CSCD 环境下协作设计过程的一种建模方法及应用*

A Modeling Method for Cooperative Designing Procedure in CSCD Environment and Its Application

陈国宁,李陶深

Chen Guoning, Li Taoshen

(广西大学计算机与信息工程学院,广西南宁 530004)

(College of Comp. and Info. Engi., Guangxi Univ., Nanning, Guangxi, 530004, China)

摘要:采用面向对象技术分析协作设计活动,使用状态图和 petri 来描述协作设计活动的动态特性,建立相应的模型,在此基础上获得 1 个支持协作的事务模型,得到对应的系统结构。

关键词:协作设计 建模 活动模型 设计事务 petri

中图法分类号:TP311

Abstract: A modeling method of designing activity in CSCD environment based on object-oriented method is introduced. The activity model is built and the state diagram and petri network are used to illustrate the dynamic characteristics of cooperative activities. The transaction model corresponding to the activity model is also presented.

Key words: CSCD, modeling, activity model, designing transaction, petri

计算机支持的协同工作(CSCW)技术的出现为现代人工作中更高质量的协作提供了强有力的技术支持,而计算机支持的协作设计(CSCD)则是 CSCW 领域的一个重要分支[1]。CSCD 环境具有开放性、持续时间长、交互性强等特点,尤其是共享对象在协作成员之间的频繁交互对设计数据管理提出了更高的要求,以数据库管理方式对这些共享数据进行管理和维护是一种可行的方法。将设计数据存储在数据库中,以事务处理的方式来访问设计数据是这类系统的主要特征。为了能保证在设计过程中维护设计数据完整性的同时,又要尽可能提高协作的效率,有必要寻找一种适用于协作应用环境,尤其是 CSCD 环境的事务模型。

为获得正确且适用的协作事务模型,首先应该对协作设计过程进行正确的分析和建模。目前针对 CSCD 应用开发的从协作过程分析到获得相应的事务模型的完整方法还很少见,为此,本文根据已有的协同活动和并行产品开发活动建模[2,3],使用面向对象方法的成功经验,结合系统实际应用环境的需要,尝试对协作活动进行分析,并给出其从任务划分到设计活动的完整的设计过程模型,然后在此模型的基础上获得 1 个支持协作的设计事务模型,以此来探讨由分析协作设计过程到映射事务模型到确定此类系统的体系结构的方法。

1 面向对象方法与活动的建模

面向对象分析方法是当前大多数大型系统的分析和设计方法,它通过将数据以及在这些数据上进行的操作进行封装和隐藏,使得变化局部化,从而使系统分析简单而清晰。同时面向对象技术通过继承机制和多态性使得人们更容易发现和利用各种关联对象的本质特性和联系,并用于构造更复杂的系统,因此面向对象方法更符合人们认识世界的过程。活动与面向对象技术有很大的相似性[3]。活动从本质上来看就是信息的处理过程,它包括被处理的信息和在这些信息上进行的相关处理(即操作);1 个活动可以由若干低级的活动组成。这些概念与面向对象的基本概念是类似的,而设计事务是实际设计活动在数据库处理范畴内的映射,因此使用面向对象方法可以更好地对活动和设计事务进行描述和建模。

2 协作设计过程模型简介

根据抽象级别以及在协作设计过程中所处的地位,将设计过程模型简单地分为元类层、任务划分

2003-04-21 收稿,2003-09-11 修回。

* 广西大学科研基金(X022089)资助项目。

层、协作设计活动层 3 个层次。

2.1 元类层

元类指的是可以根据实际应用的需要派生出子类，且它们本身不能由其他类派生而来的类。在本文的过程模型中，元类有：协作类、消息类、项目类、活动类、角色类等。

(1)协作类用于封装不同协作类型的规则以及相应的约束函数。对于不同的应用环境可能会需要不同类型的协作。以 CSCD 中小组成员间的协作为例，通常有如下 3 种形式[1,2]：①消息共享。由于对某一共享数据的并发操作而引起的 1 种隐式的协作，即通常所说的数据相关性。②同步共享协作设计。是多个协作设计人员同时在一共享空间(如白板)中协作完成某一子任务的设计的 1 种显式的协作。③对象共享。是指设计人员授权同组中的另一协作者访问其部分中间结果。不同的协作类型所采用的协作规则是不一样的，与此对应的是并发控制算法的不同，这一问题将在稍后介绍。

(2)消息类用于封装设计活动进行过程中传递的消息的属性(包括消息格式、源/目的地、消息内容等)和相应的消息收/发方法。例如，在本文的系统中使用的消息机制主要有 2 大类，一是主动消息，即消息“推”机制，类似于通知；二是被动消息，即消息“拉”机制，类似于查询。于是我们又可以由消息类派生出 2 个子类，同时还可根据需要由主动消息派生出消息广播类。

(3)角色类用于封装设计活动中，设计成员对操作对象的访问权限和访问控制。在简单的设计活动中，角色主要分为小组负责人、小组成员和组外成员[7]，它们可由角色类派生而来。

(4)项目类用于封装所有设计项目都可能具有的一些基本的属性和操作，如产品属性、项目组织和开发规则、开发目标等，是所有其他项目类的抽象基类。

(5)活动类用于封装活动类对象所拥有的共同特征，如活动标识、活动的状态信息、活动约束、操作对象以及标志活动开始或结束等的活动元操作。活动类是所有其他活动类的抽象基类。

2.2 协作任务的划分

任务划分是所有大型项目设计开发必不可少的一步，任务划分的好坏直接影响到后续设计活动的顺利与否，而任务的划分同时也将影响到活动的分割，因此首先给出设计任务划分的模型。1 个设计项目根据需要可分解成若干个子项目，而子项目还可分解成更小的子项目，如此递归进行，直至根据相关程度无法再分，最终形成了一棵项目任务划分树。这里只讨论三级划分，其中树根为该设计项目 WP，中间结点为各级子项目 SWP，叶结点为用户任务 UDW。

根据面向对象分析的方法和原则，任务划分树中的每一个结点都应看成一个实体(即对象)，其中包含了属性以及对这些属性进行的相关操作。对于设计项目、子项目和用户任务对象而言，其属性包括协作类型、角色、资源、消息和规则等的相关信息[2]，而操作则包括数据操作、与规则对应的约束函数、存储方法、通信方法等[3]。在相应的任务划分模型中，任务划分是 1 个至上而下的过程，位于树根的是项目元类，而下层的子项目类在拥有设计项目对象规定的所有属性(除具体任务外)和操作的前提下，可根据实际需要进一步扩充完成本子项目所需的成员、资源、本小组开发的具体实施规则以及子任务合并方法等属性和操作。同样，用户任务在接受子项目的所有属性和操作的同时，也可以扩充具体的用户任务属性和所需资源等内容。因此，设计项目、子项目和用户任务之间实际上是继承和派生关系。于是得到任务分解模型如图 1 所示。

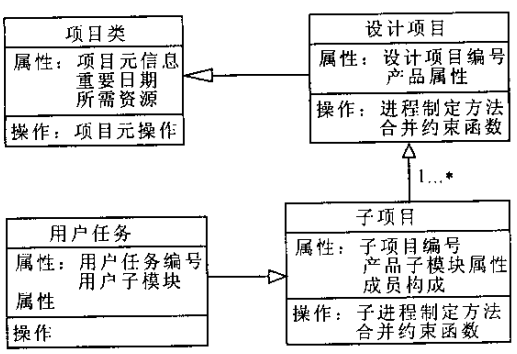

图 1 项目任务划分模型

2.3 协作设计活动模型

设计活动是为完成设计项目而进行的一系列对工程数据对象的操作，根据设计项目的层次划分，设计活动也相应的具有层次性。1 个项目设计活动 WPA 可由 1 个或多个子项目设计活动 SWPA 组成，1 个子项目设计活动又可由 1 个或多个用户设计活动 UDA 组成。一般只有处于叶结点的用户设计活动承担具体的设计任务，如数据访问、设计数据更新等，而处于中间结点及根结点的子项目活动与项目活动只负责协调工作与集成测试。

根据上述任务划分模型得到相应的设计活动模型如图 2 所示。其中项目设计活动、子项目设计活

动、用户设计活动分别从任务划分模型中的设计项目类、子项目类和用户任务类继承所有完成这些任务所需的属性和操作，从而使相应的活动在原先划分好的项目（任务）范围内进行，并按项目要求来验收。

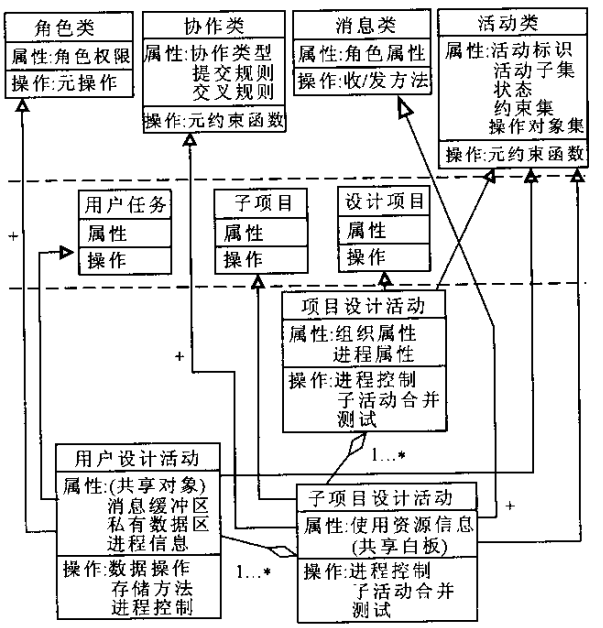

图 2 设计活动结构模型

活动模型中常用的属性如图 2 所示。

（1）活动标识，用于唯一的标识 1 个活动。项目设计活动、子项目设计活动、用户设计活动的标识分别为 WPA _ ID、SWPA _ ID、UDA _ ID。

（2）活动子集，用于表示该活动由哪些子活动组成，通常是子活动标识的集合。如项目设计活动的活动子集可写为

WPA. ＜child set＞::＝SWPA _ ID^{+} | UDA _ ID*。

（3）活动状态，表示活动的当前状态。其中用户设计活动的当前状态为图 3 所示的状态之一。

（4）约束集，表示活动实施过程中必须遵循的规则。以用户设计活动的约束集为例，包括提交规则、交叉规则、操作对象完整性约束和用户自定义规则等。其中提交规则和交叉规则来自协作类，操作对象完整性约束和用户自定义规则可以直接作为活动类的成员。这里提交规则是指在活动完成并进行提交处理时应遵循的规则。例如在共享对象的协作方式中，发生协作的 2 个活动间必须满足提交和夭折依赖[4]，即只有在共享对象拥有者（指活动）提交，使用方才能提交，如果拥有者夭折，则使用方也要夭折。

交叉规则定义各子活动间的数据流相关性，同时也规定了属于 2 个不同用户的活动在结果合并时，是两者同时“出现”在合并结果中，还是只允许其中之一“出现”在结果中。另外，交叉规则也定义了操作的相容性，例如，在传统的读写冲突模型中，对同一数据对象的读写，写写操作对是不相容的（根据可交换性准则），而读读操作则是相容的，于是其交叉规则可表示为：$\oplus$：R(a)×W(a)→False；$\oplus$：W_1(a)×W_2(a)→False；$\oplus$：R_1(a)×R_2(a)→True[6]。

（5）操作对象集，指该活动操作的全体对象，其中每个活动都有其工作区域来存放这些对象。此外，各具体的设计活动可根据需要增加特定的属性。例如在协作产品设计中，小组成员常常要开辟 1 个空间——共享白板，共同讨论设计问题。

活动模型中的有关操作为与上述属性有关的处理和元操作，如保证某协作规则（提交/交叉规则）的协议，控制访问权限的协议，消息收发的处理等，这里不再详述。

2.4 协作设计活动的动态模型

图 2 只给出设计活动间的关系，是 1 个静态模型，为了更好地分析和描述协作设计活动，本文还应分析其动态模型，这里使用状态图来描述设计活动执行过程中的动态变化过程。图 3 对协作设计活动的状态进行简单描述，实际状态要复杂得多。

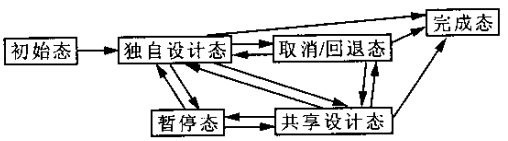

图 3 设计活动状态

为了更清晰地分析协作活动中的状态和事件，可以使用 petri 网[5]来描述协作过程。以共享对象的协作方式为例，其协作过程为：1 个设计活动开始（P0）后进行初始化（T0），然后申请所需资源（P1），当资源空闲（P3），且协作允许（ss）则开始协作处理（P2）；这时与其同组的另一设计活动向它提出使用某一对象的请求（P1’），前 1 个活动通过对象代理机制将对象使用权授予提出请求的活动（T2），这时两个设计活动都进入协作状态（P3，P2’）；当协作完成时，如果提出请求的活动先完成，为了保证使用数据正确，该活动必须先等待，直到前 1 个活动完成后（T3，P4），它才能完成（T3）——这一规定就是前述的提交依赖。图 4 所示的 petri 网描述了这一过程。其它类型的协作也可用类似的方法进行描述和分析。

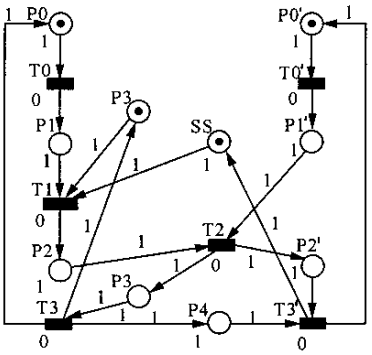

图 4 共享对象协作方式 petri 图

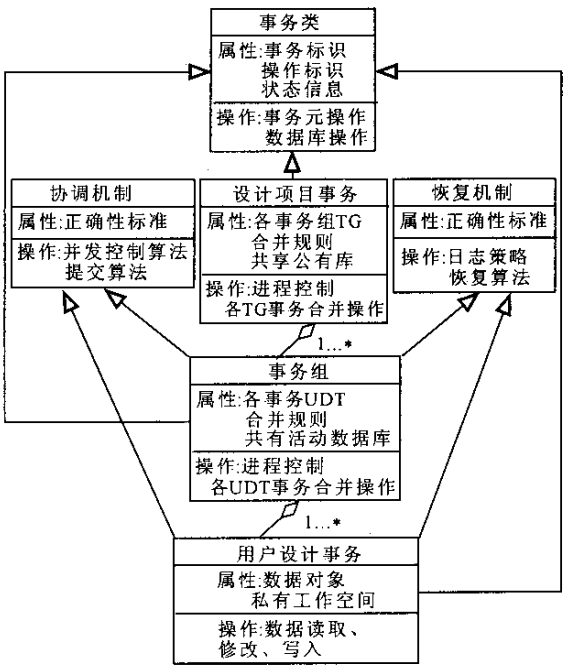

图 5 协作设计事务关系模型

3 支持协作的设计事务模型

根据上述过程模型的描述,活动模型可以直接映射到事务模型中,其映射规则为:

(1) 活动的层次结构→事务的层次结构;

(2) 活动标识→事务标识;

(3) 活动状态→事务状态(根据需要可增删部分状态);

(4) 操作对象集→私有/公有数据库;

(5) 活动的完整性(由对象约束和用户自定义约束构成)→正确性准则/恢复依据;

(6) 协作类中的交叉规则和提交规则→并发控制算法/提交算法。

所得到的事务模型如图 5 所示,它们的模型描述如下:

(1) 1 个设计项目的所有活动映射成 1 棵事务树,具有与活动模型对应的层次结构。

(2) 每个用户设计事务都有自己独立的私有工作空间,存放用户设计事务的操作对象和设计(中间)结果,未经授权任何人都不能访问他人的私有工作空间。每组协作设计事务都有一个公有活动数据库,用于存放本协作活动小组各成员的(中间)设计结果,该结果可供同一小组的成员访问。项目事务有 1 个共享公有库,用于存放本项目的全局设计数据和信息,可供参与项目的所有设计人员访问。事务对处于其上一级的共享库中数据对象的访问采用 check out/check in 机制。

(3) 协作事务是一个以 Transaction _ begin 为开始标志,Transaction _ commit 或 Transaction _ abort 为执行结果,Transaction _ complete 为终止标志的操作集合。这些标志统称为事务元操作。

(4) 协调机制采用的正确性准则根据实际项目所使用的对象完整性和用户所要求的完整性确定,一般采用语义一致性。并发控制算法和提交算法由协作类型对应的操作规则(交叉规则和提交规则)确定,如传统的读写冲突模型可采用两阶段封锁和两阶段提交协议,而共享对象协作方式则可采用支持协作的多粒度封锁协议[7]和带有时限的工程设计事务提交协议[8]。

(5) 恢复机制中的日志读写策略和恢复算法应根据正确性准则和所采用的协作类型来设计。如在共享对象的协作方式下可采用基于 savepoint 机制和日志的恢复方法[7]。

(6) 协作事务存在以下状态:

① 未执行:事务未启动的状态;

② 正在执行:事务调用 Transaction _ begin 或 Transaction _ resume 后进入的状态,为设计事务正常运行的状态;

③ savepoint 中间态:协作设计事务执行过程中的一系列语义一致状态,用户可决定在该状态下进行提交或将中间结果存入公有活动库的 savepoint 临时区;

④ 准备提交:设计事务申请提交后等待父事务(服务器)响应时的状态;

⑤ 提交成功:父事务同意提交后,设计事务所处的状态;

⑥ 准备夭折:设计事务准备取消时所处的状态;

⑦ 夭折完成:设计事务已成功取消,私有空间已被清空的状态。

根据对协作系统的分析，本文设计了 1 个支持协作设计的工程数据库事务管理子系统的功能结构，如图 6 所示。其中各子模块所实现的功能由事务模型确定，这里不再详述。

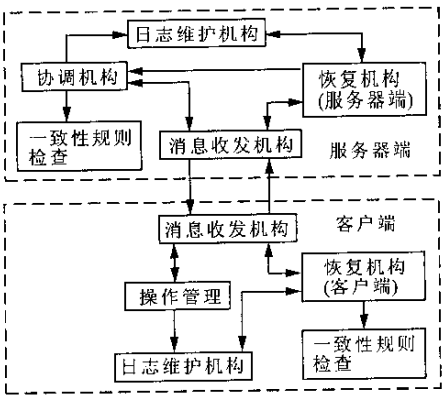

图 6 协作设计系统体系结构

4 结束语

协作工作方式已经无可争议地成为当前工作方式的主流，为了更好地建立计算机支持的协作设计环境，首先必须要对协作设计活动建模，考虑到活动的特点，本文采用面向对象的方法来分析协作设计活动，并建立相应的模型。在此基础上，获得了 1 个支持协作的事务模型，并得到对应的系统结构。

当然，本文所给出的方法和模型还比较粗糙，不够细致，活动模型到事务模型间的映射还不能做到无缝。而且本文只讨论了简单情况下的协作活动模型，对于任务（或事务）分解的一些具体要求还未做讨论，同时协作活动的动态模型过于简单，这些都是今后需要进一步完善的。

参考文献：

1 史美林，向 勇，杨光信，等. 计算机支持的协同工作理论与应用. 北京：电子工业出版社，2000.

2 李敏强，王 琛，周 静. CSCW 系统中协同机制及协同活动模型. 系统工程与电子技术，2000，22(4)：28～31，33.

3 施 群，秦现生，彭炎午. 并行工程环境下产品开发活动模型的建立与研究. 制造业自动化，2000，22(3)：28～30，33.

4 齐 进，周伯鑫，徐南荣. 支持合作过程的工程数据库设计事务模型. 计算机研究与发展，1998，35(10)：891～895.

5 左凤朝. 基于 petri 网的数据库系统并发控制模型. 计算机工程与应用，2002，13：198～200.

6 Marek Rusinkiewicz，Wolfgang Klas，Thomas Tesch，et al. Towards a cooperative transaction model-the cooperative activity model. In：Proceedings of the 21th International Conference on Very Large Databases (VLDB'95)，Zurich，Switzerland，1995. 194～205.

7 陈国宁. 支持协作设计的工程数据库事务管理的研究（硕士学位论文）. 南宁：广西大学，2002.

8 Chen Guoning，Li Taoshen，Liao Guoqiong. Commit mechanism of engineering database supporting cooperative design transaction. Proceeding of ISFST'2002，Wuhan，China. 2002，316～322.

（责任编辑：黎贞崇）

人造红血球治疗脑血栓的动物实验效果良好

一个日本科研小组最近开发出用人造红血球治疗脑血栓等疾病的新型治疗方法，已经进行的动物实验证明该方法疗效良好。

人造红血球是用脂质膜包裹的可输送氧的血红蛋白粒子，直径约 200nm，大小仅有红血球的 4%，可在因梗死而变得狭窄的血管通过，把氧输送到需要的地方。

研究人员先让实验鼠出现脑血栓症状，然后给一部分实验鼠注射人造红血球。一天后他们发现，与没有注射人造红血球的实验鼠相比，接受治疗的实验鼠脑部浮肿的区域缩小了 40%。研究人员认为，是人造红血球输氧使得病情好转。

在心肌梗死实验中，研究人员让实验鼠心脏冠状动脉发生堵塞，然后注入人造红血球，结果动脉很快就恢复畅通，血流量恢复到了正常状况下的 90%。

（据《科学时报》）