d。

逝去时间和持续时间需要时间单位中的逝去时间和时间单位中的持续时间。

逝去时间和持续时间比较改变了后置条件的错误状态。

逝去时间和持续时间比较 产生了时间比较结果。

时间比较结果可以是 e<d, e=d 或 e>d。

时间比较结果初始是 e<d 或 e=d 和最终 e=d 或 e>d。

启动器使能器和受影响物集检查需要使能器和受影响物集。

如果时间比较是 e<d, 在使能器和受影响物集检查消耗时间比较结果的情况下, 使能器和受影响物集检查将会发生，否则使能器和受影响物集检查将被跳过去。

启动器使能器和受影响物集检查需要使能器集。

启动器使能器和受影响物集检查产生集批准。

集批准可以被可授予或拒绝。

如果集批准被拒绝，在中止和通知消耗集批准的情况下，中止和通知会发生，否则中止和通知将被跳过去。

中止和通知将过程状态从操作(t<n) 变为中止，后置条件变为错误。

中止和通知生成中止报文。

如果时间比较结果是 e=d, 在结束消耗时间比较结果的情况下，中止报文结束发生，否则结束将被跳过去。 结束将过程状态从操作 (t<n) 变为完成(t=n) 以及后置条件变为正确。

过程执行和时间递增需要可执行仪器集。

如果集批准被授权，在过程执行和时间递增消耗集批准的话，过程执行和时间递增发生， 否则程执行和时间递增将被跳过。

时间递增消耗集可以吗?

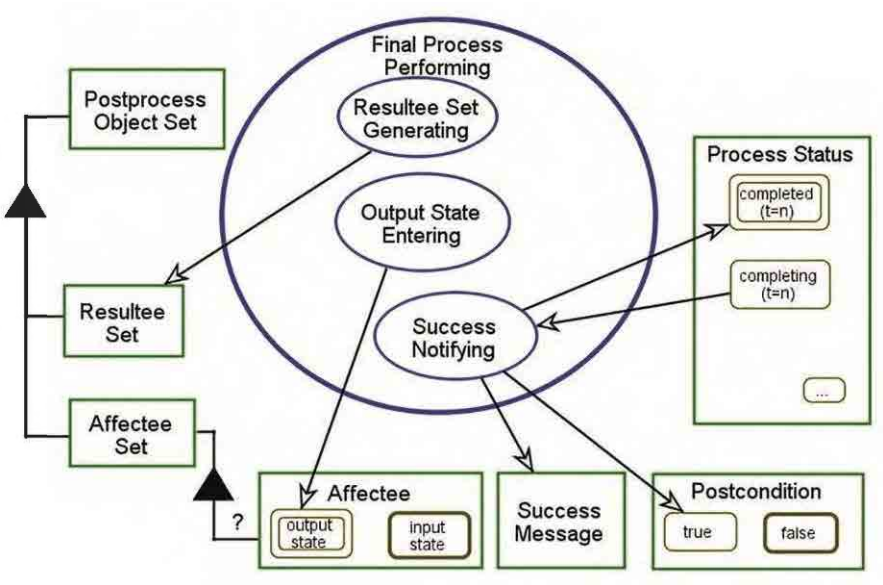
时间递增 生成时间单位中的 elt=1..ext 逝去时间。

过程执行和时间递增改变了时间单位中的逝去时间值 e。

过程执行和时间递增 调用逝去时间和持续时间比较。超时异常处理消耗 e>d 时间比较结果。

图 C.29 主要过程操作被放大为 SD1.2.2

C.6.9 最终过程操作放大为SD1.2.3



来自 SD1.2 的最终过程操作在 SD1.2.3 中放大至并行结果物集生成、输出状态进入和成功通知，以此排序。

后条件对象集包含结果集和受影响物集。

受影响物集包含可选择的受影响物。

受影响物 可以为输入状态或输出状态。

受影响物初始为输入状态和最终为输出状态。

过程状态可以是完成 (t=n)、正在完成 (t=n) 或其它状态。

过程状态最终为完成 (t=n)。

后置条件可以是对或错。

后置条件初始为错误。

结果物集生成受影响物集。

输出状态进入将受影响物变成输出状态。

成功通知将后置条件变成正确。

成功通知生成成功报文。

图 C.30 最终过程操放大为 SD1.2.3

DC

附 录 D
（资料性附录）
OPM 动态性和仿真

D.1 OPM 可执行性

一个OPM模型提供了可执行性，即在一个合理设计软件环境中通过动画执行其模型去模拟一个系统的能力。

D.2 变化和效果

一个对象的变化是处于该对象状态中的一种改变。更具体地说，一个对象的变化反映在另外一种状态替代了目前的状态。可导致其发生变化的唯一的事物就是一个过程。过程导致的变化是通过将处于一定状态下作为一个输入的对象输出到另一种状态中去。因此，一个事物的变化意味着该对象在其所处状态中的变化。

有状态对象可能会受到影响，即它们的状态会发生变化。这种变化机制强调了对象与过程之间亲密且不可分割的关系。状态中的这种变化是过程对对象所产生影响的这种效果。

效果而因此被定义为一个过程所导致一个对象状态中的变化。

尽管术语“变化”和“效果”几乎同义，但在其用法上却存在着一种微妙差异。效果用于所指过程会对对象做些什么，而变化则意味着由于过程产生的一种结果对对象发生了什么样的变化。效果的上述定义在随后的附录中被精简为输入和输出关联这种概念。

D.3 存在与转换

当一个过程作用于一个对象时,唯一一种可能性是对象发生变化。一个过程可以影响改变一个事物，但它也可做的更激烈些：它可以生成一个对象或消耗一个对象。转换这个术语包含了使一个过程可通过它们对一个对象采取行动的这样三种附加模式：构造、效果和消耗。

构造与创造、生成或产生是同义的，效果与改变或转换是同义的，消耗则与淘汰、终止，废止或破坏是同义的。一个过程对对象所产生的效果是将对象从它的一种状态转变为另一种状态，而对象则依然存在，并仍会维持其在过程发生之前所拥有的身份。构造和消耗改变了对象的绝对性的存在，因此比效果具有更深刻的变革性。

当一个过程构造（发生、生成、创建或导致结果发生）一个对象时，就意味着这个之前并不存在的对象已经发生了根本的转变。这种转变使得这个对象脱颖而出，成为系统中可识别且具有意义的对象。也就从此值得作为一个新的、独立的实体来对待和参考。

当一个过程消耗（消除或毁坏）一个对象时，其意味着这个以前存在并在系统中可识别且具有意义的对象已经发生了根本性的转变。结果，这个对象就不再存在于系统中并不再可识别。

D.4 OPM原理的时间轴

默认情况下，一个放大过程中的执行时间轴起始于图形顶端，终止于图形底部，除非另有指示要偏离时间轴。这些指示包含可能会导致循环的过程范围内部的特殊对象过程方法（OPM）内部事件和那些

名称是或者结尾于短句“异常退出”的过程。无论其图形位置如何，如果该过程被调用的话，情景过程，即该过程处于被嵌入其中的放大过程将会立即且无条件地退出。

过程椭圆型的最高顶点充当为一个参照点，这样，一个参照点高于其对等体（多个）的过程更早启动。如果两个或多个过程的参照点处于相同的高度（几个图形单元内，例如像素，公差），这些过程则同时且并行开始。

D.5 定时事件

到目前为止所呈现的事件皆为对象或状态事件：他们发生在一个特定对象开始存在或进入一个特定状态的时刻。相反，定时事件取决于系统中一个特定时间的到来，如下所示。

一个状态事件可以代表一个时间事件，如图 D.1 所示。

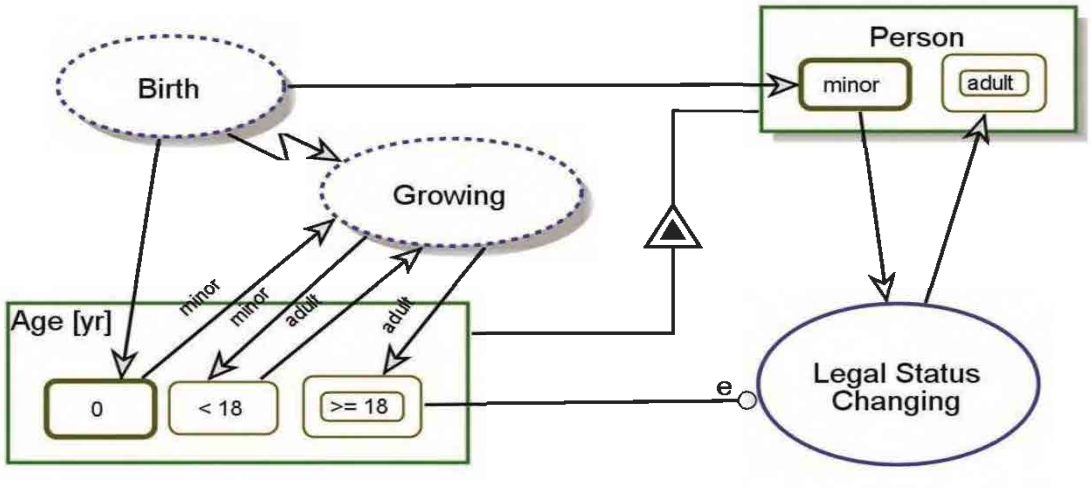


图 D.1 18 岁时从未成年人到成年人的法制体系模型变化

Figure D.2 显示了引发法制体系状态变化的系统时钟。

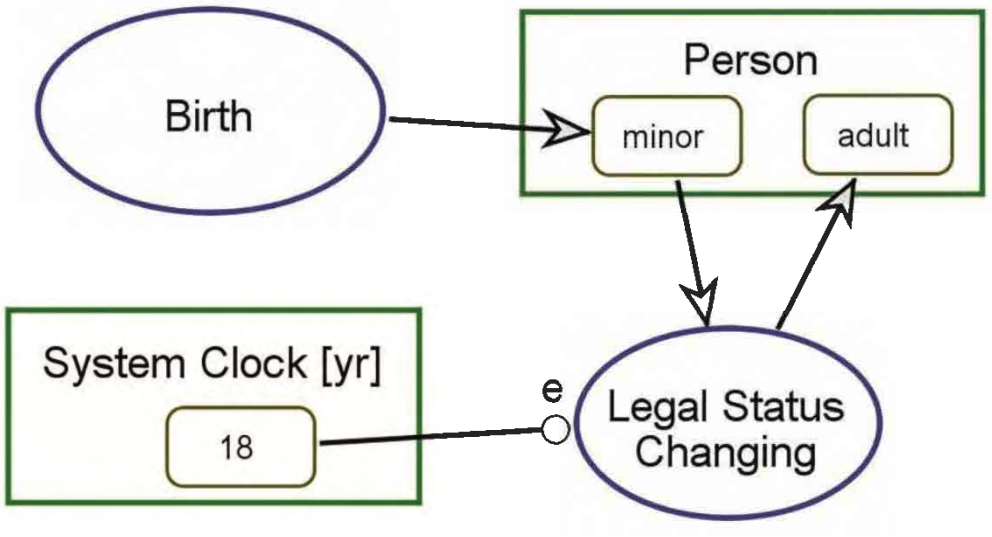


图 D.2 引发法制系统状态变化的系统时钟事件

D.6 对象历史和生命周期图

在任何时间点，对象都可以是其状态之一或存在于两种状态之间的过渡。

一个生命周期图的图形显示的是一个在系统寿命期间的任意一个时间点，系统中究竟存在有什么样的对象、每个对象处于何种状态和哪个过程处于活动期。

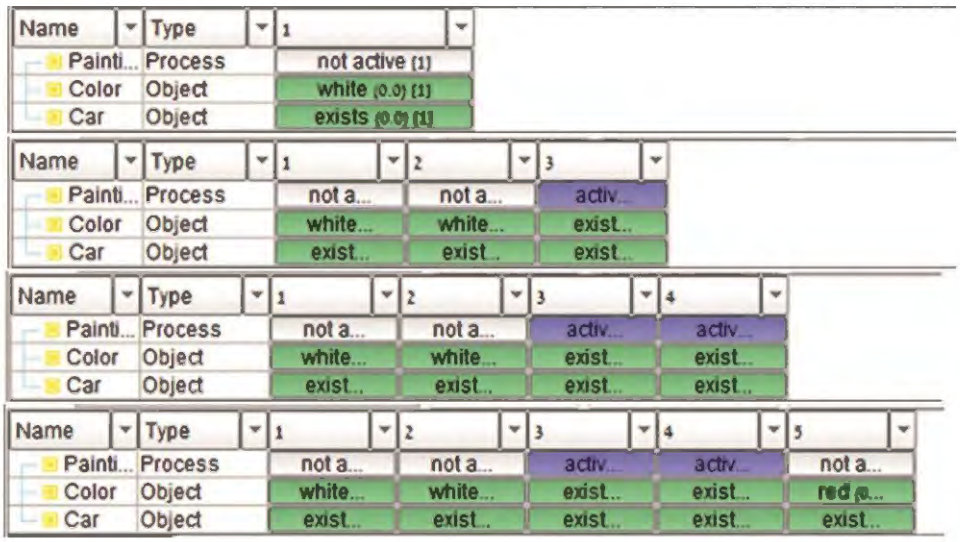


图 D.3 汽车涂漆四个生命周期图例子

图 D.3 中所示的四个生命周期图记录了随着时间的推移，汽车涂漆系统的历史。这四个生命周期图垂直堆叠地显示出来以便于对它们的审查。第一个图中只显示了第一个时间段。喷涂未被激活，汽车呈白色。

第二个图中显示了前三个时间段。在第三个阶段中，喷涂激活，汽车不再是白色。第四个阶段发生了同样的情况，如第三图所示。最后，在第五个阶段中，如图的底部所示，喷涂不再有效，车呈红色。

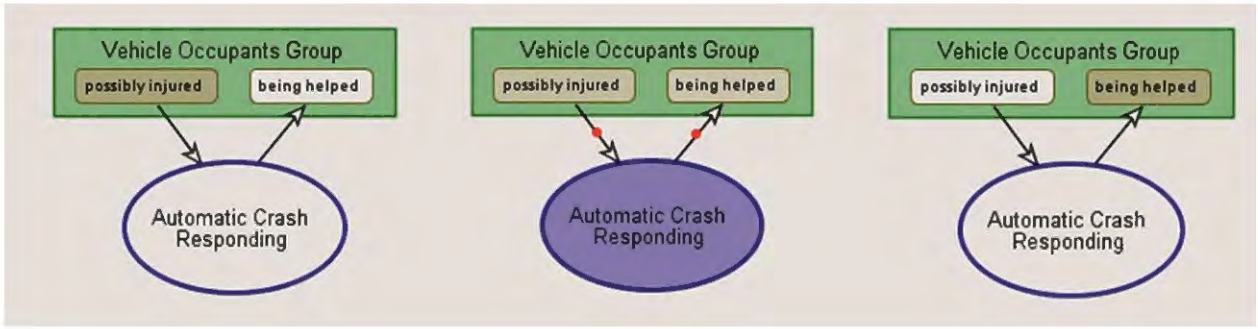


图 D.4 执行用于自动碰撞响应的 OPM 模型

图 D.4 呈现了三个 OPCAT 视频截图，展示了执行一个 OPM 模型的三个阶段。左侧屏幕截图显示自动碰撞响应过程发生之前的系统。这个阶段中，车辆乘员组处于其输入状态，也许受伤，由实心（棕色）的状态进行了标记。

中间的屏幕截图显示了运行的过程，用实心（蓝色）来标记。在自动碰撞响应处于活跃期间（即当

它执行时），对象车辆乘员组处于从输入状态，也许受伤状态到其输出状态，即被帮助状态的过渡阶段。这由两个半实心的状态来标记。

在观察动画时你看发现输入状态在逐渐淡出，而输出状态变得坚实起来。同时，两个红点沿着输入-输出关联对进行移动，意味着系统的“控制”或系统在每个时间点所处的位置。一个红点从输入状态移动到影响过程。同时，第二点从该过程沿着输出连接移动到输出状态。

最后，右边的屏幕截图显示了自动碰撞响应过程终止后的系统。在这个阶段中，车辆乘员组处于其输出状态，正在得到帮助。

实施系统模型的动画具有几种好处。第一，它是一个动态的可视化辅助，有助于建模人员和目标受众人员皆可随着时间的推移去遵循并理解系统行为。第二，宛如一种编程语言的调试器，它便便于系统动态性的验证，并在其执行控制流程中发现逻辑设计错误。因此，强烈建议在其构造期间频繁性地将系统模型驱动起来。

D.7　过程持续时间

系统时间单位是用于指定系统中所有过程的全部持续时间种类的默认时间单位，除非有一个用于特定过程的明确的不同时间单位，在此情况下，该时间单位覆盖系统时间单位。

表达一个 OPD 中相关过程属性值的一个严谨方式是使用展示-表征和特化关联。假设以下是相关过程属性，实例 1 展示了两种方式来以图形形式配置属性：

——时间测量单位；

——持续时间参数，其可为下列之一：

　　——三个值，分别代表最短、预期和最长持续时间；

　　——两个值，分别代表最短和最长持续时间；或

　　——一个值，代表最短和最长持续时间。 及，

——持续时间分布名称和其一个或多个参数。

以下是可能标准分布及其参数：

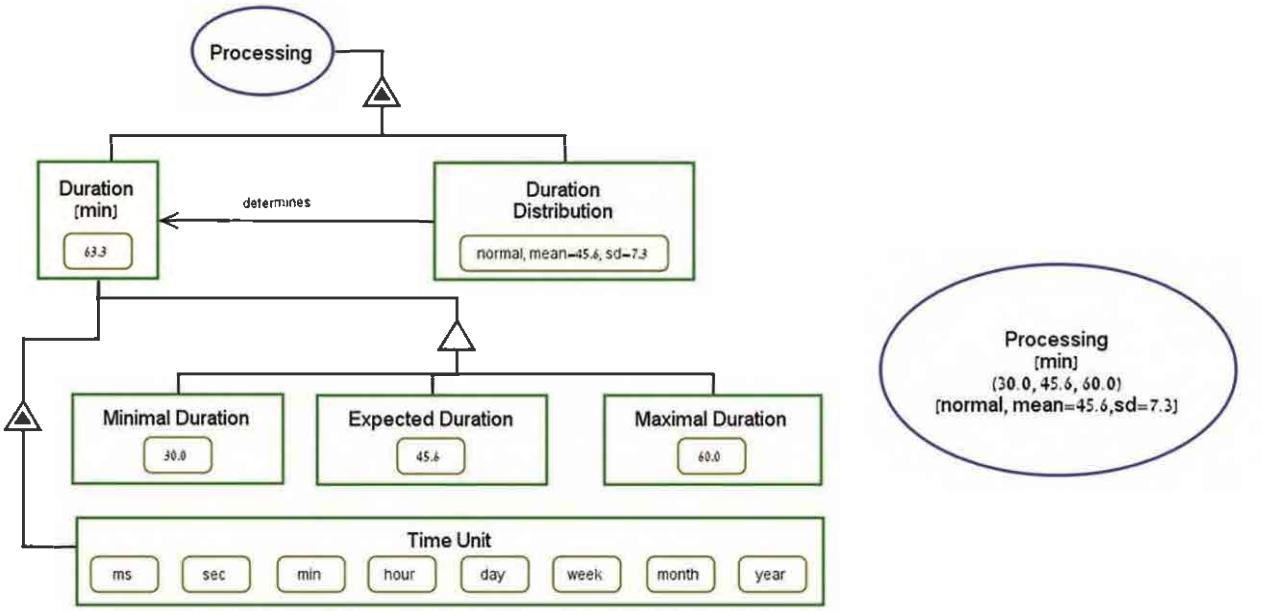
——正态分布，mean=xx；sd=yy；

——统一分布，a=xx，b=yy； and，

——指数分布，lambda=xx。

注：秒的时间测量单位习惯上是默认，常被省略。

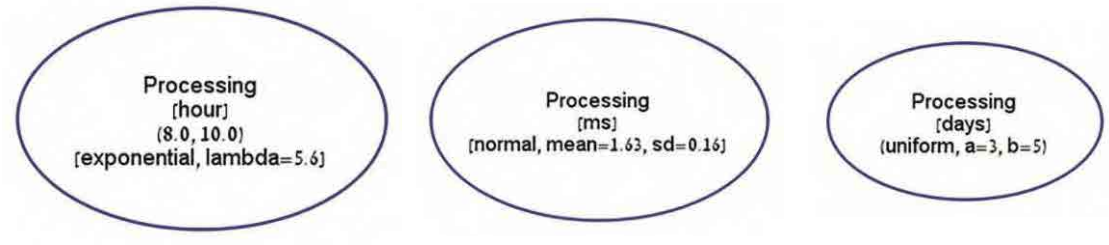
示例1：图 D.5 说明了一个带有属性值过程处理持续时间的元模型。左边是完整的元模型。右边的过程显示了一个记录左边所有数据的严谨方式，除了一个运行时间属性的（实际）持续时间。在该例子中的持续时间分布为正态，具有平均值 45,6 分钟和标准偏差 7,3 分钟。



过程处理分别显示 30.0、45.6 和 60.0 分钟最短持续时间、预计持续时间和最长持续时间以及带有参数平均值=45.6 和 sd=70.0 的正态持续时间分布。

图 D.5 带有属性值的过程处理持续时间

示例2：图 D.6 提供了过程持续时间例子。



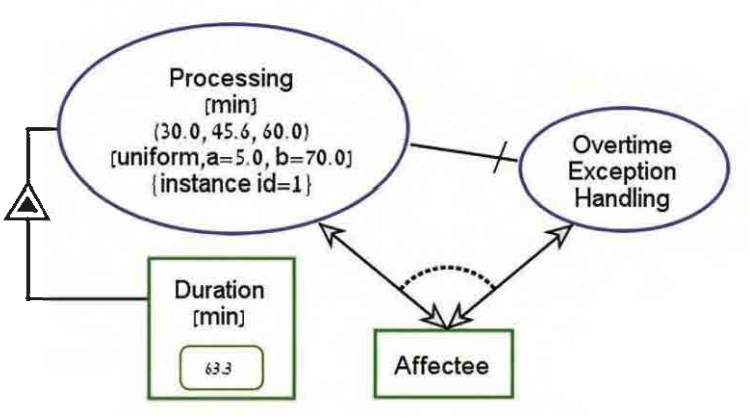
过程处理展示了 8 小时和 10 小时的最短持续时间和最长持续时间以及拥有参数λ = 5.6 的指数持续时间分布。

过程处理展示了具有参数平均值=1.63 和 SD=0.16 毫秒的正态持续时间分布。

过程处理展示了拥有参数 a=3 和 b=5 天的统一持续时间分布。

图 D.6 过程持续时间例子

示例3：图 D.7 中，过程处理〔实例 id=1〕持续时间为 63,3 分钟，因此超时异常处理发生。



过程处理分别展示了 30.0、45.6 和 60.0 分钟最短持续时间、预计持续时间和最短持续时间以及拥有参数 a= 5.0 和 b=70.0 的统一持续时间分布。

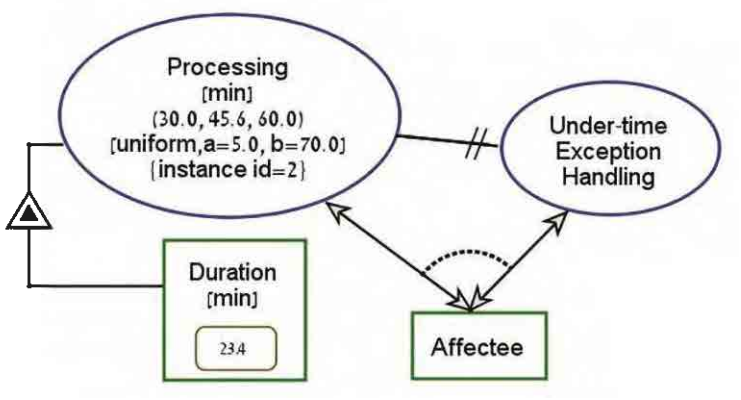
或是过程处理或是超时异常处理影响受影响物。

如果过程处理持续时间超过 60 分钟，则超时异常处理发生。

超时异常处理影响受影响物。

图 D.7 超时异常例子

示例4：图 D.8 中，过程处理 {实例 id=2}持续时间是 23.4 分钟，因此时间不足异常处理发生。



过程处理分别展示了 30.0、45.6 和 60.0 分钟的最短持续时间、预计持续时间和最长持续时间以及拥有参数 a= 5.0 和 b=70.0 的统一持续时间分布。

或者过程处理或者时间不足异常处理会影响受影响物。

如果过程处理持续时间短于 60 分钟，则时间不足异常处理则会发生。

时间不足异常处理影响受影响物。

图 D.8 时间不足异常例子

参 考 文 献

[1]ISO/IEC 14977:1996, Information technology — Syntactic metalanguage — Extended BNF)

[2]ISO/TC 184/SC 5. Terms of Reference: Study Group to Explore OPM for Modeling Standards, 2009

[3]ISO/TC 184/SC 5 N1070 Object Process Methodology Study Group – Interim Report 2010

[4]ISO/TC 184/SC 5 N1111 Object Process Methodology Study Group – Final Report 2011

[5]Bibliowicz A. A Graph Grammar-Based Formal Validation of an Object-Process Diagram, M. Sc. Thesis, Technion, Israel, 2008

[6]Bibliowicz A., & Dori D. A Graph Grammar-Based Formal Validation of Object-Process Diagrams. Soft. Syst. Model. 2012, 11 (2) pp. 287–302

[7]Crawley E. F, Malmqvist J., Östlund S., Brodeur D. R. Rethinking Engineering Education: The CDIO Approach. Springer, 2007

[8]Dori D. Object-Process Methodology - A Holistic Systems Paradigm. Springer Verlag, Berlin, 2002

[9]Dori D. Words from Pictures for Dual Channel Processing: A Bimodal Graphics-Text Representation of Complex Systems. Commun. ACM. 2008, 51 (5) pp. 47–52

[10]Dori D., Feldman R., Sturm A. From conceptual models to schemata: An object-process-based data warehouse construction method Inf. Syst. 2008, 33 (6) pp. 567–593

[11]Dori D. Object-Process Analysis: Maintaining the Balance between System Structure and Behavior. Journal of Logic and Computation. 1995, 5 (2) pp. 227–249

[12]Dori D. Object-Process Methodology – A Holistic Systems Paradigm, Springer Verlag. Foreword by Edward Crawley, Berlin, Heidelberg, New York, 2002

[13]Dori D., Reinhartz-Berger I., Sturm A. LNCS 2813, pp. 570-572, 2003

[14]Dori D. The International Journal on Very Large Data Bases. 2004, 13 (2) pp. 120–147

[15]Estefan J. Survey of Model-Based Systems Engineering (MBSE) Methodologies 2. Differentiating Methodologies from Processes, Methods, and Lifecycle Models. Jet Propuls. 2008, 25 pp. 1–70. Available at: http://www.omgsysml.org/MBSE_Methodology_Survey_RevB.pdf

[16]Grobshtein Y., & Dori D. Generating SysML Views from an OPM Model: Design and Evaluation. Syst. Eng. 2011, 14 (3) pp. 327–340

[17]Myersdorf D., & Dori D. The R&D Universe and Its Feedback Cycles: an Object-Process Analysis. R & D Manag. 1997, 27 (4) pp. 333–344

[18]Oliver D.W., Andary J.F., Frisch H. Model-based systems engineering. In Handbook of Systems Engineering and Management, pp. 1361-1400, 2009

[19]Osorio C.A., Dori D., Sussman J. COIM: An Object-Process Based Method for Analyzing Architectures of Complex, Interconnected, Large-Scale Socio-Technical Systems. Syst. Eng. 2011, 14 (3)

[20]Peleg M., & Dori D. The Model Multiplicity Problem: Experimenting with Real-Time Specification Methods. IEEE Trans. Softw. Eng. 2000, 26 (8) pp. 742–759

[21]Peleg M. J., and , D., A Methodology for Eliciting and Modeling Exceptions. (4), pp. 736-747, 2009 [22] OPCAT. Enterprise Systems Modeling Laboratory, Technion, Haifa, Israel, http://esml.iem. technion.ac.il/opm/

[23]Ramos A.L., Ferreira J.V., Barceló J. LITHE: An Agile Methodology for Human-Centric ModelBased Systems Engineering. IEEE Trans. Syst. Man Cybern. A Syst. Hum. 2012

[24]Reichwein A., & Paredis C. Overview of Architecture Frameworks and Modeling Languages for Model-Based Systems Engineering. Proceedings of the ASME 2011 International Design Engineering Technical Conferences Computers and Information in Engineering Conference, 1-9, 2011

[25]Reinhartz-Berger I., & Dori D. A Reflective Metamodel of Object-Process Methodology: The System Modeling Building Blocks. In: Business Systems Analysis with Ontologies, (Green P., & Rosemann M. eds.). Idea Group, Hershey: 2005, pp. 130–73

[26]Sharon A., de Weck O., Dori D. Model-Based Design Structure Matrix: Deriving a DSM from an Object-Process Model. Syst. Eng. 2012, pp. 1–14

[27]Somekh J., Choder M., Dori D. Conceptual Model-Based Systems Biology: Mapping Knowledge and Discovering Gaps in the mRNA Transcription Cycle. PLoS ONE. 2012 Dec. 20, 7 (12) p. e51430. DOI:10.1371/journal.pone.0051430

[28]Soffer P., Golany B., Dori D. ERP Modeling: A Comprehensive Approach. Inf. Syst. 2003, 28 (6), pp. 673–690

[29]Sturm A., Dori D., Shehory O. An Object-Process-Based Modeling Language for Multi-Agent Systems. IEEE Trans. Syst. Man Cybern. C. 2010, 40 (2) pp. 227–241

[30]Sturm A., Dori D., Shehory O. Application-Based Domain Analysis Approach and Its ObjectProcess Methodology Implementation. Int. J. Softw. Eng. Knowl. Eng. 2009 February, 19 p. 1

[31]Yaroker Y., Perelman V., Dori D. An OPM Conceptual Model-Based Executable Simulation

Environment: Implementation and Evaluation. Syst. Eng. 2013, 16 (4) pp. 381–390