文章编号: 1001- 2486(2003) 02- 0099- 05

HLA 中的所有权管理及其改进*

刘宝宏, 黄柯棣
(国防科技大学机电工程与自动化学院, 湖南 长沙 410073)

摘 要: 介绍了分布式系统中所有权管理的概念和描述方法, 总结了对象和属性所有权的基本性质并对所有权管理的方法进行了系统的分类。分析了 HLA 中的所有权管理及其不足, 并提出了改进后的所有权管理方法。提出了两种新的所有权管理管理方案, 增加了对对象所有权管理的支持, 解决了不可分对象组所有权转移的原子性问题。

关键词: 所有权; 所有权管理; HLA
中图分类号: TP391. 9 文献标识码: A

The Ownership Management and Its Improvement in HLA

LIU Bao-hong, HUANG Ke-di
(College of Mechatronics Engineering and Automation, National Univ. of Defense Technology, Changsha 410073, China)

Abstract: The concept and description of the ownership management in the distributed collaboration system were proposed. The basic properties of objects / attributes ownership were summarized systematically. Then based on this knowledge, categories of the ownership management methods from different aspects were obtained. We discussed the limitation of HLA ownership management and improved the scheme. In our scheme, we proposed two new ownership management methods, provided the support to the object ownership management and solved the problem of atom attributes ownership management.

Key words: ownership; ownership management; HLA

在分布式对象系统中, 所有权的概念有双重含义: 第一, 从访问控制和系统安全的角度讲, 对象的所有权是指该对象归谁(个人或组织) 所有。只有具有相应的所有权, 才能运行或访问该对象实例或对该对象的实现进行修改。第二, 从信息处理的角度讲, 对象的所有权是指在协作联邦中一个对象实例由哪个成员负责运行, 属性的所有权是指一个对象的属性由哪个成员负责计算和更新[1]。

本文将从信息维护的角度讨论所有权管理问题。通过有效的所有权管理可以使分布式系统中的各个成员更好地进行协作, 更自然地对实际问题进行建模, 同时可以对系统资源进行更为合理有效的配置。

1 问题的提出

在一个基于 HLA 的仿真系统中经常会遇到如下情形: 某个对象 M 的属性 S, 首先由成员 A 进行计算, 然后再由另一个成员 B 进行计算, B 计算后将结果公布出去, 供其他成员使用。在这里 A、B 所计算的是同一个对象的属性。A 的计算结果只是中间结果, 仅供 B 使用。类似的情况在分布式协作建模中是很常见的。这是所有权转移问题。

另一种典型的情况是: 一个对象可能会扮演不同的角色, 与每个角色对应的是不同的对象属性, 这些属性可能会分布在不同的成员中。这是对象所有权的分布问题。在 HLA 的实现中, 对象由某个成员创建, 创建后它可能仅拥有该对象的部分属性。当其他成员发现该对象已创建时, 通过接口服务获得它们各自公布的对象属性的所有权。

* 收稿日期: 2002- 09- 25
基金项目: 国家部委预研基金资助(413040403)
作者简介: 刘宝宏(1974—) , 男, 博士生。

以上这些情况在分布式仿真系统中是十分常见的。所以开展所有权管理方面研究具有重要意义。

2 对象和属性所有权描述及其基本性质

用 F 表示成员, 记一个联邦为 $FN=\{F_1, F_2, \cdots F_m\}$, F_i, $i=1, 2, \cdots m$, 为联邦 FN 的联邦成员。设 T 为系统运行时间集合, $t \in T$, 为任意的时刻。

为描述方便不考虑对象的继承、包容等特性, 将一个对象类描述为如下 3 元组:

$$CLS::=\{NM, AT, BH\}$$

其中, NM 表示类的名称; AT 表示类的属性集合; BH 表示类的行为集合。

在所有权意义下, 一个对象可以描述为:

$$OBJ::=\{ID, CLS, AOL, OWNS\}$$

其中, ID 表示对象标识; CLS 表示对象所属的类; $OWNS$ 表示对象的所有权; $AOL \subset AT \times FN$ 表示对象属性到成员之间的映射, 其中的每个元素表示对象 OBJ 的一个属性的所有权为某个成员所有。

不考虑对象的行为, 可将一个对象简记为: $Obj=\{a_1, a_2, \cdots a_n\}$, a_i 为对象的属性。CLS_{obj} 为对象 Obj 所属的对象类。$CLS_{obj}=\{a_1, a_2, \cdots a_n\}$, 为类的属性集。一个属性 a 表示的是对象的属性还是类的属性可以通过上下文很容易地区分。如果 Obj 的属性 a_i 由成员 F 负责公布并进行属性更新, 我们说 F 拥有对象 Obj 的属性 a 的所有权, 记作 $Obj(a) \succ F$。F 不拥有对象 Obj 的属性 a 的所有权记作 $Obj(a) \neg\succ F$。F 试图放弃对象 Obj 的属性 a 的所有权, 记作 $Obj(a) \searrow F$。F 试图获取对象 Obj 的属性 a_i 的所有权, 记作 $Obj(a) \nearrow F$。$CAN\_PUBLISH(CLS(a), F)$ 表示成员 F 可以公布类 CLS 的属性 a。$CAN\_UPDATE(CLS_{obj}(a), F)$ 表示成员 F 可以更新对象 CLS 的实例 obj 的属性 a, $ID(obj)$ 表示取对象 obj 的标识。

如果属性 $a_i, a_{i+1}, \cdots, a_j (i<j)$ 是不可分的, 称这些属性构成对象 Obj 的一个不可分属性集或紧耦合属性集, 记作 $Atom=<a_i, a_{i+1}, \cdots, a_j>$。

采用上面的记号, 对象及其属性的所有权具有如下性质:

1) $\forall t$, *if* $Obj(a_i) \succ F_j$, *then* $Obj(a_i) \neg\succ F_s$, $s=1, 2, \cdots, m \wedge s \neq j$

2) *if* $(Obj(a) \succ F) \wedge (a \in Atom_i)$, *then* $a_k \in Atom_i \rightarrow Obj(a_k) \succ F$

3) $\exists t, \exists a \in Obj(a \neg\succ F_i), F_i \in FN$

4) $\neg \exists Obj, Obj \neg\succ F_i, F_i \in FN$

5) $CAN\_PUBLISH(CLS(a), F_j) \wedge (Obj(a) \nearrow F_j) \wedge (((Obj(a) \succ F_i) \wedge Obj(a) \searrow F_i) \vee Obj(a) \neg\succ FN) \rightarrow OBj(a) \succ F_j$

6) $ID(Obj(a) \succ F_i) \equiv ID(Obj(a) \succ F_j), ID(Obj \succ F_i) \equiv ID(Obj \succ F_j)$

7) $Obj(a) \succ F_i \Leftrightarrow CAN\_UPDATE(F_i, CLS_{obj}(a)) \wedge CAN\_PUBLISH(F_i, CLS_{obj}(a))$

8) $\exists A_i \subset Obj (i=1, 2, \cdots p$ 且 $p<n), \bigcup A_i = Obj$ 且 $A_i \cap A_j = \Phi (i \neq j)\ s.t.\ A_i \succ F_i, F_i \in FN$

以上是分布式协作建模中对象和属性的所有权的基本性质, 是一般的所有权管理方案都要遵守的。

3 所有权管理的实现

3.1 所有权管理方法的分类

(1) 强制型和协商型。强制型是指所有权的拥有者强行放弃其所有权, 而不管有无接收者。协商型是指通过协商进行所有权的交接[1]。

(2) 推模式和拉模式。推模式是指由欲放弃所有权的一方发起所有权转移过程。拉模式是指由欲获取所有权的一方发起所有权转移过程[1]。

(3) 对话式和竞争式。在竞争式所有权管理中, 一个成员出让某些属性的所有权, 所有需要该属性所有权的成员竞争该属性的所有权。在对话式所有权管理中, 所有权的转移通过对话实现[2], 例如, 成员 A 想要将属性 a 的所有权转移到成员 B, 则它直接询问成员 B 是否需要该属性的所有权, 如果 B 给

出肯定的回答, 则 A 将属性 a 的所有权转移给 B, 否则 A 继续持有 a 的所有权或进一步协商。

3.2 HLA 对所有权管理的实现

HLA 引入所有权管理的主要目的是为了支持在分布式联邦系统中对给定的对象实例进行协作建模, 另外通过所有权管理可以更好地实现现实世界到模型的映射, 简化模型的实现和进行负载平衡[3,4]。HLA 中加入到联邦中去的联邦成员通过所有权管理服务在成员之间转移对象实例属性的所有权。也就是说, 在 HLA 中所有权管理是在对象属性层面上进行的。一个对象的不同属性可以分布在多个成员之中。在仿真运行过程中, 属性的所有权可以从一个成员转移到另一个成员, 但在任意时刻, 一个属性只能由一个成员所有。HLA 的所有权管理方案分为推模式(Push Mode) 和拉模式(Pull Mode) 两种[1, 5]。其中推模式又分为强制型所有权转移和协商型所有权转移。图 1 和图 2 分别给出了协商型推模式和协商型拉模式的交互图[5]。强制型的调用关系比较简单, 这里不再赘述。

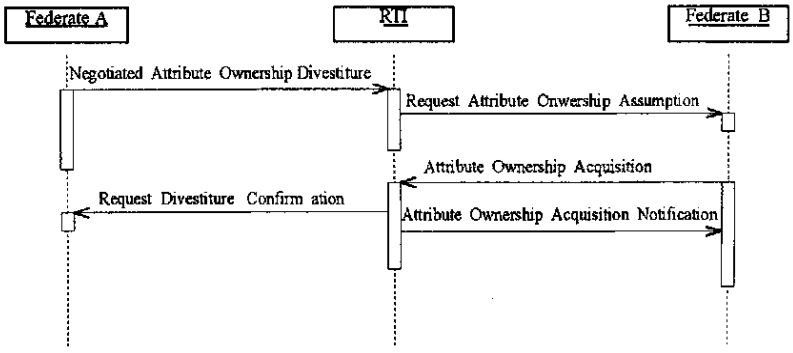

图 1 协商型推模式所有权转移交互图

Fig. 1 Negotiated ownership push interaction

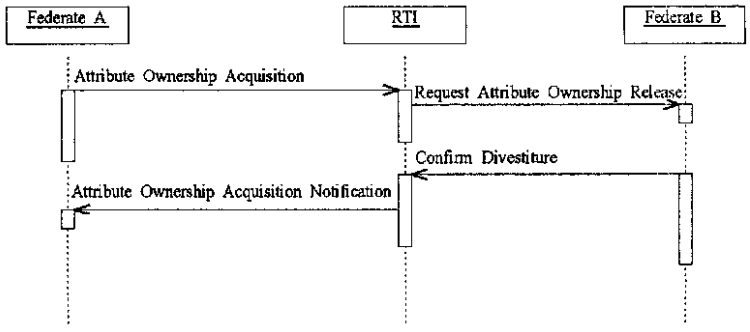

图 2 协商型拉模式所有权转移交互图

Fig. 2 Negotiated ownership pull interaction

3.3 HLA 所有权管理方案的不足

HLA 仅仅给出了在分布式建模与仿真中初步的所有权管理方案, 该方案集中于对象属性的所有权管理。目前该方案还有很多不足[1, 2]:

(1) HLA 的接口规范中的所有权转移是基于对象的属性的, 并不支持对象所有权的转移。但在某些情况下直接进行对象所有权的转移可能更直观更方便些。在这种情况下, 对象并不总是属于某一个成员, 根据需要, 它可以将整个对象的所有权转移给另一个对象。

(2) HLA 接口规范并没有描述如何处理多个成员同时要求一个对象属性所有权的情况, 最终到底哪个对象获得其所有权是不定的。它所支持的是竞争式的属性所有权获取, 在这种方式下, RTI 有时不能把所有权传给真正需要该属性的成员。

(3) HLA 没提供对不可分属性集的处理。由于采用竞争式所有权管理, 当有多个成员都对一个对

象的不可分属性集中的某些属性感兴趣时,一个对象的不可分属性集中属性的所有权可能被转移到不同的成员。这就违反了所有权的属性。

(4) HLA 的所有权管理规范并不支持对话式的所有权交接,即并不存在直接指明要把所有权转移给哪个成员的方法,这使得 RTI 有时不能把所有权传给真正需要该属性的成员。

3.4 改进的所有权管理方案

为了克服 HLA 中的所有权管理方案的不足,在 HLA 所有权管理的基础上提出了新的所有权管理的方案。该方案解决了原有所有权管理方法的不足,并增加了对象所有权管理方案。

3.4.1 对话型所有权管理

将前面的所有权管理分类进行组合,就产生了 8 种组合。但是,强制型只能是推模式的竞争式。因为对话式肯定是协商型,同时在拉模式中一个对象要想获得对象或属性的所有权,就一定要和其当前的所有者进行协商。对话式主要体现在两个对象之间就所有权问题进行协商。所以实际上共有 5 种所有权转移方式。HLA 不支持对话型所有权管理。所以,在 HLA 的基础上增加了两种所有权管理方式:协商型对话式推模式和协商型对话式拉模式。通过对话式所有权管理可以解决前面谈到的 HLA 所有权管理的第 4 条不足,并丰富了所有权管理的语义。但是需要注意的是对话型所有权管理的引入增加了成员之间的耦合性,这一点在具体应用时是需要认真考虑的。

3.4.2 对象所有权转移

一个完善的所有权转移方案应该提供对对象所有权转移的支持,为此设计了对象所有权管理方案。如果对象比较小,采用移动 Agent 方法可能是一个比较好的实现,这一点已经有很多讨论,这里不再赘述。但是,在实际的分布式建模与仿真系统中,对象往往都是比较大的,所以很多系统并不适合用移动 Agent 的方法来实现。一个较为现实的方案是借助于 HLA 的所有权管理,在此基础上寻找解决方案。在分布式仿真中,在仿真之前各成员可以将相关对象类的实现下载到本地。这样就可以根据需要创建对象所有权转移时要生成的实例。然后为每个对象添加一个属性:ObjectOwner,该属性表示对象的所有权。在对象所有权转移之前先转移该属性的所有权,获得其所有权的成员发送一个回应,欲转移所有权的成员利用回应信息转移整个对象的所有权到指定的成员。接收方负责公布对象属性并进行属性的更新。而原来拥有该对象的成员不但要删除本地对象,还要向 HLA 的运行时间支撑框架 RTI 注销它所公布的关于该对象的信息。分两步走的主要目的是减少因盲目转移产生的系统性能损失。

3.4.3 不可分属性组的原子性保证

在一个对象中,其中某些属性是密切相关的,如一个实体的位置和速度,称这些属性为不可分属性集或紧耦合属性集。如前所述,一个好的所有权管理方案应该能够保证这些不可分属性的所有权在同一个成员内。这一要求一是为了保证属性间逻辑上的关联,二是为了减少网络上的数据量。在具体实现时,如果是对话式所有权转移,可以通过指定紧耦合属性组的所有权接收者的方式来保证其原子性,可见通过引进对话式所有权转移同时解决了所有权转移的原子性问题。对于竞争型所有权转移,实现起来要麻烦一些。这时所有权转移给谁是无所谓的,只要有一个成员全部接收即可。此时可以先将属性组中的一个属性所有权转移,然后再将属性组中的其它属性转移到第一个属性所有权的接收者。

3.4.4 实现中的几个问题

从实现上讲,我们的所有权管理方案至少有三种实现方式:

第一,通过移动 Agent 的方式来实现。尤其是对象的所有权转移,当对象较小时比较适合用该方法实现。

第二,在 HLA 所有权管理的基础上实现。在 HLA 的所有权管理接口标准中,有些接口有一个称为 User Supplied Tag 的参数,该参数主要是用来在成员之间传递优先级和其它一些配置信息。可以利用该参数来实现改进的所有权管理。例如,在实现推模式的对话式所有权转移时可以通过将该参数设为所有权接收方的成员句柄来实现。通过对这些接口进一步包装可以为用户提供更为直观的接口。

第三,通过交互的方式来实现。该方法实现起来比较灵活,可以实现任意的所有权管理方案。例如,当一个成员要放弃某个对象的所有权时,可以定义一个名为 ObjOwnershipTransRequest 的交互,其

格式如下:

```
……
(interactions
 (class InteractionRoot  reliable timestamp
      (class RTIprivate  reliable timestamp)
  (class ObjOwnership TransRequest best_effort receive
    (parameter Sender)
    (parameter Receiver)
    (parameter ObjectHandle)
    (parameter ObjectName)
    (parameter ClassName)
  )
……
```

这种方法的不足是缺乏通用性,重用性比较差,每个应用都要定义各自的用于所有权管理的交互。所以该方法主要用于标准的所有权管理服务,不能提供具有特殊要求的所有权管理场合。

4 结束语

目前已经有一些研究人员对分布式协作系统中的所有权管理进行了研究与实现,其中以 HLA 中对所有权管理的研究最具代表性。不过总的来看,对所有权管理的研究还缺乏系统性。本文系统总结了对象和属性的所有权的基本性质,给出了所有权管理方案的分类,分析了 HLA 中所有权管理方案的不足,并提出了改进后的方案。方案中引入了对话型所有权管理,比较直观,应用起来比较自然,但是可能会增加成员之间的耦合性。同时引进了对象的所有权转移方案,并解决了紧耦合属性组所有权转移的原子性问题。改进的所有权管理可以应用到物流管理、分布式多分辨率建模中。

参 考 文 献:

[1] IEEE Std. 1516.1. IEEE Standard for Modeling and Simulation (M&S) High Level Architecture (HLA). Federate Interface Specification[S]. 2000(9).

[2] Geoff Sauerborn, Gary Moss, et al. HLA Ownership Management Services: We Almost Got it Right[C]. In the processing of 2000 Fall simulation Interoperability Workshop.

[3] Marco Brass, Nico Kuijpers. Realizing a Platform for Collaborative Virtual Environments based on the High Level Architecture[C]. In the processing of 2000 Spring simulation Interoperability Workshop.

[4] Hessam S Sarjoughian, Bernard P Zeigler. Models and Representation of Their Ownership[C]. In the processing of 2000 Winter Simulation International Conference.

[5] DMSO. RTI1.3- Next Generation Programmer's Guide[Z]. Sept. 1999. http://www.dmso.mil.