第 26 卷第 11 期
2004 年 11 月

电 子 与 信 息 学 报
Journal of Electronics & Information Technology

Vol.26No.11
Nov. 2004

CORBA 规范的形式化描述及分析 [1]

郑 红　　李师贤 *

(华东理工大学计算机科学与工程系　上海 200237)
*(中山大学计算机科学系　广州 510275)

摘　要：CORBA 为构建大规模分布式应用程序提供了一套公共对象服务规范，其规范主要以 IDL 语言编写，只提供对象的静态行为描述. 提出并应用扩展有色 Petri 网对 CORBA 对象进行形式化模拟和分析，结果表明扩展有色 Petri 网模型不仅能够清楚描述对象的静态行为，同时也能较好地模拟对象分布性和动态行为.

关键词：CORBA, CORBA 规范，Petri 网

中图分类号：TP31　　**文献标识码**：A　　**文章编号**：1009-5896(2004)11-1830-07

Formal Description and Analysis on CORBA Specifications

Zheng Hong　　Li Shi-xian*

(*Dept of Computer Science and Engineering,*
East China University of Science and Tech., Shanghai 200237, *China*)

*(*Dept of Computer Science, Zhongshan University, Guangzhou* 510275, *China*)

Abstract　CORBA has provided a set of Common Object Services (COS), which help users to build large-scale distributed applications, but Common Object Services Specifications (COSS) do not include integrated formal description. Petri nets are a powerful instrument for modeling, analyzing, and simulating dynamic systems with concurrent and non-deterministic behavior. An extended colored Petri net is introduced to express the behaviors of individual objects, the concurrency between different objects as well as the intra-object concurrency in the context of CORBA, and gives an example for formal description of CORBA objects.

Key words　CORBA, CORBA specification, Petri nets

1 引言

CORBA(Common Object Request Broker Architecture) 是国际对象管理组织 OMG (Object Management Group) 定义的一套面向对象中间件技术标准，OMG 提供的 CORBA 公共对象服务规范 (Common Object Serves Specification, COSS)，为用户建立分布式大规模系统提供了标准解决方案 [1−3]。

CORBA 规范使用接口定义语言 (Interface Defined Language, IDL) 描述 CORBA 系统中各个组件的基本情况及组件在系统中承担的任务，IDL 不是编程语言，它没有控制结构. COSS 主要是由 OMG 以 IDL 定义和文本方式描述，缺乏严格的形式化对象动态语义，CORBA 规范并没有从功能实现的细节上规定如何实现 CORBA 服务，对于基于 CORBA 的软件开发而言，

[1] 2003-01-29 收到，2003-07-22 改回
国家自然科学基金委员会与香港研究资助局联合科研资助基金项目 (79910161989)，教育部科学技术研究重点项目 (01077) 和教育部优秀青年资助计划 (EYTP) 资助课题

程序设计人员可以自由地选择实现方式，因而 CORBA 服务实现具有一定的灵活性。但在某些场合下，这样的灵活性可能造成实现者理解上的二义性。

目前，在对象系统行为语义方面已做了较深入地研究 [4−10]，提出了一些用于异构分布式对象系统的行为规范形式化方法。但若把它们真正应用到实际系统中，在具体实现上还存在一些问题，如 π 演算 [4] 描述语义很抽象且较难理解；Z 语言 [5] 描述异构系统并发问题也存在一定的困难；进程代数方法 [6] 过于抽象，不适合描述基于 CORBA 的应用等。为此，本文提出一种扩展有色 Petri 网，并将它作为形式化工具用于 CORBA 接口描述。用扩展有色 Petri 网建模 CORBA 对象接口，可使用已有的 Petri 网分析技术进行模拟、验证。结果表明，我们提出的方法对于 CORBA 服务规范，具有图形动态模拟和形式化描述分析功能，且易于实现。

2 扩展有色 Petri 网

本节基于有色 Petri 网 (CPN)，同时引入测试弧和抑制弧，以扩展有色 Petri 网，并对扩展的有色 Petri 网 (Extended Colored Petri Net, ECPN) 模型进行分析。有关有色 Petri 网的基本概念和术语，请参阅文献 [11]。

2.1 扩展的有色 Petri 网的形式定义

ECPN 由一个静态文法描述 (Staticgrammer gramma Description, SD) 和面向对象有色 Petri 网构成，其中文法描述由一组类似 IDL 定义的文法组成。

定义 1　ECPN 是一个二元组，ECPN=(SD,CPN)，其中

SD 是网系统接口的文法描述及库所和变迁定义；

CPN=$(\Sigma, P, T, A, N, C, G, E, I)$ 是 ECPN，应满足如下条件：

(1) Σ 是一个非空的有限颜色集；

(2) P 是一个有限库所集；

(3) T 是一个有限变迁集；

(4) A 是一个有限弧集，$A = A_1 \cup A_2 \cup A_3$，$A_1$，$A_2$ 和 A_3 两两互不相交，其中：A_1 是一个普通有色网弧集；A_2 是一个测试弧集；A_3 是一个抑制弧集；

(5) N 是一个结点函数，N：$A \to P \times T \cup T \times P$；$P, T$; $\boldsymbol{N}(A)$ 是有向网，$\boldsymbol{N}(A) = \{N(a)|a \in A\}$；

(6) C 是一个颜色函数，C：$P \to \Sigma$；

(7) G 是一个识别函数，G：$T \to$Exprs，且满足：$\forall t \in T$：$\mathrm{Type}(G(t)) = B \wedge \mathrm{Type}(\mathrm{Var}(G(t))) \subseteq \Sigma$；其中 B 是布尔类型值 true 或 false，Type( ) 是变量或表达式的类型函数，Var( ) 表示表达式的变量集；

(8) E 是一个弧表达式函数，E：$A \to$ Exprs，且满足：$\forall a \in A$：$\mathrm{Type}(E(a)) = C(p(a))_{\mathrm{ms}} \wedge \mathrm{Type}(\mathrm{Var}(E(a))) \subseteq \Sigma$；

(9) 其中 $p(a)$ 表示节点函数 $N(a)$ 中的库所，$C(p)_{\mathrm{ms}}$ 是表示 $C(p)$ 的多重集；

(10) I 是一个初始化函数，I：$P \to$ Exprs，且满足：$\forall p \in P$: $\mathrm{Type}(I(p)) = C(p)_{ms}$，其中 Exprs 是一个封闭表达式，即表达式中不含有变量。

在下面的定义 2 中，颜色集决定了可用于 CPN 网标注中的托肯 (token) 类型、运算和函数，例如弧表达式、识别、初始化表达式等。

弧集 A 又分为 3 个互不相交的子集 A_1, A_2 和 A_3，这里 A_1 是一般的有色网弧集，A_2 是测试弧集，若 $(p,t) \in A_2$，则当 $E(p,t) \in M(p)$ 时，t 才可以使能，且在变迁 t 引发后，p 中的托肯保持不变。图中，测试弧是用有向弧前加一粗线杠表示 (见图 1)。A_3 是抑制弧集，且在图表示中，将有向弧的箭头用一个小圆圈代替 (见图 1)。若 $(p,t) \in A_3$，则当 $E(p,t) < b > \notin M(p)$ 时，t 才可以使能，即当 $M(p)$ 中包含的托肯有绑定值时，t 不可使能。节点函数把每条弧映射

到一个二元组，第一个为原节点，第二个为目标节点，且这两个节点必须属于不同的类型，即(变迁 T，库所 P) 或 (库所 P，变迁 T)。ECPN 允许一对有序点之间存在多条弧，因此，弧 A 为一个独立集合。

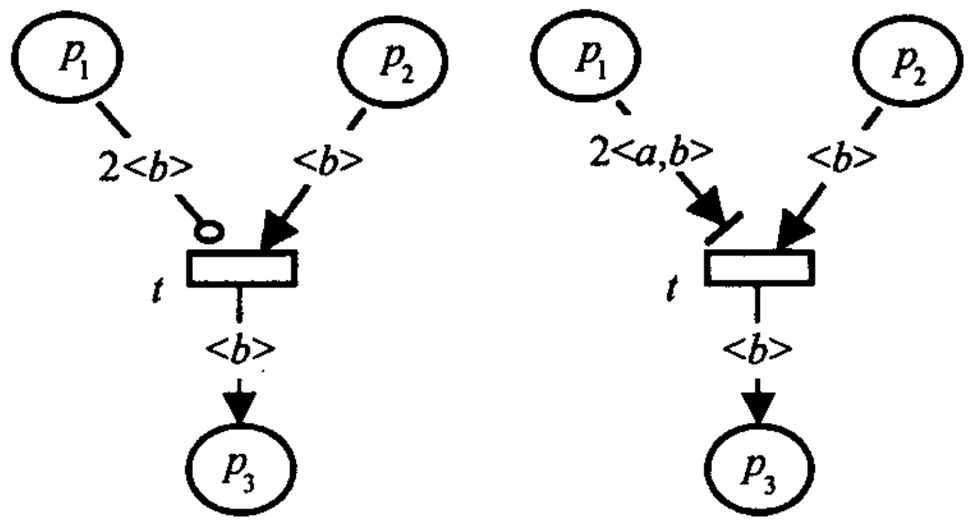

图 1 抑制弧和测试弧应用示例

颜色函数 C 把每个库所 p 都映射到一个颜色集 $C(p)$，也就是说 p 中的每个托肯必须属于颜色类型集 $C(p)$。

识别函数 G 把每个变迁 t 都映射到一个布尔表达式 B，并用一组布尔表达式 $[\mathrm{Bexpr}_1, \mathrm{Bexpr}_2, \cdots, \mathrm{Bexpr}_n]$ 作为识别函数，等价于 $\mathrm{Bexpr}_1 \wedge \mathrm{Bexpr}_2 \wedge \cdots \wedge \mathrm{Bexpr}_n$。当识别函数为空时，表示其值为真。

弧表达式函数 E 把每条弧 a 都映射到一个具有类型 $C(p(a))_{\mathrm{ms}}$ 的表达式。这意味着 $E(a)$ 的每次求值都生成一个依赖于相邻位置颜色集的多重集合。

初始化函数把每个位置 p 映射为不包含变量的表达式即封闭表达式，其类型为 $C(p(a))_{\mathrm{ms}}$。与前面类似，初始化表达式的类型也能为 $C(p)$ 或取空。

2.2 ECPN 的动态行为

我们用 $\mathrm{Var}(t)$ 表示 t 的变量集，$E(x_1, x_2)$ 表示 (x_1, x_2) 的表达式：

$$\forall t \in T : [\mathrm{Var}(t) = \{v | v \in \mathrm{Var}(G(t)) \vee \exists a \in A(t) : v \in \mathrm{Var}(E(a))\}]$$

$$\forall (x_1, x_2) \quad \left[E(x_1, x_2) = \sum_{a \in A(x_1, x_2)} E(a) \right] \in (P \times T) \bigcup (T \times P)$$

定义 2 一个变迁 t 的绑定是定义在 $\mathrm{Var}(t)$ 上的函数 b，且满足：

$$\forall v \in \mathrm{Var}(t) : b(v) \in \mathrm{Var}(t); G(t) < b >$$

变迁 t 的绑定就是把变迁 t 的每个变量 $\mathrm{Var}(t)$ 用一种合适的颜色代替，且识别函数 $G(t)$ 在绑定 b 必须为真，即 $G(t) < b >= \mathrm{true}$。通常，绑定写成 $< v_1 = c_1, v_2 = c_2, \cdots, v_n = c_n >$，其中 $\mathrm{Var}(t) = \{v_1, v_2, \cdots, v_n\}$。变迁 t 的所有绑定的集合记为 $B(t)$。

定义 3 一个托肯是一个二元组 (p, c)，其中 $p \in P$, $c \in C(p)$；一个绑定是一个二元组 (t, b), 其中 $t \in T$, $b \in B(t)$，所有托肯构成的集合用 TE 表示；所有绑定构成的集合用 BE 表示。因此，一个标识是定义在 TE 上的一个多重集合，而一个引发步 (firing step)Y 则是定义在 BE 上的一个非空有限多重集合。即标识是由托肯构成的，而引发步则是由绑定构成的。ECPN 网的初始标识 M_0 是对初始化表达式求值而得到的：$\forall (p, c) \in \mathrm{TE}$：$M_0(P, c) = (I(P))(c)$。

对于每个标识 $M \in \mathrm{TE}_{\mathrm{ms}}$，都唯一地决定了一个 M^*，M^* 定义在 P 上，满足 $M^*(p) \in C(p)_{\mathrm{ms}}$，使得：$\forall_p \in P$，$\forall_c \in C(p) : (M^*(P))(c) = M(p, c)$; 而对于每一个定义在 P 上

的函数 M^*，对所有 $p \in P$，满足 $M^*(p) \in C(p)_{\text{ms}}$，就唯一地确定了一个标识 M，使得：$\forall (p,c) \in \text{TE}$：$M(P,c) = (M^*(P))(c)$．由此，对于标识 M 以及相应的函数 $Y^*(t) \in B(t)_{\text{ms}}$，是有限非空的。

定义 4　在标识 M 下，一个引发步 Y 是使能的，当且仅当对 $\forall p \in P$：

$$(1) \qquad \sum_{(t,b)\in Y \wedge (p,t)\in A_1 Y A_2} E(p,t) < b > \leq M(p)$$

$$(2) \qquad \forall (t,b) \in Y \wedge (p,t) \in A_3 : E(p,t) < b > \notin M(p)$$

若 Y 在 M 下是使能的，则存在 $(t,b) \in Y$，那么称变迁 t 在标记 M 下对绑定 b 是使能的，这时也认为 t 是使能的。若引发步 Y 中的绑定的个数大于 1，则这些元素是并发使能的。若 $|Y(t)| \geq 2$, 这时变迁 t 本身是并发使能的。

当一个引发步使能时，它可以引发，即将托肯从输入库所中移去，并加入到输出库所中，引发涉及到的托肯个数和颜色由弧表达式决定，而表达式的值则通过引发时的绑定求得。

定义 5　当一个引发步 Y 在标识 M_1 下使能时，它可以引发，把标识 M_1 转换到 M_2: $\forall p \in P$

$$M_2(p) = \begin{cases} M_1(p) - \sum\limits_{(t,b)\in Y \wedge (p,t)\in A_1} E(p,t) < b > + \sum\limits_{(t,b)\in Y} E(t,p) < b >, & (p,t) \in A_1 \\ M_1(p), & (p,t) \in A_2 Y A_3 \end{cases}$$

上式中的第一个累加是移去的托肯，第二个累加则是新产生的托肯。若 M_2 是从 M_1 经过一个引发步 Y 的引发而得，称 M_2 从 M_1 直接可达 (directly reachable)，记作：$M_1[Y > M_2$．

一个引发步的引发是一个原子事件，且一个引发步的引发，不一定引发其中所有并发使能的绑定元素，但可以只引发其中几个。

下面举例说明 ECPN 中抑制弧和测试弧的图形表示和应用方法。

例 1　在图 1(a) 中，t 的输入库所为 p_1 和 p_2, $(p_1,t) \in A_3$, $(p_2,t) \in A_1$, $E(p_1,t) = 2 < b >$, $E(p_2,t) =< b >$, $E(t,p_3) =< b >$，且均为常量表达式，则 t 使能当且仅当 p_2 中至少含有一个托肯 b，且 p_1 中含有托肯 b 的个数小于 2．t 引发后，p_1 中的托肯保持不变，一个托肯 b 由 p_2 移到 p_3．类似地，在图 1(b) 中，$(p_1,t) \in A_2$, $(p_2,t) \in A_1$, $E(p_1,t) = 2 < a,b >$, $E(p_2,t) =< b >$, $E(t,p_3) =< b >$, 则 t 使能当且仅当 p_1 中至少含有两个托肯 $< a,b >$, p_2 中至少含有 1 个托肯 b．t 引发后，p_1 保持不变，p_2 中减少一个托肯 b，p_3 增加 1 个托肯 b．

定义 6　一个有限引发序列 (finite occurrence sequence) 是由标识和引发步组成的序列：

$$M_0[Y_1 > M_1[Y_2 > M_2 \cdots M_{n-1}[Y_n > M_n$$

其中 $n \in N$，且对于 $i \in \{1,\cdots,n\}$, $M_{i-1}[Y_i > M_i$．M_0 是引发序列的起始标识，M_n 是结束标识。非负整数 n 称为引发序列的长度。类似地，可以定义无限引发序列：$M_0[Y_1 > M_1[Y_2 > M_2 \cdots$, 仍把 M_0 称为引发的起始标识，但没有结束标识，也没有引发序列的长度。

定义 7　如果存在一有限引发序列，M' 是其起始标识，M'' 是其结束标识，且满足：$\exists n \in N$, $M'[Y_1 Y_2 \cdots Y_n > M''$，则称标识 M'' 是从 M' 可达的，其中 $Y_1 Y_2 \cdots Y_n$ 为 n 个引发步。若一个标识由初始标识 M_0 可达，简称其为可达的。所有从 M' 可达的标识集合记为 $R(M')$．

3 CORBA 服务规范的 ECPN 模型

我们用 ECPN 建模 CORBA 系统中的对象类的动态行为规范。一个 ECPN 可提供一个或

多个 IDL 接口中所定义服务的动态行为模型，提供相应接口的动态行为语义．每个在 CORBA-IDL 接口定义的服务，都映射为一个 ECPN 中相应的输入、输出库所和变迁．对于每个输入库所，它的托肯类型集是相应服务所有输入 (in) 、输入输出 (inout) 参数的类型集；对于输出库所，其托肯类型是所有输出 (out) 、输入输出 (inout) 参数的类型集及该服务可能的结果类型．如果服务出现异常，就要引入一个服务异常库所，它仅来自特定异常变迁的输入弧．异常库所的托肯类型为 <Exception> ， Exception 是所有 IDL 类型的超类，出现异常时，异常变迁可中断对服务的正常调用．客户向输入库所发送请求调用服务，如果调用正常结束，则从输出库所将调用结果返回给客户，否则从异常库所返回异常．

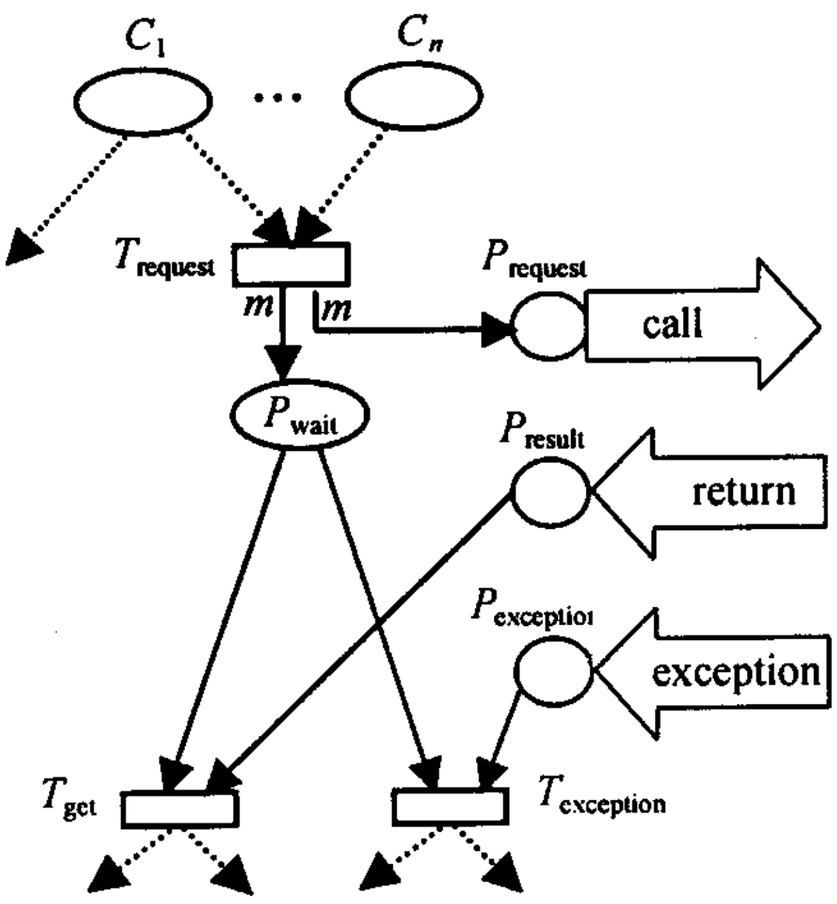

图 2 客户 / 服务器协议语义模型

CORBA 系统是由相互作用的对象构成，根据 IDL 定义的接口， CORBA 对象之间的通讯不需要了解对象的位置以及实现细节．客户对象向服务对象发出服务请求，服务对象提供远程接口 [3] ． CORBA 对象之间的调用遵循客户 / 服务器协议．根据 ECPN 定义，我们给出该协议的详细语义 (如图 2 所示) ．指定 3 种变迁，即

(1) 请求 (call) 变迁：调用对象提供的服务．

(2) 响应 (return) 变迁：返回对应的请求结果．

(3) 异常 (exception) 变迁：中断正常服务过程并处理异常．

图 2 中，变迁 T_{request} 是请求变迁， T_{get} 是对应的响应变迁， P_{wait} 是引入的一个等待库所． P_{request} 和 P_{result} 分别是服务请求库所和请求结果库所． $P_{\text{exception}}$ 是一个服务异常库所．变迁 $T_{\text{exception}}$ 获得服务异常信息． $E(T_{\text{request}}, P_{\text{wait}}) = E(T_{\text{request}}, P_{\text{request}}) = m$ 是两个弧表达式，这里 m 是变量．但是，一旦 T_{request} 引发， m 有确定的值，且 m 的取值等于客户 $C_1, \cdots, C_n$ 中请求服务的个数．对于发送请求的对象， P_{wait} 可实现同步调用，即只有等待获取 T_{get} 结果，才能进行下一步；也可实现异步调用，无需考虑请求结果，可执行其它的动作． CORBA 对象的内部调用相对简单，无需引入请求和响应变迁，只要执行服务即可．在多个用户同时请求服务时， T_{get} 和 $T_{\text{exception}}$ 可并发．

下面给出一个简单的包含两个接口 Department 和 User 的 CORBA 系统，如图 3 所示．

```
interface Department {
exception UserExisted;
exception noSuchUser
Department newUser(inout string name, out ineger ID);
raises(UserExisted);
User findUser(out string name, inout integer ID);
raises(noSuchUser);
};
interface User{
void operate(in string subject, in float account);
}
```

图 3 Department 接口和 User 接口

假设一个单位是一个 IDL 对象接口，其中包含一组用户，且每个用户的编号、名称是唯一的．当输入一个编号，在单位里查找这个用户时，若该用户存在，则输出他的相关信息，否则引发异常，显示没有此用户．当插入一个用户时，如果该用户已存在，则引发异常，否则增加该用户进行初始化操作，输出结果．

对于 Department 接口，其行为规范如图 4 ．图 4 中，变迁 T_1 有两条输入弧，从库所 Department 到 T_1 的弧是抑制弧，如果库所 I_InsertUser 中的托肯和 Department 中托肯相同，那么 T_1 不能引发．库所 Department 到 T_3 是测试弧，意味着当库所 Department 和库所 I_FindUser 有同种托肯时，变迁 T_3 才可使能．

容易看出，图 4 能够清楚地模拟 Department 的动态行为．事实上，我们可以应用 Petri 网分析技术，基于可达图算法，从形式上严格证明图 4 中 Department 的动态行为和相应的 CORBA 系统规范是一致的．

假定在库所 Department 中有一个单位的用户，如果同时有一个插入用户请求和查找用户请求，那么 T_1 和 T_3 可并发．本例中，对于任一请求插入服务，要么存在发生序列 $T_1 \to T_2$ ，使得服务正常结束，从输出库所 O_InsertUser 返回调用结果，要么引发变迁 T_{e1} ，将异常信息从异常库所 E_userExisted 返回给客户．若有来自多个客户的请求同一种服务，例如有多个查找用户请求，则也能应用这个模型模拟，此时需要在请求调用中加入不同客户的识别标识作为参数，送给相应输入库所．

```
Specifies Department
Place User<string, User>;
Place I_InsertUser<string>;
Place E_UserExisted<Exception>;
Place O_InserUser<string, integer>;
Place I_FindUser<integer>;
Place O_FindUser<string, integer>;
Place E_noSuchUser;
```

```
transition T1{
   Precondition {true}
   action {a=newuser()}
   }
transition T2{
   action {a.operate();}
   }
transition Te1{
   Precondition {ture}
   action {nameAlreadyExists();}
}// 省略了部分变迁定义
```

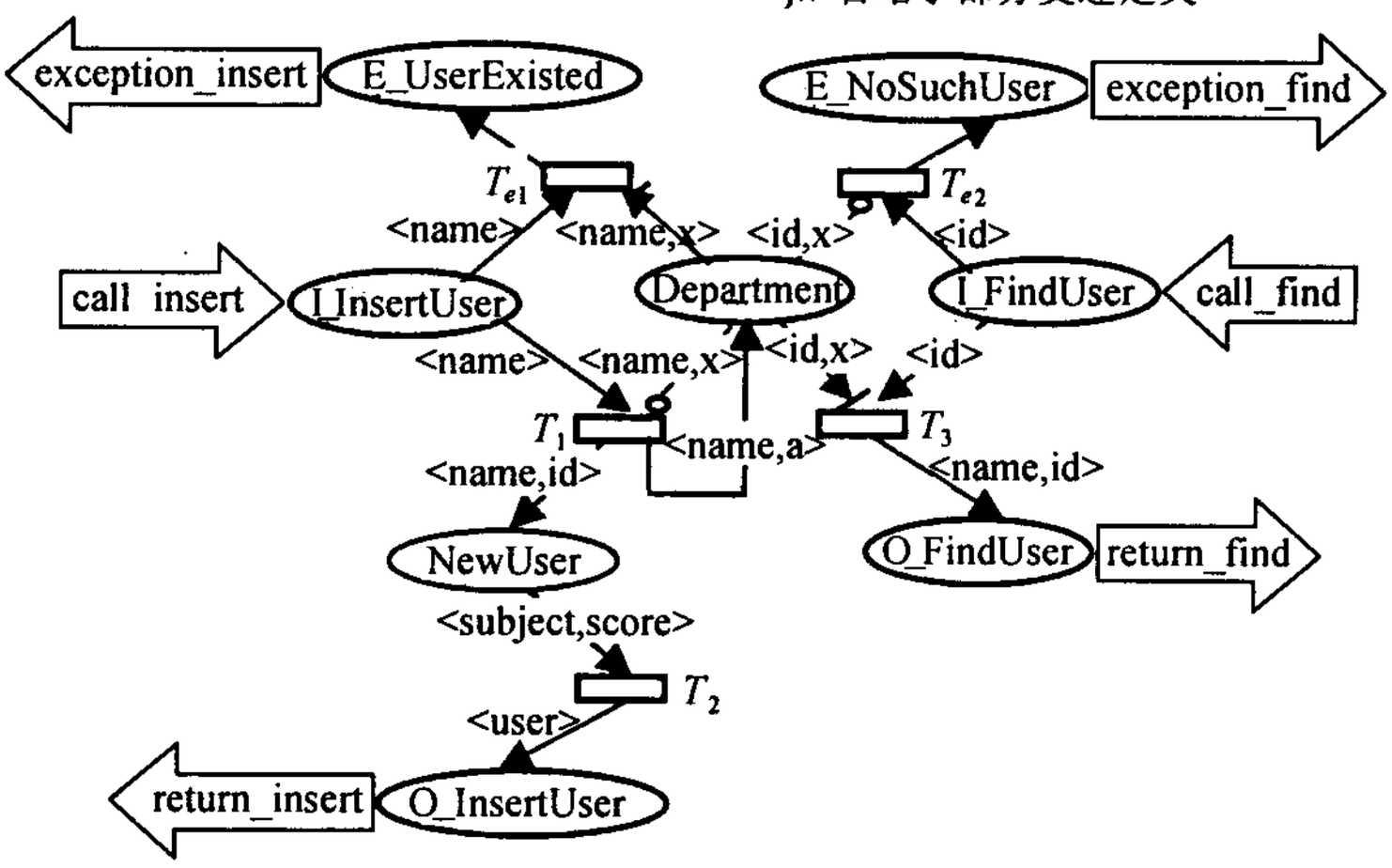

图 4　Department 的行为模型

下面分析 Department 接口的几个主要性质．假设 id 表示编号， name 表示用户名．

性质 1　对任一用户查找请求， Department 都能够响应并返回一个结果或异常．

证明：若有客户请求查找编号为 30 的用户，则请求变迁 call_find 引发，且库所 I_FindUser 有一个编号 30 的托肯．如果库所 Department 中有编号 30 ，则 T_3 引发．而 T_3 引发使得库所

O_FindUser 中含有托肯 <ComLab1, 30>(假设编号 30 对应的用户名为 " ComLab1 ") . 最后响应变迁 return_find 返回查找结果, 即存在引发序列: (id=30)[call_find>(I(I_FindUser)=30, I(Department)=< 30, ComLab1>)[T_3 >(I(O_FindUser)=< 30, ComLab1>)[return_find> .

如果库所 Department 中没有编号 30 , 类似可得到引发序列:

(id=30)[call_find>(I(I_FindUser) =30, I(Department)=< 30, ComLab1>)

[T_{e2} >(I(E_NoSuchUser)=30, NoSuchUser)[exception_find> . 证毕

性质 2 对任一插入用户请求, Department 都能够响应并返回成功或异常信息.

证明 : 其证明过程与性质 1 类似.

性质 3 Department 系统对任一查找或插入请求, 都能及时响应并返回结果或异常信息.

证明 : 由性质 1 和性质 2 直接得证.

4 结论

本文提出基于扩展的有色 Petri 网 (ECPN) 形式化描述方法, 并应用其给出了一个 CORBA 对象服务接口规范的 ECPN 模型. 尽管是一个简单例子, 但它具有一般性, 基本上反应了 CORBA 服务的动态、并发等分布式特性, 符合 CORBA 规范原则. 在今后的研究中, 将进一步完善 ECPN , 应用 Petri 网的分析方法, 描述和分析 CORBA 公共服务规范, 这些工作对 CORBA 系统的理解和分析及实现基于 CORBA 的应用具有重要的作用和实际意义.

参 考 文 献

[1] The Object Management Group. The Common Object Request Broker: Architecture and Specification,Version 2.4.2, 2000

[2] The object Management Group, CORBA Specification, 2000

[3] Slama D, Garbis J, Russell P. Enterprise CORBA, Prentice Hall PTR, 1999: 67–120.

[4] Walker D. Objects in the π-calculus. *Information and Computation*, 1995, 116(2): 253–271.

[5] Swatman P A, Swatman P M C, Duke R. Electronic data interchange: A high-level formal specification in object-Z, Proceedings of 6th Australian Software Engineering Conference, Sydney, Australia, Springer-Verlag, July 1991: 341–354.

[6] Gaspari M, Zavattaro G. A process algebraic specification of the new asynchronous CORBA messaging service. *ECOOP'99*, LNCS 1628, Lisbon, Portugal, Springer-Verlag, June 1999: 495–518.

[7] Hong Zheng, Shi-xian Li. The description of CORBA objects based on Petri nets. Lecture Notes in Computer Science LNCS 2495, Springer-Verlag, Shanghai China, Oct. 2002, 48–57.

[8] Jang-Eui Hong, Doo-Hwan Bae. Software modeling and analysis using a hierachical object-oriented Petri net, *Journal of Information Science*, 2000, 130(1–4): 133–164.

[9] Bastide R. Approaches in unifying Petri nets and the object-oriented approach, Proceeding of the International Workshop on Objected-Oriented Programming and Models of Concurrency, 1st Workshop on Object-Oriented Programming and Models of Concurrency, OO-MC'95, 16th International Conference on Applications and Theory of Petri Nets, ICATPN'95, Torino, Italy, June 1995: 238–257.

[10] Kilov H, Ross J. Information Modeling—An Objected Oriented Approach, Prentice-Hall, 1994, Chapter 7. Standards.

[11] 袁崇义. Petri 网原理. 北京: 电子工业出版社, 1998: 88–95.

郑 红: 女, 1973 年生, 博士, 研究方向: 形式化方法、 CORBA 与软件体系结构.

李师贤: 男, 1944 年生, 教授, 博士生导师, 研究方向: 软件工程、分布式计算、面向对象技术等.