一个面向对象的二级并发模型*

王 戟 陈火旺

(长沙工学院计算机系,长沙 410073)

摘要 本文提出了一个面向对象的二级并发模型 FORCE－Model. 它从需求规范的角度充分开发对象间和对象内两个层次上的并行性,从而将面向对象与并发性有机地结合起来,提供了有效的实时系统建模框架. 基于该模型,我们开发了一个多视点可视规范语言族 FORCE－Language.

关键词 面向对象,需求规范,并发,实时系统.

面向对象方法学为软件系统的规范、设计和程序设计提供了一种新的方法学. 它影响着整个软件开发周期. 这也带来了一些需要仔细研究的问题,尤其是如何以一种更合理的方式规范并发性. 本文的研究目的即是将面向对象与并发性有机地结合起来,形成有效的软件系统建模框架.

面向对象通过其灵活的模块化能力提供了一种自然的建模框架;另一方面,并发不仅是现实世界中系统的固有属性,而且也是减少处理时间,提高处理能力的关键点. 因此,面向对象并发模型可望得益于以上两点. 但从目前来看存在如下一些问题:

第一,理想的面向对象并发模型还未形成. 面向对象为并发提供了灵活的途径和机制. 异步消息、远程过程调用和同步汇集均与其消息传递模式匹配. 已有将面向对象和并发性结合起来的程序设计语言[3]. 但是高层次的需求规范模型和语言尚未见到.

第二,面向对象对软件开发有两个相互关联方面的影响:方法和语言机制. 对象作为其核心,成为面向对象建模的基本概念. 我们认为,不仅对象与对象之间存在并发,对象内部亦存在并发. 现实世界确实如此. 然而,现有模型仅规范对象间的并发.

本文针对实时系统开发提出了一种新的面向对象二级并发模型 FORCE－Model. 它把对象作为并发 Agent,以刻划现实世界中的实体. 系统是由一组有内部并发的对象组成. 对象间的并发消息传递刻划系统元素间的交互作用. 这样,FORCE－Model 具有二级并发特性:对象间和对象内并发. 对象间并发的基本方式是一种异步消息传递,而对象内并行也是一种异步. FORCE－Model 不拘泥于"一切皆对象". 它的二级并发机制不仅充分开发了并发性,而且加大了对象粒度,提高了模型的抽象层次. 基于 FORCE－Model,我们开发了基

* 本文 1992－03－02 收到,1992－06－23 定稿

本文得到国家自然科学基金的资助.作者**王戟**,25岁,博士生,主要研究领域为需求工程,面向对象方法学,自动推理. **陈火旺**,58岁,教授,主要研究领域为软件工程,人工智能.

本文通讯联系人:王戟,长沙 410073,长沙工学院计算机系

于图形的多视点需求规范语言族 FORCE－Language .

FORCE－Model & Language 是实时系统开发的形式化方法和工具研究 FORCE(Formal methOds and tools for the development of Real－time reaCtive systEms)的一部分.

1　FORCE－Model

1.1　对　象

FORCE－Model 中,计算或信息处理由一组抽象实体对象和对象间的消息传递完成.每个对象都有其内部状态和状态上的处理操作.对象的内部状态由该对象的可观察属性的值的序偶确定.在接收到外界的消息后,它将发生状态转换,并会进行相应基本动作:状态改变、消息传递及其它处理操作.故定义一个对象,要规范它的可观察属性,可接收的消息模式和状态转换序列.我们设定一个对象有一个无穷长的队列按到达序存放待处理消息.对于一个对象,我们规定了三种模式:休眠、等待和活动.对象的初始模式是休眠,在接收一个消息后,它转为活动模式.当一个活动的对象在其当前状态停下以等待某一个特定转换条件(如某一特定的消息传递)满足时,它处于等待模式.当条件满足后,它又进入活动模式.当对象处理完所有消息后,它又处于休眠模式.

以 FORCE－Model 的对象概念为基础,我们对待开发系统可用层次分解的方法获得其组成对象集[7].一个对象可看成为子问题域的抽象,可以分解为一组子对象和子对象间的消息传递.

FORCE－Model 中,以 Statecharts[1,2]为基础来描述对象.对象的状态可以进行层次分解:AND 分解和 OR 分解.显然,这就形成了对象内部的并发性.对象在同一时刻可以有多个不同的活动状态,也可以有多个同时发生的转换.

图 1 说明了对象内部的并发机制.Q 的状态 AND 分解为 S 和 S′.S 状态 OR 分解为 A、B、C、D.S′状态 OR 分解为 E、F、G、H.转换 α2:B→C,β2:F→G 可并发执行.但 α1:A→B 必须在 β1:E→F 完成后才可执行,β3:G→H 必须在 α2:B→C 完成后才可执行.这表明了两个并发又相互影响的处理进程.

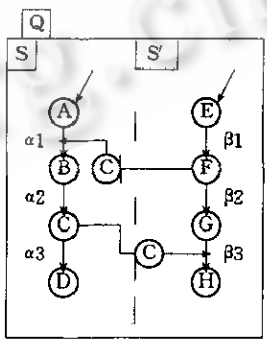

图1

FORCE－Model 的对象内并发与 D. Harel 的 Statecharts 的并发有相近之处,都用 AND 分解来表示并发.但有着不同:第一,Statecharts 的并发机制主要是依靠其传播机制,它将转换后的动作传播至各状态转换的激励中.而在 FORCE－Model 中,我们把动作作为对象间并发的接口预留了下来.用状态历史对对象内的并发进行控制.其次,我们将时间域引入控制阀中,使得状态活跃的历史情况可用来控制并发,以适应实时系统规范所需的严格复杂的时限表述.此外,转换可引入前后置条件表示,使模型的抽象层次提高.故 FORCE－Model 克服了 Statecharts 存在的一些弱点:有较好的结构化机制和较强的时序表示.Statecharts 本身的结构化机制较弱,在 STATEMATE 中,对系统的建模全部使用一张 Statecharts 图表示系统的动态性质.虽然有 ZoomIn 和 ZoomOut,但仍是繁琐的,而且其 Activitycharts 和 Statecharts

的对应关系很微妙，不直观. FORCE－Model 基于面向对象严格区分了二个不同层次的并发，在对象内，我们认为各个状态之间是“可见的”. 用状态(历史)刻划对象内的并发是合理的. 对象内的并发被严格地封装在对象内部. 这符合信息隐蔽原则.

1.2 消息传递

FORCE－Model 把消息传递作为对象间的并发机制. 对象之间可同时发生多个消息传递. 它的基本方式是异步通讯. FORCE－Model 中，对象间的相互作用表现为产生和接收服务请求. 在一次交互作用中，一个对象(请求者)产生一个服务请求给另一个对象(服务者). 一个请求者可同时给多个服务者发请求；一个服务者也可接收多个请求者的请求. 请求者在向服务者发出请求后，可以按自身的状态继续执行或等待回复，它无须考虑服务者的当前状态和模式. 这都反映在 FORCE－Model 的异步通讯模式中. 同步通讯是异步通讯的一种特例. 故 FORCE－Model 的并发有很强的能力. 我们没有假定一个整个系统的全局时钟. 一般而言，任何没有制约关系的事件被认为是可同时发生的. 对象是最大限度的并行单元. 这是与分布式实时系统相适应的.

FORCE－Model 中，对象间的消息传递称为 Activity. 当服务者接收一个 Activity 后，它将产生一系列状态变换. 故一个 Activity 的处理对应着一系列状态变换. 消息传递不具有广播性质，即请求者必须知道服务者名字才可以发送请求. 消息传递总在有限时间内到达服务者并且保持到达序. 注意，对象在被定义时是以类的概念定义的，服务者的名是形式上的虚名.

1.3 类与继承

类与继承是面向对象的基本特点. 用类和继承可以有效地组织分析问题域. FORCE－Model 以语义网的方式规范了类与对象之间的实例化关系，超类与子类之间的继承关系，并扩充了对象之间的聚集关系. 但是目前仍不清楚如何在一个并发模型中有效地发挥作用，有一点可以肯定：类可以有效地组织问题域. 继承作为一种增量式规范手段是可取的. 聚集作为将来通讯的受限传播机制的基础正在考虑中.

2 FORCE－Language

作为 FORCE－Model 的描述语言，我们开发了可视需求规范语言族 FORCE－Language. 它从组成、活动和状态等三个视点刻划系统的需求规范. 对应的，F－Entitycharts(实体图)、F－Activitycharts(活动图)、F－Statecharts(状态图)三者组成了 FORCE 需求规范.

F－实体图标识问题域中的对象、类以及它们之间固有相互关系，即表征问题域中所有对象和类的组织. 它是一个有向图，其结点为对象和类. 对象是问题域中有意义实体的抽象；类是一组可用统一属性和操作刻划的对象的描述. 有三种边刻划结点之间的固有关系. (1)类之间的继承关系. (2)对象之间的聚集关系，它实际上是问题域中实体间的一些常识的描述(例如，微机系统由主机、显示器、键盘和打印机组成). 图 2 给出了一个带窗口的闹钟系统的 F－实体图.

所示的闹钟系统由 Alarm、Bell 和 Clock 三对象组成，分别属于 Alarm_Clock、Window 和 System_Clock 类. 当 Alarm 闹时，它打开一个窗口. 当闹钟不闹时，窗口是关闭的. 闹钟

显示当前时间，并可显示闹时. System_Clock 提供标准时，Alarm_Clock 的可观察属性是 current_time 和 alarm_set(闹时集合). Window 的可观察属性是 display_area.

F－活动图用来为系统进行概念建模，刻划系统中各对象之间的相互通讯，标识对象所展示的服务行为(即可接收的消息模式). 它描述的是问题本身. F－活动图的结点是对象，边即为对象间的交互通讯，基本模式为：〈消息名〉〈消息数据〉. 一次典型的通讯是接收某请求后经过处理再向其它对象发消息. F－活动图是 ERD(实体/关系图)和 C/DFD(控制/数据流图)的一种结合；对象则作为控制和数据的枢纽.

图 3 给出了图 2 所示闹钟系统的 F－活动图. Alarm 从 Clock 取标准时，并可向 Bell 发出(打开/关闭)窗口的消息. 一组系统按扭可以控制闹钟的状态. 注意，此间出现了两种通讯：一种是等一答通讯 time(st?)表示对象要等待回复数据；另一种是 openwindow 等其他消息传递，无须等待回复.

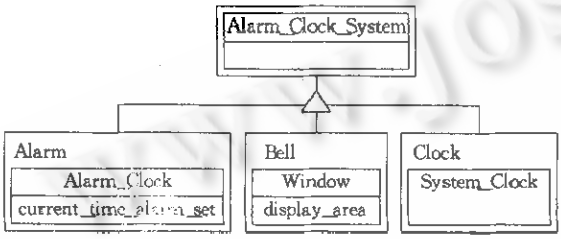

图2 F－实体图例子

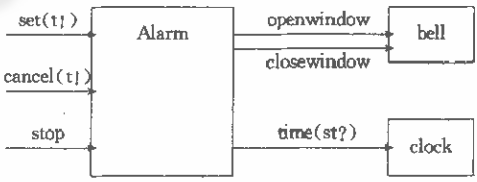

图3 F－活动图例子

F－实体图和 F－活动图从组成和活动两方面构画了系统需求. F－状态图则从状态即行为的角度刻划系统规范的动态行为——控制时序，尤其是时间限制关系. F－状态图对 Statecharts 的转换进行了扩充，不仅可以刻划实时系统中严格的时序限制，而且可有效地控制对象内并发. 它在同一时刻可以有多个活动状态，多个状态转换也允许同时进行，状态可以层次地分解为低层的 F－状态图，从而可以充分地表达状态和转换间的逻辑和时序关系.

F－状态图对 Statecharts 的扩充从语言上体现在对状态转换标号的扩充. State－charts 中状态转换标号为 α[C]/β，其中，α 为激励事件，C 为转换条件，β 为动作. F－状态图中转换标记则形如：

$$\langle\alpha\rangle\langle C_1\rangle\langle C\rangle/\langle\beta\rangle\langle C_2\rangle$$

其中，α 为激励事件，即为对象外部发来的各种消息传递；C_1 为对象当前状态发生该状态转换的前置条件；C 为控制表达式，即对象的其他状态的(历史)情况的控制阀；β 为动作，例如对象对外部发出的各种消息传递；C_2 为对象完成该状态转换后应满足的后置条件. 这里，前后置条件用来规范转换的功能行为. 控制表达式形如：(s，て)或(¬s，て)其中，s 为状态集中某状态，て为时间域. C_1 和 C_2 为控制表达式，则 $C_1 \wedge C_2$ 亦为控制表达式. 时间域て形如以下三种形式之一：

$$\langle t_1,t_2\rangle,[t_1,t_2],\langle t_1,t_2\rangle!,\ \text{其中}\ t_1\leqslant t_2\leqslant 0.$$

控制表达式的含义是：

(s，$[t_1,t_2]$)被满足 iff 距当前时刻前$|t_1|$到$|t_2|$时间域中每一时刻 s 均活动；

(s,⟨t_1,t_2⟩)被满足 iff 距当前时刻前|t_1|到|t_2|时间域中有一时刻 s 为活动;

(s,⟨t_1,t_2⟩!)被满足 iff 距当前时刻前|t_1|到|t_2|时间域中有一时刻 s 为活动,且距当前时刻前|t_1|+1 时,s 不活动;

(¬s,[t_1,t_2])被满足 iff 距当前时刻前|t_1|到|t_2|时间域中每一时刻 s 为不活动;

(¬s,⟨t_1,t_2⟩)被满足 iff 距当前时刻前|t_1|到|t_2|时间域中有一时刻 s 为不活动;

(¬s,⟨t_1,t_2⟩!)被满足 iff 距当前时刻前|t_1|到|t_2|时间域中有一时刻 s 为不活动,且距当前时刻前|t_1|+1 时,s 活动.

控制表达式的思想出自[4]. 这里我们引入用来扩充 Statecharts 以控制对象内的实时并发. 状态活动的历史情况可以清晰地被表达以控制当前的行为.

图 4 所示描述了 Alarm_Clock 的 F—状态图. 可以看到,有三个并发的行为:其一是不断取标准时;其二是接收消息 set(t!)和 cancel(t!);其三是控制闹时. 当前时间属于闹时集时,发送消息打开窗口,若有 stop,则关闭窗口,否则 60 时间单元(秒)后关闭窗口.

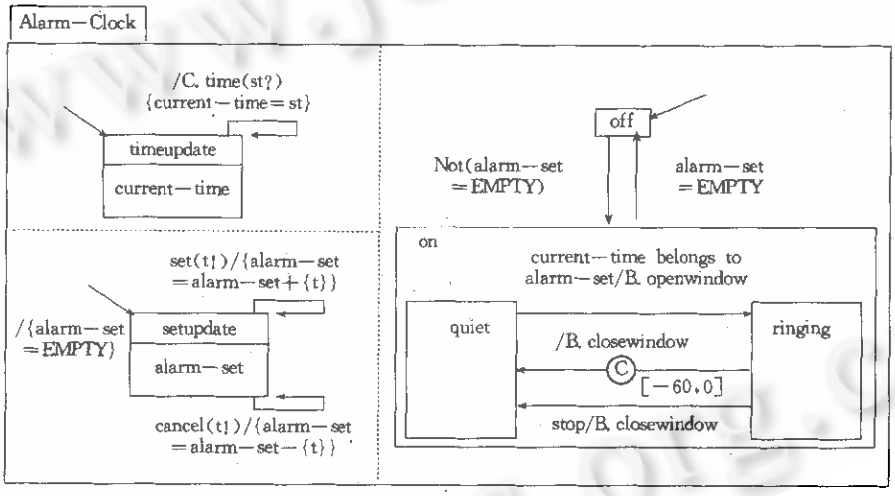

图4 F—状态图例子

3 FORCE—Model 的语义

本节,我们非形式地讨论 FORCE—Model 的二级并发的语义.

首先,我们讨论对象间并发. 三元偶⟨a,M,b⟩用来描述对象间的请求/服务,其中,a 为请求的对象名,b 为服务的对象名,M 为通讯的消息. 我们用请求/服务序列来描述系统的行为. 注意,存在多个同时活跃的请求/服务. 为了便于分析,我们使用一个简化了的模型,即用考虑各种可能性的交互来模拟并行性,用交织序列来表征并发.

F—状态图中,出现在状态转换标号中动作部分的消息传递形式是:服务者. 消息名⟨参数⟩. 用请求/服务序偶则可表示成:

⟨Self,消息名⟨参数⟩,服务者虚名⟩

同样,在状态转换标号中激励部分的消息传递形式用请求/服务序偶则可表示成:

⟨ * ,消息名⟨参数⟩,Self⟩

那么,一个F－状态图定义的类X的外部行为是它所有可能的请求/服务序列的集合.我们把它定义为类的迹,记为Tclass(X).

我们把类的迹与F－活动图相结合,可将原迹中的服务者虚名替换成实名,即成为对象的迹Tobject.

系统行为则由交互对象的各种可能的迹组成的集合给定,记为Trace.定义Trace|object为迹投影,它仅留下与对象object有关的请求/服务.把它进行局部化,Local(Trace|object),即将请求/服务序偶中对象object名换为Self,其它请求者的对象名则换为*.

由对象$O_1,\cdots,O_n$组成的系统S的行为是如下集合:

$$Tsystem(S)=\{Trace\mid \forall\ o\in\{O_1,\cdots,O_n\},\ Local\ (Trace|o)\ \in Tobject(o)\}$$

这表明系统的迹是所有对象所可能发生的合理交互,它刻划了问题和问题域所会发生的合法行为的组合.可以看出,对于一个有穷对象组成的系统的有穷请求/服务序列,可以判定它是不是一个可能发生的合法序列.

对于表现为同时有多个活动状态和转换发生的对象内并发,我们用标记映射Marking来定义.这里我们不考虑激励事件,动作和条件,只考虑控制表达式.Marking M是映射:

$$S\times\{\cdots\cdots-2,-1,0\}\rightarrow\{T,F\}$$

其中,S为状态集合.控制表达式的满足关系可如下形式定义:

M Sat $(s,\langle t_1,t_2\rangle)$ iff $\exists\ t_3(t_1\leqslant t_3\leqslant t_2)$ M $(s,t_3)=T$

M Sat $(s,[t_1,t_2])$ iff $\forall\ \ t_3(t_1\leqslant t_3\leqslant t_2)$ M $(s,t_3)=T$

M Sat $(s,\langle t_1,t_2\rangle!)$ iff $\exists\ t_3(t_1\leqslant t_3\leqslant t_2)$ M $(s,t_3)=T\wedge M(s,t_1-1)=F$

M Sat $(\neg s,\langle t_1,t_2\rangle)$ iff $\exists\ t_3(t_1\leqslant t_3\leqslant t_2)$ M $(s,t_3)=F$

M Sat $(\neg s,[t_1,t_2])$ iff $\forall\ \ t_3(t_1\leqslant t_3\leqslant t_2)$ M $(s,t_3)=F$

M Sat $(\neg s,\langle t_1,t_2\rangle!)$ iff $\exists\ t_3(t_1\leqslant t_3\leqslant t_2)$ M $(s,t_3)=F\wedge M(s,t_{1-1})=T$

状态s为活动即M(s,O)=T.一个状态转换能行当且仅当它的源状态活动,有激励事件,前置条件为真且控制表达式被满足.标记M经状态转换P后的后继标记M′即M[P]如下:

$M'(s,O)=T$　　$s\in$转换目标状态集

$M'(s,O)=F$　　$s\in$转换源状态集－转换目标状态集

$M'(s,O)=M(s,O)$　　$s\in\complement$转换源状态集$\cup$转换目标状态集

$M'(s,t)=M(s,t+1)$　　$t<O,s\in S$

那么,对于一个F－状态集,给定其初始标记M0,假设各前置条件自然满足,伴以对象间相互通讯的迹(从而获得激励事件),可形成标记序列$M_0,M_1,\cdots$.这就形成了F－状态图的执行.

4　结束语

本文给出了一个具有二级并发特性的需求分析模型FORCE－Model.它将面向对象与开发并发性有机地结合起来.对象间的并发和对象内的并发可使问题中固有的并发性充分而自然地开发出来.文献[6]给出了一种面向对象分析方法.它具有全面的面向对象特征.但

它更侧重于指南上的而非技术上的考虑.相对于一般的面向对象方法学,

面向对象=对象 & 类+继承+消息传递通讯

FORCE－Model 可以由下述等式刻划:

FORCE－Model=对象网络+对象内并发+对象间并发

对象网络=对象 & 类+继承+聚集

对象内并发=对象状态+转换+深度+控制表达式

对象间并发=对象+消息传递通讯

基于FORCE－Model,我们开发了需求规范语言族FORCE－Language.它将自动机和前后置断言两种方式结合起来以适应实时软件建模.

进一步的研究包括:开发完整的 FORCE－Language 的形式语义;与时态逻辑结合形成实时系统规范和验证的统一框架.

参考文献

1 Harel D *et al*. Stalemate: a working environment for development of complex reactive systems. IEEE TSE, 1990, **16**(4).

2 Harel D. Statecharts: a visual formalism for complex systems. Sci. of Comp. Prog., 1987, **8**.

3 Yonezawa A. ABCL: an object oriented concurrent system. MIT Press, 1990.

4 Gabrielian A, Franklin M K. Multi－level specification of real time systems. CACM, 1991, **34**(5).

5 Bear S *et al*. Graphical specification of object oriented systems. In: ECOOP/OOPSLA '90 Proceedings, 1990.

6 Coad P, Yourdon E. Object oriented analysis. Second Edition, Yourdon Press, 1991.

7 王戟,陈火旺.面向对象需求分析方法 OORAM.中国青年计算机研究进展,国防科技大学出版社,1991.

8 Wang Ji, Chen Huowang. A framework for object oriented modelling. ICCIM'91, Singapore, 1991.

A TWO－LEVEL OBJECT－ORIENTED CONCURRENT MODEL

Wang Ji and Chen Huowang

(*Department of Computer Science, Changsha Institute of Technology, Changsha* 410073)

Abstract This paper presents a two－level object－oriented concurrent model, called FORCE－Model, which combines object orientation and concurrence from requirements specification point of view and provides an effective framework for modeling real－time reactive systems. Intra－object concurrence and inter－object parallelism can be exploited effectively. Based on the model, a family of multi－view visual specification languages are developed.

Key words Object orientation, requirements specification, concurrence, real－time reactive systems.