UML 顺序图的一种形式化描述方法*

张 姝,张广泉
(重庆师范大学 数学与计算机科学学院,重庆 400047)

摘 要 统一建模语言 UML 是一种通用的图形化建模语言,在面向对象系统的分析和设计中,它已成为了事实上的工业标准。但 UML 不是形式化的建模语言,缺乏精确的、形式化的语义,因此阻碍了它的进一步发展。线性时序逻辑是并发或反应式程序动态语义的一种形式化描述语言,它适合用来精确地表示模型的动态语义。本文定义了顺序图的形式化语法,采用线性时序逻辑给出了顺序图的语义描述,并通过实例分析,对模型的某条性质进行了证明,为模型做进一步分析和验证提供了基础。

关键词:UML 顺序图;形式化语法;形式化语义;线性时序逻辑
中图分类号:TP311 **文献标识码**:A **文章编号**:1672-6693(2007)03-0042-04

A Formal Description of UML Sequence Diagram

ZHANG Shu, *ZHANG Guang-quan*
(College of Mathematics and Computer Science, Chongqing Normal University, Chongqing 400047, China)

Abstract: The Unified Modeling Language (UML) is a common graphical modeling language. It has become a de facto standard in the analyzing and designing the object oriented system. However, the UML is not a formal language, its lack of rigor and formal semantics prevents it from further developing. Linear temporal logic is a formal description language of the dynamic semantics by of the concurrent or the reactive program. It suits for precisely representing the dynamic semantics of a model. This paper defines formal syntax of the sequence diagram and gives a description of its semantics by using the linear temporal logic. Moreover, a property of the model is proved. It provides the foundation for the analysis and validation of the model.

Key words: UML sequence diagram; formal syntax; formal semantics; linear temporal logic

统一建模语言(Unified Modeling Language, UML)是一种通用的图形化建模语言,它通过定义多种图和模型元素来描述系统分析和设计的结果,尤其适用于大型、复杂系统的建模。然而,UML 却是半形式化的建模语言,其模型的语法是通过元模型以 UML 的类图的方式定义的,静态语义采用对象约束语言(Object Constraint Language,OCL)进行描述,模型的动态语义则是直接由自然语言来表达的[1]。采用这种方法描述的语义存在着许多不足:表达的不严格和不精确,对模型难以进行一致性检查和正确性分析,不利于系统的求精和验证[2]等。

形式化方法是指建立在严格数学基础上的软件开发方法,采用形式化方法对系统进行形式化分析和验证,是构造可靠安全软件的一个重要途径[3]。现有 UML 模型的形式化工作主要集中于活动图和状态图,针对顺序图进行的形式化描述却很少[4]。因此,本文以 UML 规范为基础,定义了顺序图的形式化语法,采用线性时序逻辑给出了顺序图的语义描述。这不仅有利于准确而无二义性地理解其对象的行为,而且更有助于对模型的形式化分析和验证。

1 UML 顺序图简介

UML 顺序图展示了一种交互,由一组对象和它们之间的关系组成,着重体现对象间消息传递的时

* 收稿日期:2006-09-30 修回日期:2007-04-18
资助项目:重庆市自然科学基金(No. CSTC2006BB2259);重庆市教委科学技术研究项目(No. 040803);中国科学院计算机科学国家重点实验室开放课题(No. SYSKF0303)
作者简介:张姝(1983-),女,湖北荆门人,硕士研究生,研究方向为软件工程与形式化方法。

间顺序。顺序图采用两个轴:水平轴表示参与交互的各个对象,垂直轴表示时间。顺序图中的对象用带有垂直虚线的矩形框表示,矩形框内标有类名、对象名或角色名。垂直虚线称为对象的生命线,代表在对象之间的交互作用中该对象的生命周期。对象间的通信消息通过对象生命线之间的箭头表示,箭头的形状表明了消息的类型是发送还是返回。消息按发生的时间顺序从上到下排列,每个消息旁边标有消息名,也可以加上参数[5]。图1给出了一个用户成功登录系统的顺序图。

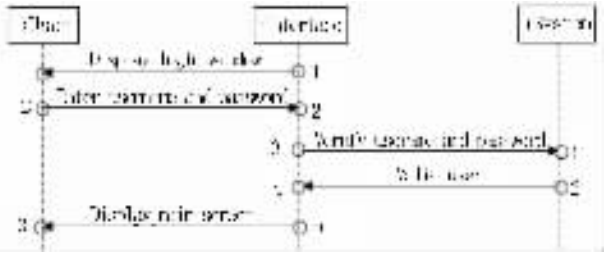

图1 用户成功登录系统的顺序图

2 UML顺序图的形式化描述

2.1 形式化语法

定义1 UML顺序图可以表示为一个四元组 $SD = <Obj, Msg, Loc, F>$,其中:

· Obj 是顺序图中对象的集合。

· Msg 是顺序图中消息的集合。

· Loc 是顺序图中位点的集合。位点是对象的生命线上发送或接收消息的点。每个位点都形如:$<O_i, 1>$,其中 $O_i \in Obj$,1 为对象 O_i 的位点。

· $F: Msg \times \{s, r\} \rightarrow Loc$ 是顺序图中从消息到位点的一个函数关系。s 表示消息的发送,r 表示消息的接收。

2.2 形式化语义

线性时序逻辑最早是由Pnueli引入计算机科学中的,是并发或反应式程序动态语义的一种形式化描述语言,它主要包括以下时序算子[6]:

□ 任一时刻算子,$\Box p$ 表示 p 总是为真或者 p 永远为真;

◇ 某一时刻算子,$\Diamond p$ 表示 p 最终为真或者 p 有时为真;

○ 下一时刻,$\bigcirc p$ 表示 p 在下一时刻为真;

μ 直到算子,$p \mu q$ 表示 p 一直为真直到 q 为真;

ω 除非(等待)算子,$p \omega q$ 表示 p 一直为真直到 q 为真,或 p 永远为真;

⊟ 过去任一时刻算子,$\boxminus p$ 表示 p 在过去总是为真;

⟐ 过去某一时刻算子,$\diamondminus p$ 表示 p 在过去有时为真;

⊖ 前一时刻算子,$\ominus p$ 表示 p 在上一时刻为真;

ζ 自从算子,$p \zeta q$ 表示在过去自从 q 为真后,p 一直为真;

β 退回算子,$p \beta q$ 表示在过去 p 一直为真,或自从 q 为真后,p 一直为真。

线性时序逻辑可以很准确地表示出对象间消息传递的时间顺序,而且有助于对模型的性质进行描述和验证。基于此,下面采用线性时序逻辑定义了UML顺序图的5种基本交互事件的语义,该顺序图的其他语义均可由这5条基本表示。

定义2 $\forall O_i \in Obj, X_i$ 为 O_i 所有状态的集合,$x_j \in X_i$,称 $G(O_i, x_j)$ 为对象 O_i 的一个格局。令 GJ 为对象的格局集合,则转换关系为 $T \subseteq GJ \times Msg \times GJ$。

① 当 O_i 不执行任何消息的收发事件时

$$G(O_i, x_j) \wedge \mathrm{NULL} \rightarrow \bigcirc G(O_i, x_j)$$

对象 O_i 不执行任何消息的收发事件时,其状态不会发生任何改变。

② 当 O_i 执行消息 $m1$ 的发送事件时

$$G(O_i, x_j) \wedge m1 \rightarrow \bigcirc G(O_i, x_j)$$

对象 O_i 执行消息的发送事件后,其状态不会发生改变。

③ 当 O_i 执行消息 $m1$ 的接收事件时

$$G(O_i, x_j) \wedge m1 \rightarrow \bigcirc G(O_i, x_k)$$

对象 O_i 执行消息的接收事件后,其状态可能会发生改变。

④ 当 O_i 发送消息 $m1$ 到 O_n 时

$$G(O_i, x_j) \wedge m1 \wedge G(O_n, x_p) \rightarrow \bigcirc G(O_n, x_q)$$

对象 O_i 发送消息到 O_n 后,O_i 的状态不会发生改变,O_n 的状态可能发生改变。

⑤ 当 O_i 从 O_n 接收消息 $m1$ 时

$$G(O_n, x_p) \wedge m1 \wedge G(O_i, x_j) \rightarrow \bigcirc G(O_i, x_k)$$

对象 O_i 从 O_n 接收消息后,O_i 的状态可能发生改变,O_n 的状态不会发生改变。

3 实例分析

下面采用本文的方法来对UML顺序图表达的一个系统模型进行形式化描述,并实现某条性质的证明。本文给出的是一电梯系统的客户需求,使用电梯系统作为本文的具体范例的原因是由于电梯系统已经成为软件工程方法学的几种重要范例之一。本文的方法同样适用于对大型、复杂系统的正确性

进行形式化分析和验证。

对于一个有 n 层的大厦,其电梯系统规格需求如下。

电梯内部有一组按钮,每个按钮用于指示一个楼层。当按下一个按钮时该按钮指示灯亮,同时电梯驶向相应的楼层,到达按钮指定的楼层时指示灯熄灭;除了最高层和最低层以外,每个楼层都有两个按钮:一个用于请求电梯向下移动,一个用于请求电梯向上移动。这些按钮在被按下后,相应的指示灯就会发亮。当电梯按照所请求的方向移动到该楼层时,按钮的指示灯自动熄灭;当对电梯没有请求时,它关门并停在当前楼层。

下面笔者证明系统具有如下性质:当在某楼层按下按钮后,必存在某一时刻,电梯到达该层。不妨假设用户在 m 层按请求向下的楼层按钮。使用线性时序逻辑描述该性质可以形式化表示为

$G(User, pressfloorbutton(m, down)) \rightarrow \Diamond G(Lift, stop(m, down))$。

其中 $G(User, pressfloorbutton(m, down))$ 表示用户在 m 层按向下的楼层按钮,$G(Lift, stop(m, down))$ 表示电梯停在 m 层,并将向下移动。图 2 为与该性质相对应的 UML 顺序图。

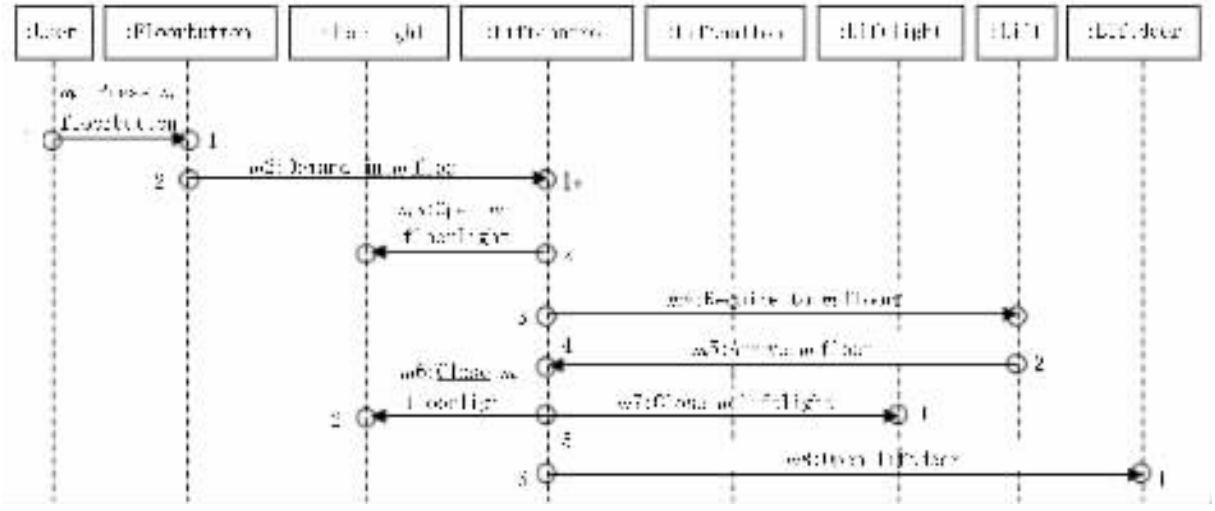

图 2　用户按下 m 层楼层按钮后的顺序图

笔者对该顺序图进行形式化描述,图中的对象有 User、Floorbutton、Floorlight、Liftcontrol、Liftbutton、Liftlight、Lift 和 Liftdoor,为了书写方便,把它们分别记为 u、fb、fl、lc、lb、ll、l 和 ld。

$SD = <Obj, Msg, Loc, F>$

$Obj = \{u, fb, fl, lc, lb, ll, l, ld\}$

$Msg = \{m1, m2, m3, m4, m5, m6, m7, m8\}$

$Loc = \{<u,1>, <fb,1>, <fb,2>, <fl,1>, <fl,2>, <lc,1>, <lc,2>, <lc,3>, <lc,4>, <lc,5>, <lc,6>, <ll,1>, <l,1>, <l,2>, <ld,1>\}$

$F(m1,s) = <u,1>$, $F(m1,r) = <fb,1>$, $F(m2,s) = <fb,2>$, $F(m2,r) = <lc,1>$, $F(m3,s) = <lc,2>$, $F(m3,r) = <fl,1>$, $F(m4,s) = <lc,3>$, $F(m4,r) = <l,1>$, $F(m5,s) = <l,2>$, $F(m5,r) = <lc,4>$, $F(m6,s) = <lc,5>$, $F(m6,r) = <fl,2>$, $F(m7,s) = <lc,5>$, $F(m7,r) = <ll,1>$, $F(m8,s) = <lc,6>$, $F(m8,r) = <ld,1>$。

在对该顺序图进行语义描述时,将其分为两种情况。

(1) 电梯停在 m 层。用户按下 m 层请求向下的楼层按钮后,楼层按钮向电梯控制器发送请求,由于电梯停在 m 层,电梯控制器只需对电梯门发送一个开门的消息即可。其语义的线性时序逻辑描述如下。

$G(u, pressfloorbutton(m, down)) \wedge m1 \wedge G(fb(m, down), off) \rightarrow \bigcirc G(fb(m, down), press)$

$G(fb(m, down), press) \wedge m2 \wedge G(lc, formerstate) \rightarrow \bigcirc G(lc, demand(m, down))$

$G(lc, demand(m, down)) \wedge m8 \wedge G(ld, close) \rightarrow \bigcirc G(ld, open)$。

(2) 电梯不在 m 层。用户按下 m 层请求向下的楼层按钮后,楼层按钮向电梯控制器发送请求,电梯控制器查询到电梯不在 m 层,则电梯控制器需对楼层按钮灯发送一个开灯的消息。该语义的形式化描述如下。

$G(u, pressfloorbutton(m, down)) \wedge m1 \wedge G(fb(m, down), off) \rightarrow \bigcirc G(fb(m, down), press)$

$G(fb(m,down),press) \wedge m2 \wedge G(lc,formerstate) \rightarrow \bigcirc G(lc,demand(m,down))$

$G(lc,demand(m,down)) \wedge m3 \wedge G(fl(m,down),off) \rightarrow \bigcirc G(fl(m,down),on)$。

现在笔者证明:当 m 层有请求电梯向下移动的需求后,在某一时刻,电梯向下移动并将要到达 m 层,即 $G(lc,demand(m,down)) \wedge G(l,formerstate) \rightarrow \Diamond G(l,come(m,down))$。电梯的 $formerstate$ 状态可以分为以下几种情况。

① **若电梯停在 k 层($k \neq m$),则** $G(lc,demand(m,down)) \wedge G(l,stop(k,nodirection)) \rightarrow \bigcirc G(l,come(m,down))$。

② **若电梯在 k 层向下移动($k > m$),则** $G(lc,demand(m,down)) \wedge G(l,come(k,down)) \rightarrow \bigcirc G(l,come(m,down))$。

③ **若电梯在 k 层向上移动($k > m$),则终有一时刻电梯到达某一层后,向下移动。即** $G(lc,demand(m,down)) \wedge G(l,come(k,up)) \rightarrow \Diamond(\exists max: k \leqslant max \leqslant n \wedge G(l,come(max,up)))$;$G(l,come(max,up)) \rightarrow \bigcirc G(l,stop(max,down))$;$G(l,stop(max,down)) \rightarrow \Diamond G(l,come(m,down))$。

④ **若电梯在 k 层向上移动($k < m$),则** $G(lc,demand(m,down)) \wedge G(l,come(k,up)) \rightarrow \Diamond(\exists max: \{k \leqslant max \leqslant n\} \wedge G(l,come(max,up)))$。

a. **当** $k \leqslant max < m$ **时,则** $G(l,come(max,up)) \rightarrow \bigcirc G(l,stop(max,up))$;$G(l,stop(max,up)) \rightarrow \bigcirc G(l,come(m,up))$;$G(l,come(m,up)) \rightarrow \bigcirc G(l,come(m,down))$。

b. **当** $m \leqslant max \leqslant n$ **时,则** $G(l,come(max,up)) \rightarrow \bigcirc G(l,stop(max,down))$;$G(l,stop(max,down)) \rightarrow \Diamond G(l,come(m,down))$。

⑤ **若电梯在 k 层向下移动($k < m$),则** $G(lc,demand(m,down)) \wedge G(l,come(k,down)) \rightarrow \Diamond(\exists min: 1 \leqslant min \leqslant k \wedge G(l,come(min,down)))$;$G(l,come(min,down)) \rightarrow \bigcirc G(l,stop(min,up))$。

此时,电梯在 min 层向上移动,$min < m$,与情况④的证明方法类似。

因此, $G(lc,demand(m,down)) \wedge G(l,formerstate) \rightarrow \Diamond G(l,come(m,down))$ **即被证明。**

当电梯将要到达 m 层并向下移动时,电梯控制器在位点 $<lc,3>$ 对电梯发送停在 m 层的消息。电梯停在 m 层后向电梯控制器发送消息,电梯控制器分别对楼层按钮灯和电梯灯发送关灯的消息。最后,电梯控制器向电梯门发送开门的消息。

$G(lc,demand(m,down)) \wedge m4 \wedge G(l,come(m,down)) \rightarrow \bigcirc G(l,stop(m,down))$;

$G(l,stop(m,down)) \wedge m5 \wedge G(lc,demand(m,down)) \rightarrow \bigcirc G(lc,ready(m,down))$;

$G(lc,ready(m,down)) \wedge m6 \wedge G(fl(m,down),on) \rightarrow \bigcirc G(fl(m,down),off)$;

$G(lc,ready(m,down)) \wedge m7 \wedge G(ll(m),formerstate) \rightarrow \bigcirc G(ll(m),off)$;

$G(lc,ready(m,down)) \wedge m8 \wedge G(ld,close) \rightarrow \bigcirc G(ld,open)$。

性质: $G(u,pressfloorbutton(m,down)) \rightarrow \Diamond G(l,stop(m,down))$ **即被证明。**

用户在 m 层按下请求向上的楼层按钮后的情况,与上面的方法类似,同理可证。

综上所述,即可证明系统具有如下性质:当在某楼层按下按钮后,必存在某一时刻,电梯到达该层。

4 结束语

作为标准的对象建模语言,UML 提供了多种模型图来描述系统分析和设计的结果。然而,由于 UML 具有非形式化的特征,使其难以对模型的一致性和正确性进行形式化的分析和证明。本文对 UML 顺序图进行了形式化描述,并对模型的某条性质进行了证明,为进一步形式化分析打下了基础。与 Z、B、CSP、ASM 等其他形式化方法相比较,用线性时序逻辑描述的语义更加直观自然[7]。下一步的工作主要是利用本文给出的顺序图的形式化描述对模型进行分析和验证,从而进一步完善。

参考文献:

[1] 李景峰,李琰,陈平. UML 顺序图的形式化描述 [J]. 计算机科学,2002,29(6):147-48.

[2] 戎玫,张广泉. 形式化与可视化相结合的软件体系结构描述方法研究 [J]. 计算机科学,2005,32(4):205-208.

[3] 戎玫,张广泉. 模型检测新技术研究[J]. 计算机科学,2003,30(5):102-104.

[4] 黄陇,于洪敏,陈致明. UML 顺序图的结构化操作语义研究 [J]. 计算机应用,2005,25(2):359-361.

[5] 张海藩. 软件工程导论(第四版) [M]. 北京:清华大学出版社,2003.

[6] 张广泉,戎玫. 并发系统性质描述的一种形式化方法 [J]. 重庆师范大学学报(自然科学版),1998,15(1):11-14.

[7] 朱雪阳,唐稚松. UML 活动图的时序逻辑语义 [J]. 计算机研究与发展,2005,42(9):1478-1484.

(责任编辑 游中胜)