《UML》教案

PowerDesigner 面向对象模型

[《实战项目训练》教案 1](#_Toc433289868)

[UML示例 1](#_Toc433289869)

[1. 面向对象模型 1](#_Toc433289870)

[2. 动态模型 19](#_Toc433289871)

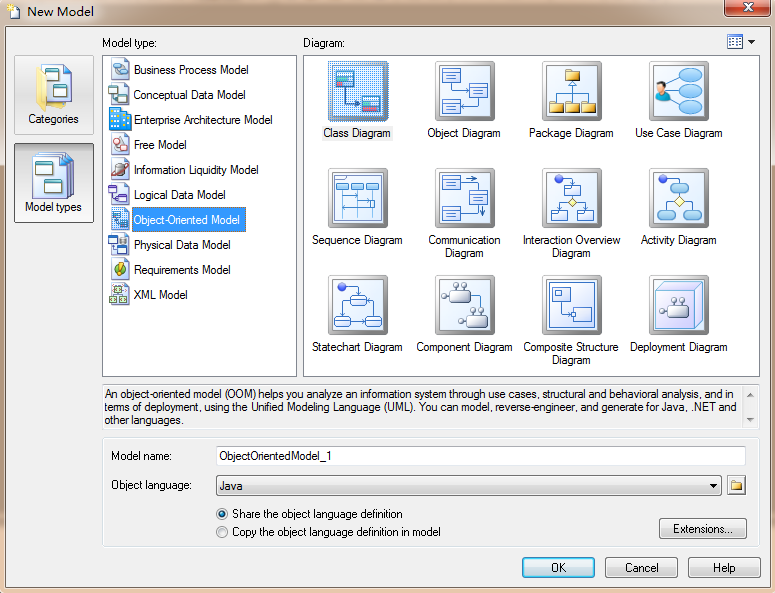
授课教师：李祥

课时：90分钟

# 面向对象模型

面向对象模型是利用UML（统一建模语言）的图形来描述系统结构的模型，它从不同角度实现系统的工作状态。这些图形有助于用户，管理人员，系统分析人员，开发人员，测试人员和其他人员之间进行信息交流。

1. 选择File-New Mode命令或者单击工具栏中的New Mode图标，弹出New Mode窗口
2. 在左侧是Mode type栏中选择Object-Oriented Model（面向对象模型）选项，在右侧输入模型名称，语言，如下图所示：



注：

静态图包括：用例图（Use Case Diagram）、类图（Class Diagram）、对象图（Object Diagram）、组件图（Component Diagram）和部署图（Deployment Diagram）。

动态图包括：状态图（Statechart Diagram）、顺序图（Sequence Diagram）、协作图（Collaboration Diagram）和活动图（Activity Diagram）。

1. Class Diagram：类图
2. Object Diagram：对象图

类图和对象图：用来定义类和对象以及它们的属性和操作。

1. User Case Diagram：用例图

主要用来描述系统的外部行为；

1. Sequence Diagram：顺序图

显示对象之间的动态合作关系，它强调对象之间消息发送的顺序，同时显示对象之间的交互。

1. Component Diagram：组件图
2. Deployment Diagram：部署图

组件图和部署图：描述实现系统的元素组织和系统环境元素的配置。

1. Statechart Diagram：状态图

描述类的对象所有可能的状态以及事件发生时状态的转移条件

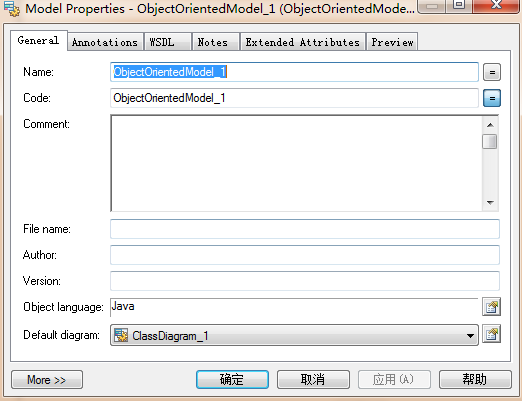
1. Activity Diagram：活动图

描述满足用例要求所要进行的活动以及活动间的约束关系，有利于识别并行活动。

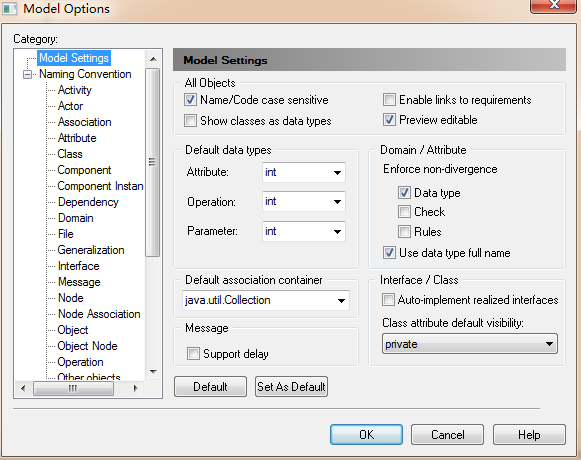
1. Collaboration Diagram：协作图

强调对象间的动态协作关系

1. Composite Structure Diagram：复合结构模型
2. 单击“确定”按钮，进入OOM设计工作区。在该工作区中可以定义OOM的属性和操作，也可以设计各种UML的图形。
3. 选择Model-Model Properties命令，显示如下图所示的OOM的Model Properties窗口



1. 定义完毕后单击“确定”按钮。然后选择File-Save As命令，以PubinfoSys.OOM这个名称将所定义的模型存盘。
2. 设置模型的选择。选择Tools-Model Options命令，打开Model Options（模型选项）窗口。如下图



1. 单击Set As Default按钮，设置为默认选项，然后单击OK按钮。

* 用例图

通常用来定义系统的高层次草图，它从用户角度描述了应用的系统功能，指出了各个功能的外部操作者。

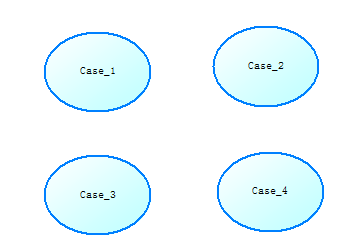
用于系统需求分析阶段，进行系统需求和功能设计，它包含执行者和用例两个要素。执行者指用户在系统中的角色，用例是用户与计算机的一次交互。用例图主要用来描述每个用例将有哪些执行者进行参与。

* 定义用例

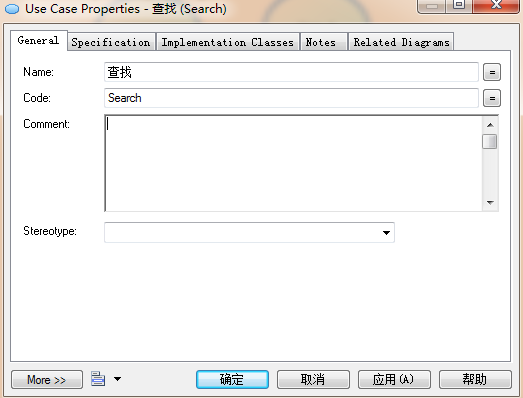
用例是外部可见的一个系统功能单元，这些功能由系统单元提供，通过一系列单元与一个或多个参与者之间交换的消息表达。

1. 定义用例图

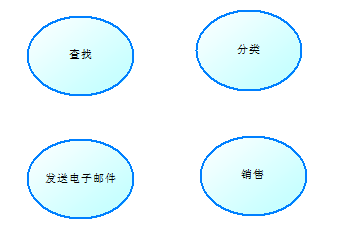
* 右键单击“新增”->Use Case Diagram
* 选择定义4个用例，如下图所示



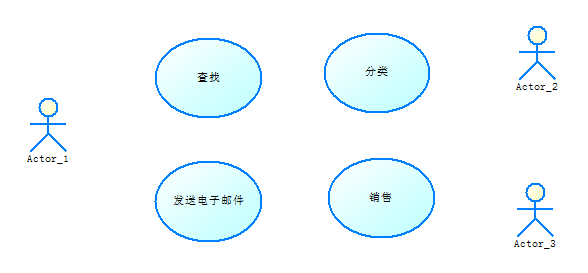
* 选中第一个用例双击，打开Use Case Properties（用例属性）窗口，将Name属性修改为“查找”，将Code属性修改为Search，如下图所示：



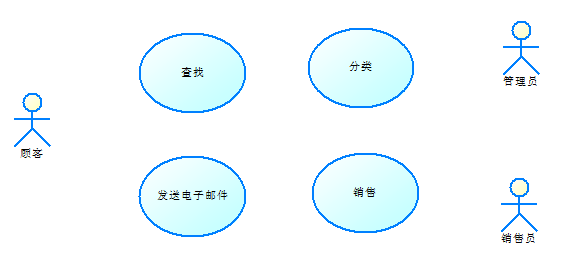
* 得用同样的方法，修改其他3个用例属性如图：



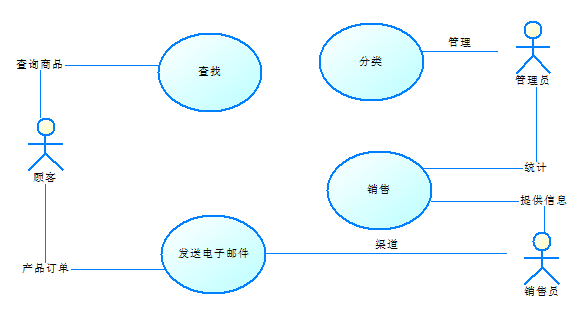
* 单击“执行者”图标定义3个执行者，如下图：



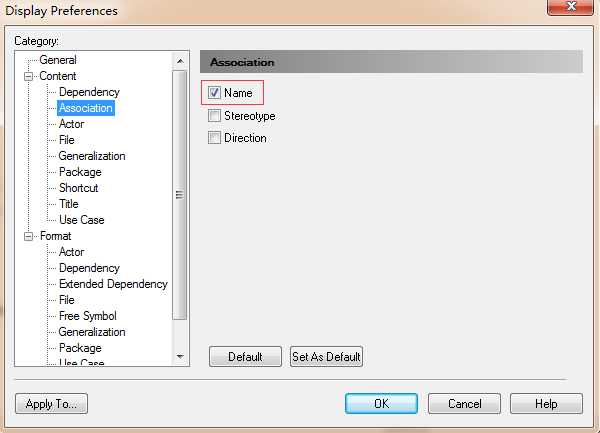
* 选中第一个执行者并双击，打开Actor Properties窗口，将Name属性修改为“顾客”，将Code属性修改为Buyer，如图：



* 利用同样的方法，修改其他两个执行者的属性
* 定义执行者和用例之间的关系。单击“关系”图标，再单击执行者“顾客”并手动鼠标至用例“查找”，释放鼠标，这时在执行者和用例之间建立了关系。双击该关系，打开Association Properties窗口，将Name属性修改为“查询商品”，将Code属性修改为SearchProduct。
* 按照上一步的方法，在其他执行者和用例之间建立关系。
* 定义完成后的用例图如下图：



注：如果关系上不显示注释，则需要修改显示参数设置，单击“工具（Tools）”->“显示参数设置（Display Preferences）”,把Name前的勾选上，如下图



* 定义角色

角色描述了与系统相互作用的一组用户，最明显的角色是系统中的人，但也可能是系统本身。

* 定义关联

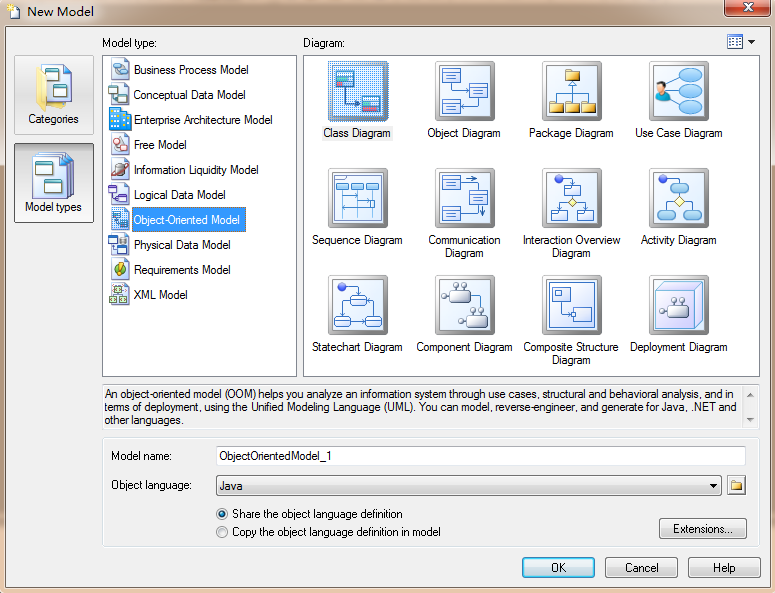
用例除了与其参与者发生关联外，还可以参与系统中的多个关系，允许定义的关联有关联，扩展，用例泛化，包括。

* 类图

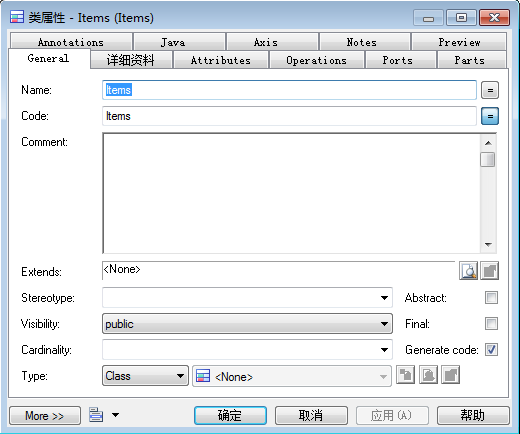
描述类与类之间的静态关系，并不能描述对象的所有细节。其中包括一系列的包，类，接口和它们之间的关系。类图是定义其他图的基础。

* 创建（Class）类

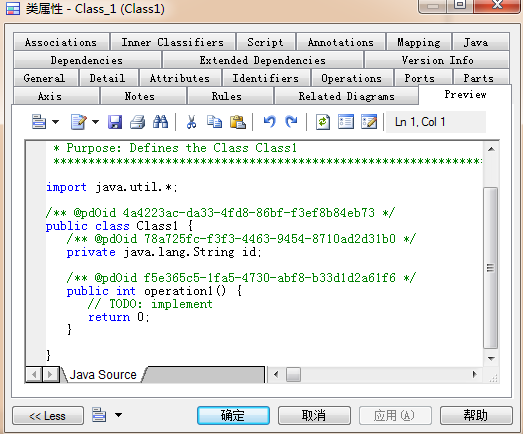
1. 选择File-New Model命令，打开New Model窗口。在窗口左侧的Model type栏选择Object-Oriented Model选项，在右侧的First diagram下拉列表框中选择Class Diagram选项，在Object language下拉框中选择Java选项，如下图：



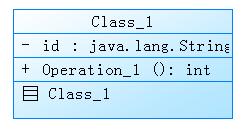
1. 在类图工作区中单击Palette工具栏中的Class图标，创建一个类，双击类图的图形符号，打开Class Properties窗口。



1. 选择Detail选项卡，可以定义类的更多属性。
2. 选择Attribute选项卡，可以为类增加属性。单击Add a Row图标，增加属性，也可以单击Add Attributes图标打开Selection窗口列表模型中其他类的属性，选择需要的属性，然后单击“OK”按钮即可。
3. 选择Operations选项卡，单击Add Operation图标，打开Selection窗口，从中可以选择该模型中其他类包含的操作。选择完毕后，单击“OK”按钮，所选的操作即可增加到当前类上。
4. 选择Inner Classifies选项卡，可以为该类定义内部类。内部类是PowerDesigner的一个分类器，在UML中分类器是一个包含有属性和操作等的模型元素。定义完毕后可以通过Preview选项卡观察当前类或接口所包含的内部类代码，如下图所示：



1. 除了以上属性外，还可以定义Associations，Identifies等属性。属性定义完毕后创建的内容如下图：



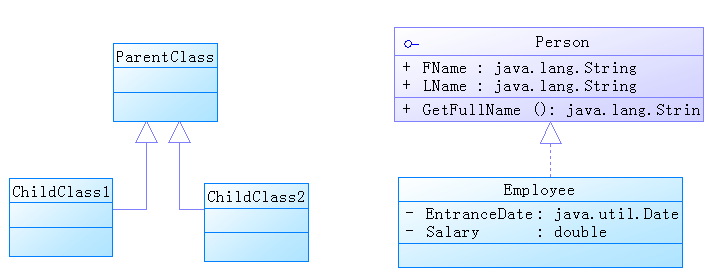
注：第四部分为创建的Inner Classifies（内部类）

* 创建联系（关系）

在类图中，联系有关联（association）、依赖（dependency）、泛化（generalization）和实现联系（realization）、聚合（aggregation）、成分（composition）、（Require Link）、（Inner Link）。

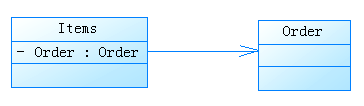
* Generalization：泛化、一般化，图标为

表示的是类与类之间的继承关系、接口与接口之间的继承关系、类与接口之间的实现关系。如果体现到java语言中，那就是反应extends和implements关键字。



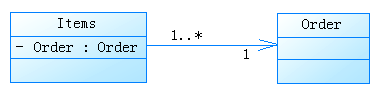
* Association：关联关系（一般关系）

关联关系描述的是类与类之间的连接，他表示一个类知道另一个类的属性和方法。关联关系可以是单向的或者双向的。在Java语言中，单向的关联关系是通过以实例变量的方式持有被关联对象的引用来实现的。一般来说是不建议使用双向的关联关系的。下面介绍单向的关联关系：

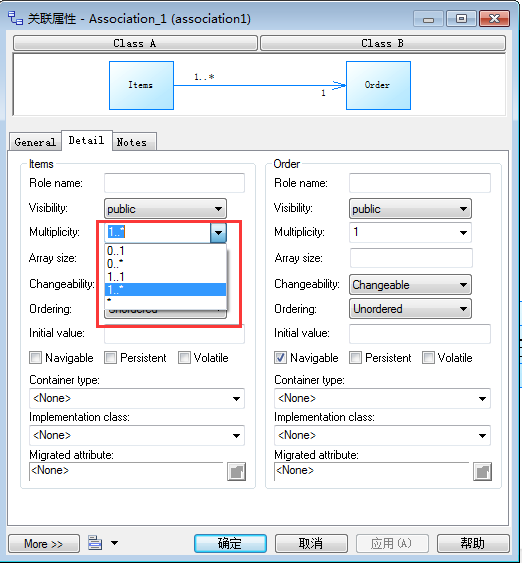


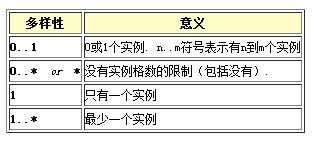
上面的类图表现的是订单和订单明细之间的关系。Items中有一个实例变量类型是Order。

每个连接都会有两个端点，上面的Items和Order就是端点，且每个端点都可以有（optional），一个基数（multiplicity），表示这个类可以有几个实例。这个类似于数据库中的1:n、m:n这些关系。我们可以给上面的例子加上基数：



上面表示的是订单与订单明细之间是1对多的关系。

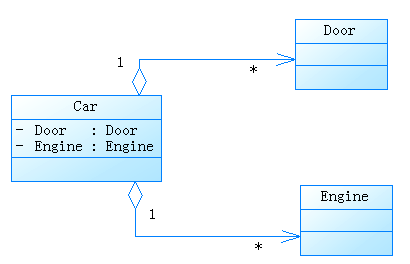




* Aggregation：聚合关系

聚合关系是关联关系的一部分，是非常强的关联关系。表示has-a的关系，是一种不稳定的包含关系。聚合关系表现的更多的是整体与部分的关系。较强于一般关联，有整体与局部的关系，并且没有整体，局部也可单独存在。如公司和员工的关系，公司包含员工，但如果公司倒闭，员工依然可以换公司。在类图中使用空心的菱形表示，菱形从局部指向整体。

例如汽车和车门、发动机之间的关系。如下图：



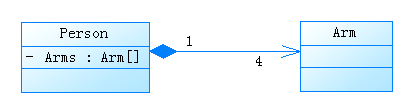
与关联关系一样，聚合关系也是通过实例变量实现的。单纯从语法的角度基本上无法判断出关联关系和聚合关系。

* Composition：组合关系（合成关系）

组合关系同样也是关联关系中的一种，这种关系是比聚合关系更加强的关系。我们前面提到，聚合关系表现的是整体与部分之间的关系，组合关系是在聚合关系的基础上，表示不可分割的整体与部分之间的关系。也就是说表示整体的对象需要负责表示部分的对象的生命周期。

“代表整体的对象负责保持代表部分的对象的存活，在一些情况下负责将代表部分的对象湮灭掉。代表整体的对象某些时候可以将代表部分的对象传递给另外一个对象，并由它负责代表部分的对象的生命周期。换言之，代表部分的对象同一时刻只能与一个对象构成组合关系。并且由后者排他的负责其生命周期。”――《Java与模式》

我们以人和手臂的关系举例，组合关系的类图如下：



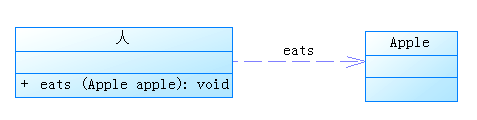
聚合和组合的区别：

这两个比较难理解，重点说一下。聚合和组合的区别在于：

1. 聚合关系是“has-a”关系，组合关系是“contains-a”关系；
2. 聚合关系表示整体与部分的关系比较弱，而组合比较强；
3. 聚合关系中代表部分事物的对象与代表聚合事物的对象的生存期无关，一旦删除了聚合对象不一定就删除了代表部分事物的对象。组合中一旦删除了组合对象，同时也就删除了代表部分事物的对象。

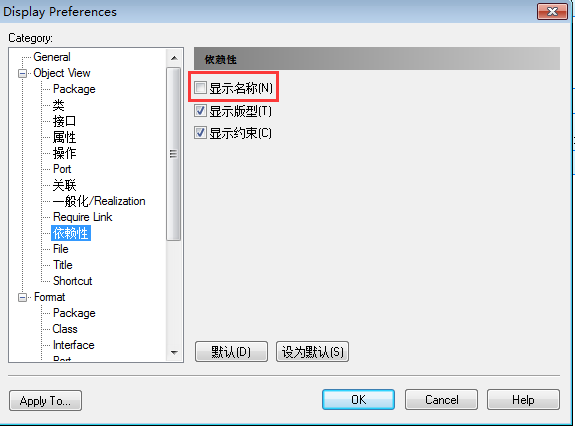
* Dependency：依赖关系

依赖关系表示一个类依赖于另一个类的定义。依赖关系是单方向的。人吃苹果，那么人依赖苹果。类图如下：



一般来说，被依赖的对象往往是以局部变量、方法参数的形式存在于对象中，与关联关系不同，它不会以成员变量的形式存在于对象中。这一点值得注意。另外，每个依赖都有一个名称。上面这个依赖关系的名称就是eats。

注：如果上面的依赖关系名称不显示出来，则单击菜单“工具”->显示参数设置，出现如下图所示，将“显示名称”前的勾选上。

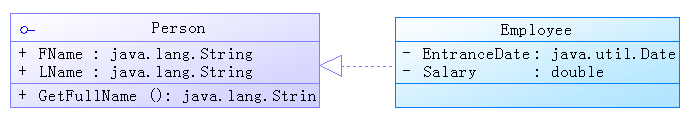


* 创建包

为了简单地表示出复杂的类图，可以把类组合成包packages。一个包是UML上有逻辑关系的元件的集合。下面这个图是一个把类组合成包的一个商业模型。Dependencies关系。如果另一个包B改变可能会导致一个包A改变，则包A依赖包B。

* 生成java的源代码

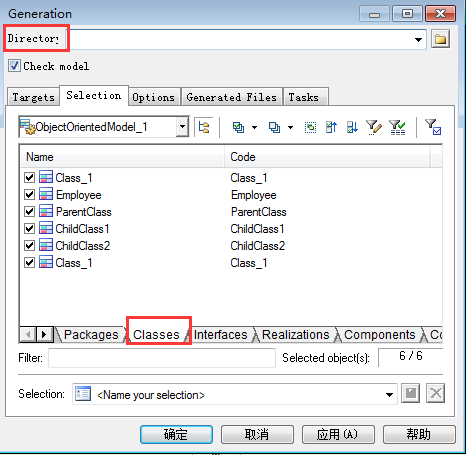
如下图所示类图，生成java源代码：



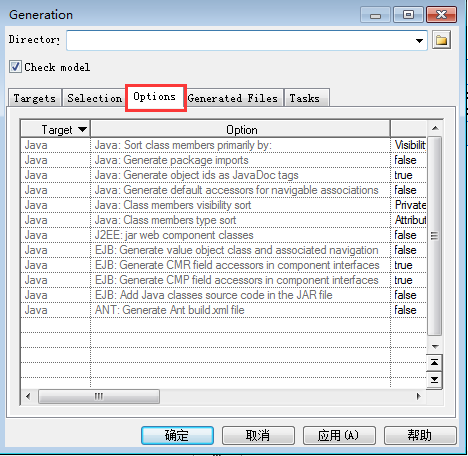
1. 在类图中选择Language-Generate Java Code命令，打开Generation窗口，显示模型中包含的类。
2. 在Selection选项卡的Classes选项卡中选择类，在Directory栏中输入Java源代码的目标路径。
3. 在Options选项卡中定义Java生成选项。
4. 在Task选项卡中定义Java生成任务选项，如下图所示：
5. 单击“确定”按钮，开始生成java源文件，生成完成显示Generated Files窗口。
6. 选择一个源文件，单击Edit按钮，将在文本编辑器中显示该文件的内容，如下图：

* 根据Java代码生成类图

1. 在类图中选择语言（Language）->Generate Java Code命令，打开Generation窗口，显示模型中包含的类。
2. 在Selection选项卡的Classes选项卡中选择类，在Directory栏中输入Java源代码的目标路径。勾选“check model”，选中后，在生成java代码时会有一定的语法检查，比如某类实现了某接口，但未实现接口的方法，它会提示。



1. 在Options选项卡中定义java生成选项。

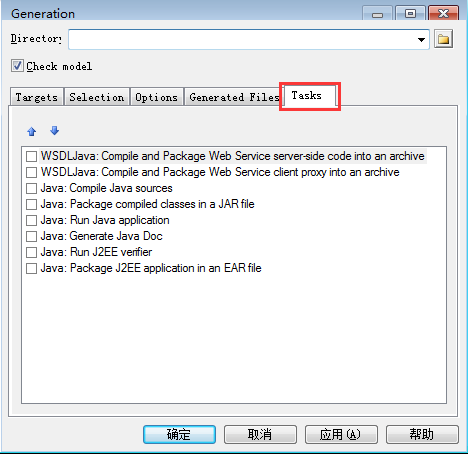


注：

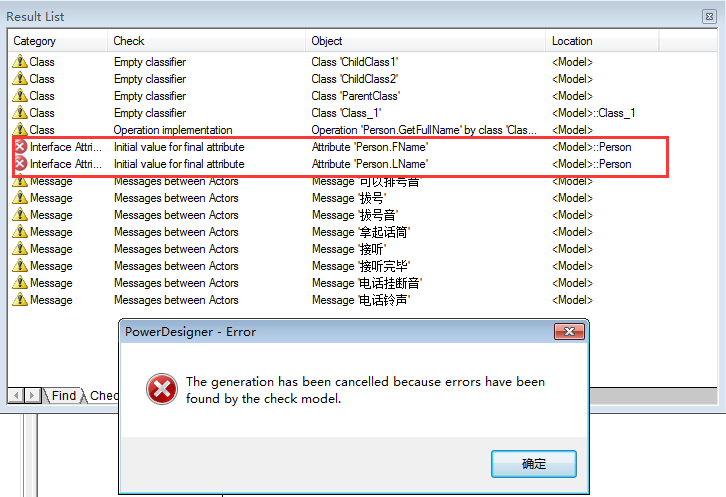
1. ANT：Generate Ant build xml file
2. Java：Generate object ids as JavaDoc tags

显示一些相关注释信息

1. 在Tasks选项卡中定义java生成任务选项，如下图所示：

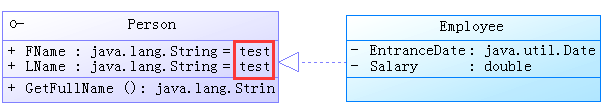


1. 单击“确定”按钮，开始生成java源文件，生成完成显示Generated Files窗口。

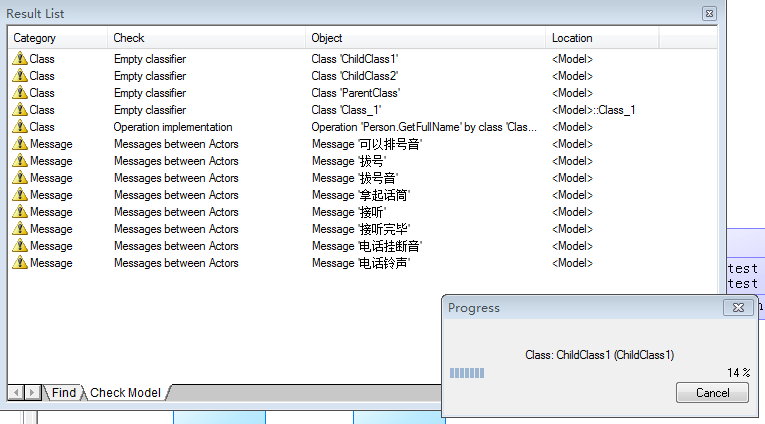


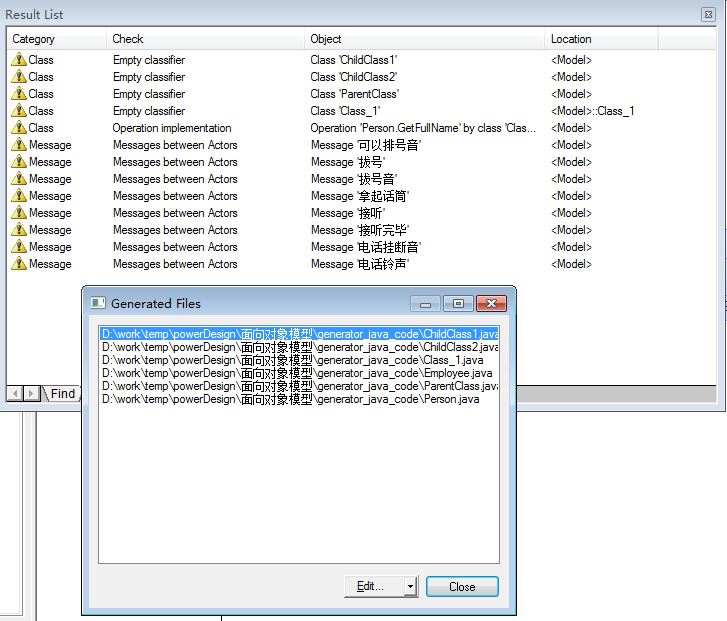
注：如果报以下错，请检查：

如果是接口里有属性是否有赋初始值，类似下图：

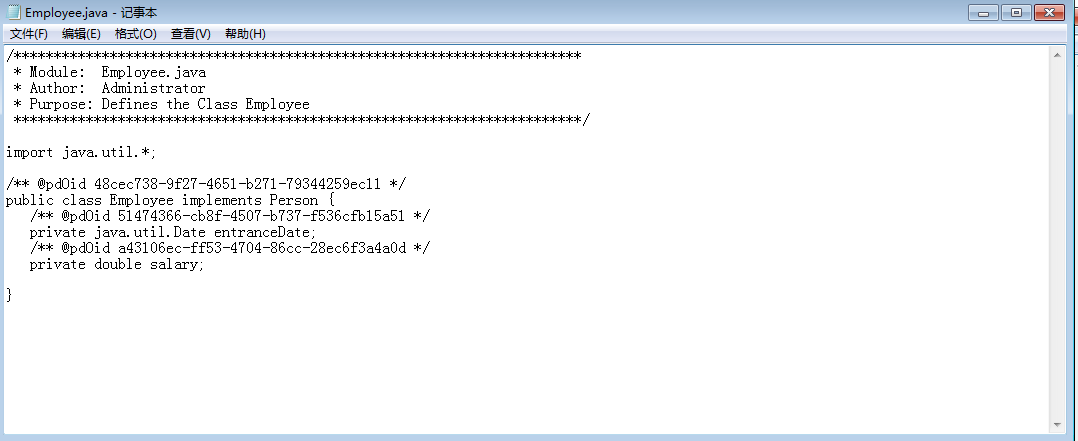


改正后，如下图所示：





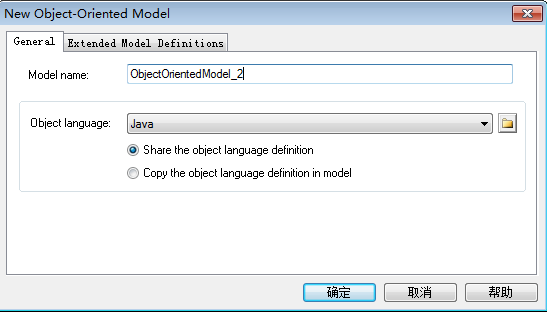
1. 选择一个源文件，单击“Edit”按钮，将在文本编辑器中显示该文件的内容。如下图所示：



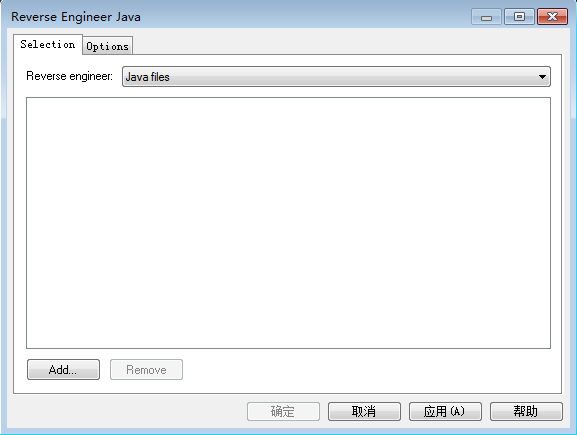
注：生成后查看代码文件，所有的属性都没有get和set方法，要想生成这两个方法，就要自己动手修改PowerDesigner的代码生成模板，可以选择Language菜单下的“Edit Current Object Language”，在弹出的窗口中修改代码模板：

* 根据java代码生成类图

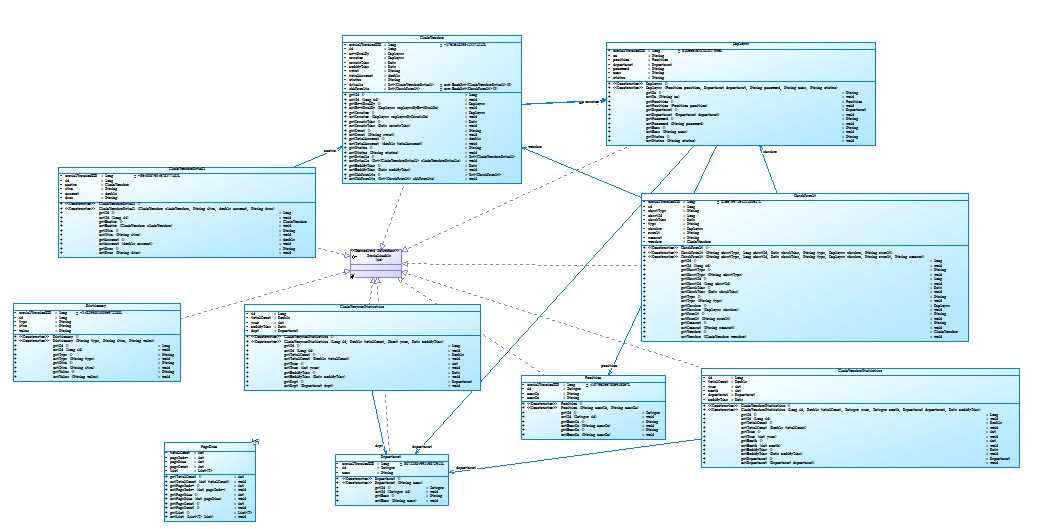
1. 选择File->Reverse Engineer-Object Lanaguage弹出如下图所示：



1. 在Object Language栏中选择java，单击“确定”按钮进入如下界面：



1. 点击Add，添加java文件，再单击“确定”按钮即可生成相应的java类图（在Reverse Engineer栏中还可以选择java字节码文件或文件目录），如下图所示：

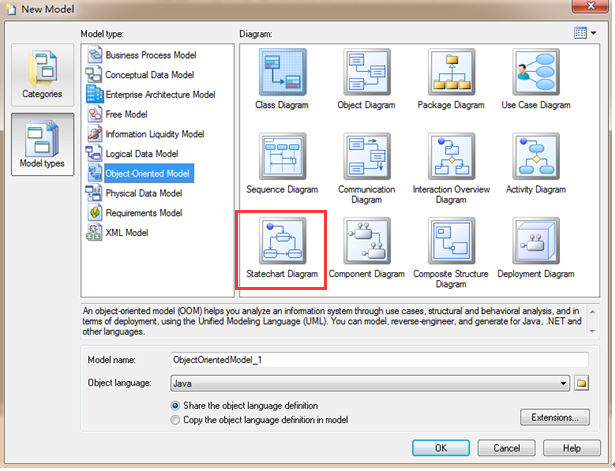


# 动态模型

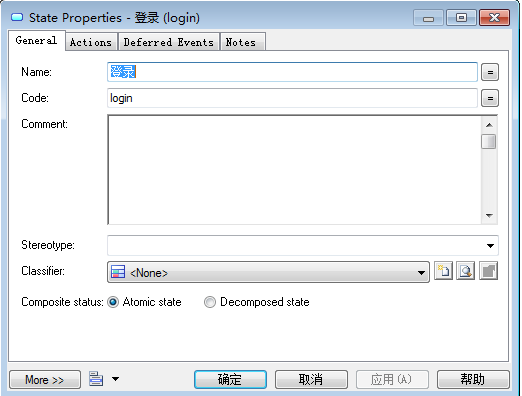
动态图包括：状态图（Statechart Diagram）、顺序图（Sequence Diagram）、协作图（Collaboration Diagram）和活动图（Activity Diagram）。

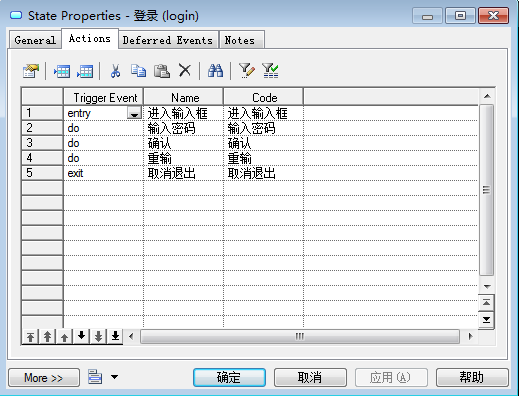
* 创建动态模型
* 状态图

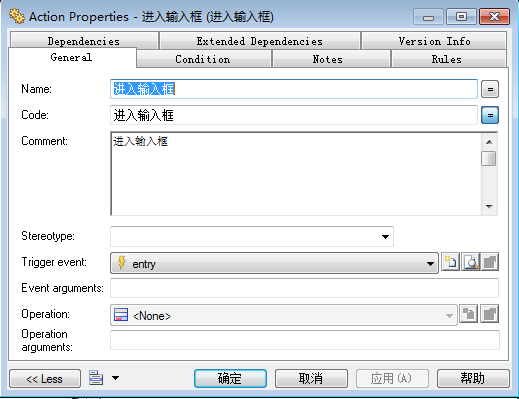
1. 选择File-New Model命令，打开New Model窗口。在窗口左侧的Model type栏选择Object-Oriented Model选项，在右侧的First diagram下拉列表框中选择Statechart Diagram选项，在Object language下拉框中选择Java选项，如下图:



1. 在状态图工作区中单击Palette工具栏中的State图标，双击状态图的图形符号，打开State Properties窗口



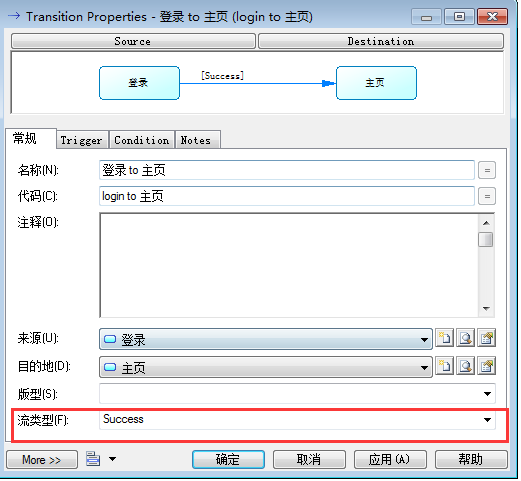




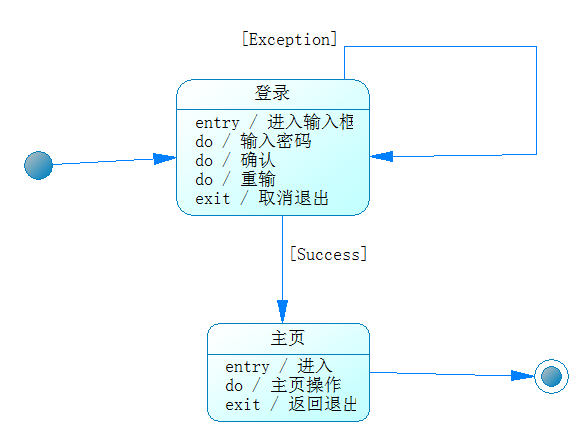
1. 在状态图工作区中单击Palette工具栏中的Transition图标，将各个状态之间连接起来，双击Transition连接线，打开Transition Properties窗口



注：当登录失败时再返回到登录。



1. 最后单击“确定”，如下图：



* 顺序图（序列图）

描述系统如何实际完成在User Case图中定义的功能。可以画出对象（类的实例）之间交互时产生的时序关系。它一方面描述了一次交互，交互中涉及类图中的类，另一方面细化了用例的描述。

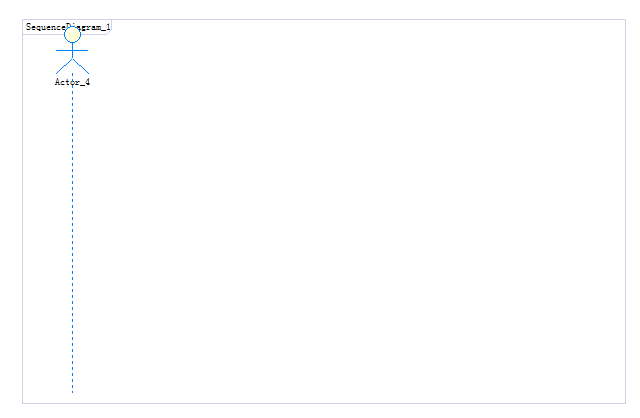
序列图有角色（Actor）、对象（Object）、消息（Message）和激活期（Activation）几个要素。

消息类型有：

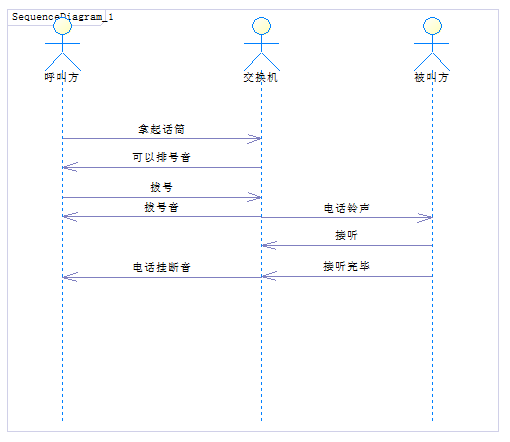
* Message：消息
* Self Message：递归消息
* Call Message：带有激活期的消息
* Self Call Message：带有激活期的递归消息
* Return Message：返回消息
* Self Return Message：递归返回消息

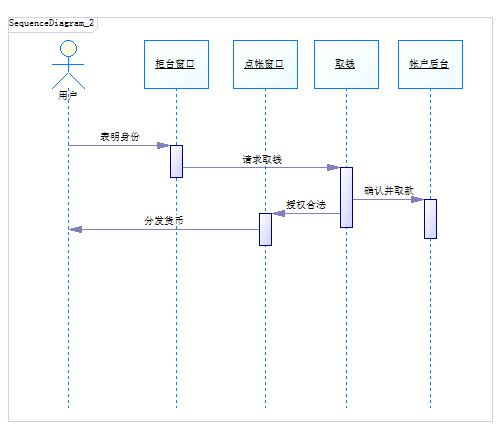
创建序列图：

1. 右键“新增”->“Sequence Diagram”
2. 单击Palette工具栏的Object图标，在空白区单击鼠标，出现对象符号如下图：



1. 在设计区空白区域右击，鼠标指针变为箭头形状，双击对象符号，弹出Actor Properties窗口，在Name文本框中输入“呼叫方”，在Code文本框中输入“Caller”，然后单击“确定”按钮。
2. 重复上面二个步骤（2）（3）添加其他两个对象，Name分别为“交换机”和“被叫方”，Code分别为Exchange和Receiver。
3. 单击Palette工具栏中的Message图标，在“呼叫方”对象下方的虚线处单击，拖动鼠标至“交换机”对象下方的虚线释放鼠标，即可增加一个消息。
4. 在设计区空白区域右击，鼠标指针变成箭头形状，双击对象符号，弹出“消息属性”窗口，在Name文本框中输入“拿起话筒”，在Code文本框中输入“LiftReceiver”，然后单击“确定”按钮。
5. 重复上面二步（5）（6），添加其他消息。
6. 最终的序列图如下图所示：



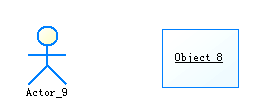


1. 选择File-Save命令，保存新建的面向对象模型。

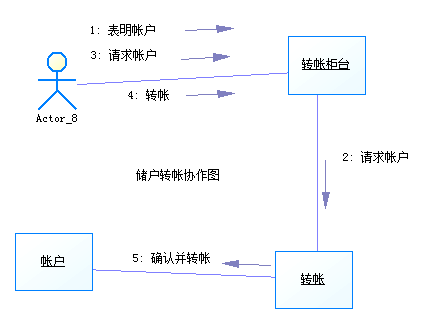
* 协作图（Collaboration Diagram、Communication Diagram）

按时空的顺序描述系统元素之间的交互和关系。

1. 右键“新增”->“Communication Diagram”
2. 单击Palette工具栏的Object图标和，在空白区单击鼠标，出现对象符号如下图：



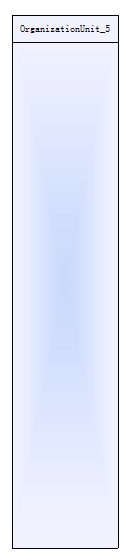
1. 单击图标连接操作人到对象。如下图效果：



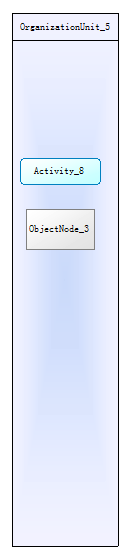
* 活动图（Activity Diagram）

描述系统元素的活动。

1. 右键“新增”->“Activity Diagram”
2. 单击机构图标（organization），如下图效果：



1. 单击Palette工具栏的行为图标和对象状态图标，在机构图标所绘区域单击鼠标，出现对象符号如下图：



1. 单击开始图标、结束图标、同步图标、转变图标、条件判断图标、文本图标，以及上面（2）（3）的图标绘制如下效果图：

