master tli tcp/dev/inet/tcp/x000207d1ca720401000000000000000

注释：参数间一定要用 TAB 隔开，IP 地址及端口以 16 进制表示：

retry—attempts：表示初次连接失败以后，还可试连的最多次数；

retry—interval：表示每两次试连之间应间隔的时间。

用户可以用 Sybase/install 下的 Sybinit 向 Interface 文件中添加一新的服务器入口，或修改已有服务器入口。亦可用编辑器手动编辑 Interface 文件（但这要求用户对 interface 文件的格式很熟悉，且修改前应作一备份）。

## 四　通讯测试及故障诊断

如果客户端、服务器端及其间的连接软件都正确安装，而 PowerBuild 客户应用无法打开 SQL Server 连接，则可借助于 Net—Library 提供的 Wdbping 检测工具。它能测试自客户端的 open client 通过网络到 SQL Server 的连接，并报告连接结果。

如果连接失败，用户可根据相应的返回信息，分析故障所在。

导致连接失败的常见原因有如下两种：

1)：网络自身不通或服务器尚未启动，或 Interface 中相应的服务器入口信息不正确。

此时 Wdbping 的反馈信息是：

Server is not responding

Loaded Netlib. DLL Dynamic Link Library

2)：在客户端的系统 PATH 中未设置驱动路径。

此时 Wdbping 的反馈信息是：

Can not load Netlib. DLL. Dynamic Link Library

一般而言，Wdbping 试成功时，PowerBuilder 客户应用即可成功访问 Sybase SQL Server 的 Sybase 数据库。

电脑与信息技术　1996 年第 5 期

# 面向对象的概念建模

厦门大学　王周敬

**摘　要**　概念建模的目的是建立开发者与用户之间通信所需的概念知识。概念模型的图形表示往往是对现实世界的最高层抽象，它使开发者与用户通信更方便，使模型更容易理解。本文给出一种面向对象概念建模方法，它将图形表示、E—R 方法、关系演算、时态逻辑溶为一体，支持对象结构、行为和对象演变建模的一体化和封装。

**关键词**　概念模型　面向对象　对象类型

## 一　引言

信息系统的概念建模是面向应用，而不是面向计算机的。传统的概念建模是面向数据或过程。面向数据的概念模型多数是基于 E—R 模型，它首先对数据进行建模，而后才对数据的操作进行构造；而面向过程的概念建模，是先对信息处理活动建模，而后利用数据字典来定义数据的结构。由于面向对象的模型要描述的对象比 E—R 模型中的关系和数据字典要复杂得多，因此传统的概念模型无法反映面向对象的模型，主要表现是：(1)无法描述复杂的结构化信息；(2)无法描述对象的抽象继承；(3)无法描述对象的行为和约

束信息；(4)无法描述对象的封装。

文献[1]提出了三种对系统进行描述的模型：对象模型、动态模型和功能模型；文献[2]提出了用对象图描述动态模型的状态变化，用前置条件和后置条件来加强对功能模型中的操作的行为约束，但它们存在需要多种模型支持及形式化分析难以进行等问题。

## 二 概念模型

对象对应于现实世界中的实体，系统中存在许多对象类型，它们之间存在一定联系和约束。对象类型联系图 ORD 是描述系统的对象类型的结构信息以及对象类型间的联系和约束，它在 E—R 方法基础上，增加描述现实世界复杂对象、对象类型的抽象继承和对象类型的约束信息等功能。ORD 构造规划如下：

• 每个对象类型可能有多个属性，每个属性有相应的类型，该类型可以是原子类型，也可以是对象类型；

• 每个对象类型可以有唯一的对象标识，对象标识是属性集的子集；

• 每个属性可以有若干谓词公式，它描述了对象的静态约束信息；

• 每个联系型的对象类型可以有若干属性，并可联接一个或多个对象类型；

• 对象类型之间可以有 1：1、1：n、m：1、m：n、ds、gs、ps 联系，其中 ds、gs、ps 分别是不相交子类、派生子类、部分子类(形式定义见文献[3])。

图 1 给出了订货系统的订货单 order 对象类型、产品 product 对象类型、客户 customer 对象类型以及 order 的子类新订货单 new—order 对象类型、已处理订货单 handled—order 对象类型和过期订货单 back—order 对象类型的 ORD。

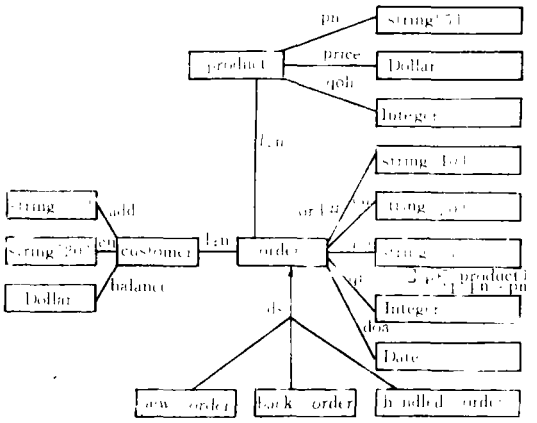


图 1 简化订货系统的 ORD

ORD 描述了对象类型的逻辑结构及其相互联系，但作为概念模型，它还不够完全，概念模型还需要描述对象操作的行为及其约束信息。为此，我们引入对象操作图 OOD，OOD 构造规划如下：

• 每个操作有一个操作名；

• 每个操作可以有若干参数，每个参数具有相应的类型；

• 每个操作可以有若干对象类型的输入对象；

• 每个操作可以有若干对象类型的输出对象；

• 每个操作可以有若干时态逻辑公式，它描述操作发生的时间约束；

• 每个操作可以有若干行为规则，它描

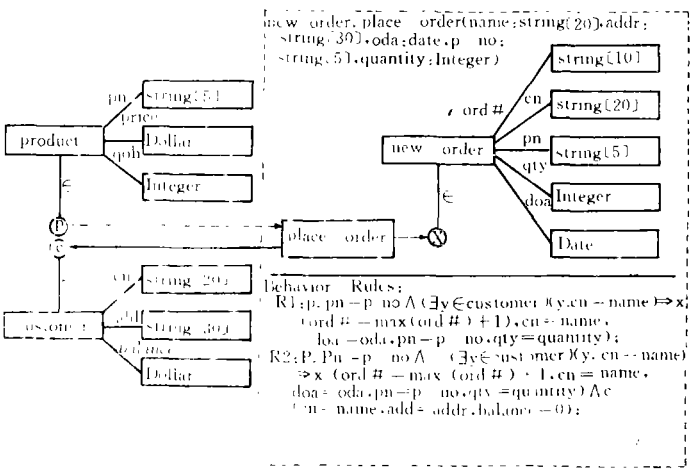


图 2 new__order.place__order 的 OOD

述操作的行为约束，行为规则包含在输入对象集上的前置条件和在输出对象集上的后置条件二个部分。

图 2 给出了对象类型 new __ order 的操作 place __ order 的 OOD。操作 place __ order 的参数部分、行为规则部分以及操作对应的对象类型结构信息被封装在一个大的虚框内，而与操作相关的其它对象类型表示在虚框外。它指出操作 place __ order 对 product 的对象是只读的，当客户订货是本企业的产品且是老客户时，操作产生一新的订货单对象信息；当客户订货是本企业的产品且该客户是新客户时，操作将产生一新订货单对象和该客户对象信息。操作行为规则中，前置条件限制客户订货产品是企业的产品(p. pn＝p __ no)是重要的，它保证了子类与超类的属性约束一致性，即保证了超类 order 中(∃ p∈product)(p. pn＝pn)的约束成立。

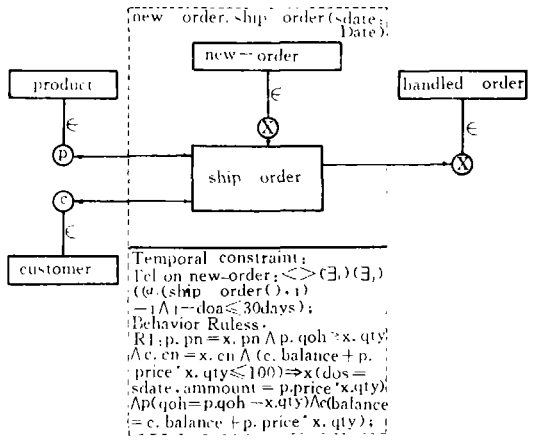


图 3　new __ order. ship __ order 的 OOD

图 3 的 new __ order. ship __ order 的 OOD 给出了对象的删除、修改、只读以及产生的表示方法，它同样给出了对象状态变化的表示方法；一个输入对象可以从某对象类中移出，经过修改后，把它放到另一个对象类中。时态约束指出了客户订货将在 30 天内得到办理，其中@是操作(事件)第 i 次发生的时间(日期)函数。

对象类型的 ORD 图和所有对应的 OOD 构成了对象类型的概念模型，对象类型 order，new __ order 的部分文本规范描述如下：

```
Object Type order;
  Identifier ord#:string[10],{订单号}
  Attribute cn:string[20]{客户名}
            pn:string,[5](∃ p∈product)(p. pn=pn);
{产品号}
            qty:Integer,{订货量}
            doa:Date;{订货日期}
  Subtype new __ order, back __ order, handled __ order:
ds;
  ...
End object Type;
ObjectType new __ order;
  operation place __ order(name:string[20],addr:string
[30],oda:Date,
          p __ no:string[5],quantity:integer);
    object referenced p∈product, produced x∈new __ or-
der,c∈customer;
    Behavior Rules:
      R1:p. pn=p __ no ∧ (∃ y∈customer)(y. name=
name)=>x(ord#=max(ord#)+1,cn=name,doa=
oda,pn=p __ no,qty=quantity);
      R2:p. pn=pn __ no ∧ ¬ (∃ y∈customer)(y. name
=name)=>x(ord#=max(ord#)+1,cn=name,
        doa=oda,pn=p __ no,qty=quantity) ∧ c(cn=
name,add=addr,balance=0);
  End operation;
  operationship __ order(sdate:date);
    object updated p∈product, c∈customer,consumed x
∈new __ order,
          produced x∈handled __ order;
    Temporal constraint:
      Tc1 on new __ order:<>(∃ i)(∃ j)(@(ship __ order
(),i)=j∧j __ doa≤30day);
      Behavior Rules:
        R1:p. pn=x. pn ∧ p. qoh≥x. qty ∧ c. cn=x. cn ∧
(c. balance+p. price * x. qty≤100)
          =>x(dos=sdate,ammount=p. price * qty) ∧
p(qoh=p. qoh __ x. qty)
            ∧ c(balance=c. balance+p. price * x. qty);
  End Operation;
End Object Type;
```

## 三　概念建模方法

概念建模包括以下几个步骤：

1. 识别问题域的对象及其联系，建立如图 1 所示的 ORD。

2. 对每一个对象类，识别其上的操作及其操作的时态约束和行为规则，建立操作的OOD。如果操作太过复杂，把它分解成低一层操作。

3. 对每一个操作需检验：(1)操作的前置条件是否与对象的静态约束相一致；(2)操作只能产生合法的对象，操作的结果不能与输出对象类型的静态约束相违；(3)操作能反映需求。

4. 验证每个对象类型的模型是否满足用户需求，如果不满足，是否要进行对象分解。验证工作需要自动快速地生成一个原型来支持；将对象类型的ORD和OOD转换成对象类型的文本规范描述，然后，将对象类型的结构信息转换成关系模式，将抽象继承关系转换成一组关系模式，使得子类的关系模式包含超类关系模式的属性和关键字；将操作转换成主语言的函数或过程。这时，前置条件被转换成循环条件或选择条件，输出被转换成append/insert和replace/update语句，而对象的consumed被转换成删除语句。

由于需求的变化等原因，如对于订货系统，要求只能对老客户或已预付款的新客户办理订货业务，并且要求对未预付货款或预付不足的订货新客户去函，则需要对对象进行分解，以适应新的要求，其步骤如下：

①识别受需求变化影响的对象类型。订货系统受影响的对象类型是new __order。

②修改受影响对象类型的结构，包括对象类型的属性、继承关系等。对于new __order，增加一个属性prepay表示订货预付款情况，将new __order分解成二个子类：可接收办理的订货单accepted和有效订货单valid。

③对修改后或新的对象类型分析并修改其对象的静态约束。

④对修改后或新的对象类型分析并修改其操作，包括操作的参数、输入/输出对象、时态约束和行为规则。

new __order分解后，操作place __order的变化见图4。

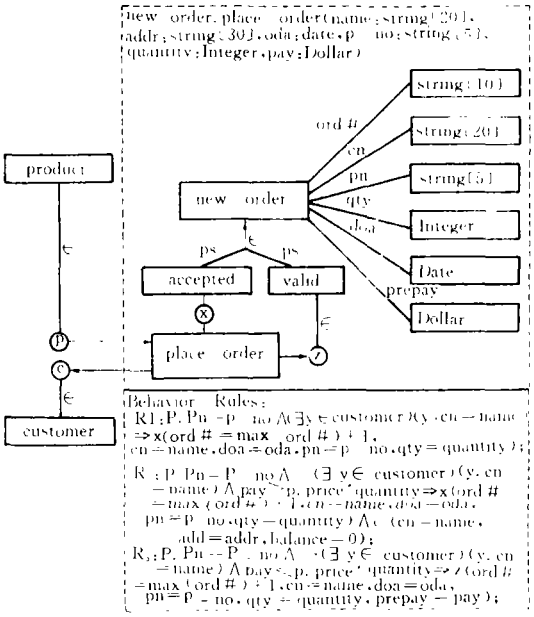


图4 对象分解后new __order.place order的OOD

## 四 结束语

面向对象概念模型具有描述现实世界的结构、行为、继承和封装等特征，它支持对象类型、对象、对象演变和对象间联系的建模。文中建模方法具有如下特点：(1)建模具有图形表示，因此模型直观且易理解；(2)对象类型的结构、行为和联系可并存在同一个模型里；(3)它允许分析员在某一时间里集中对对象的操作建模，这将大大减少概念建模的复杂性；(4)具有数学基础，它支持静态和时态约束建模，支持模型特性验证的形式化分析；(5)模型易于快速产生原型。

### 参考文献

[1]Rumbaugh,J.,et al.,Object－Oriented Modeling and Design, Prentice－Hall,1991;

[2]Hayes,F.,and Coleman, D., Coherent models for object－oriented analysis,in Proceeding of 1991 OOPSLA Conference, 1991;pp. 171－183.

[3]Kung,C.,Object Subclass Hierarchy in SQL:A Simple Approach,Commun. ACM 33,117－125(1990).

[4]侯正风，周国祥．面向对象数据库系统及其设计方法．微型计算机，1996;(1),pp:7—9.