维普资讯 http://www.cqvip.com

计算机科学 1994 Vol. 21 №. 2

MIDS/BUAA 的对象数据模型及其特征

车敦仁　麦中凡
（北京航空航天大学计算机系　北京 100083）

摘要　The main characteristics of MIDS data model are 1) frame, rule and object are unified smoothly, 2) the most distinguishing feature of it is OO, 3) value-supported and strong type supported. This paper summarizes the MIDS data modeling philosophy, MIDS object model and its characteristics, and especially stresses and discusses the values and rule objects of MIDS in detail.

关键词　Data model, Multimedia, Object, Frame, Rule, Trigger, OO, Value, IDB.

1. 引言

尽管 OOP 风范站在软件开发方法学的高度几乎从形式上将“程序＝数据结构＋算法”这个经典模型予以摒弃，但是数据模型设计仍然是进行任何一个计算机系统（软件、硬件和应用）设计时无法回避的一个首要问题。面向对象开发（OOA，OOD）的本质是分析和发现问题域中的对象原型，设计与实现解空间（模型化世界）中的对象模型。

我们在进行 MIDS/BUAA 多媒体智能数据库系统的设计时，同样也把对对象数据模型的确认和设计当成首要任务去进行。所以详细论述 MIDS 的对象数据模型是本文的中心议题。

2. MIDS 对象数据模型概述

2.1　MIDS 数据模型的哲学

• MIDS 服务器是一个页服务器（Page-server[14]）。

• 提供一个单一的全局服务器进程（Single global server process[2]）。

• 多个客户进程（Client process）可在网络中随意分布。

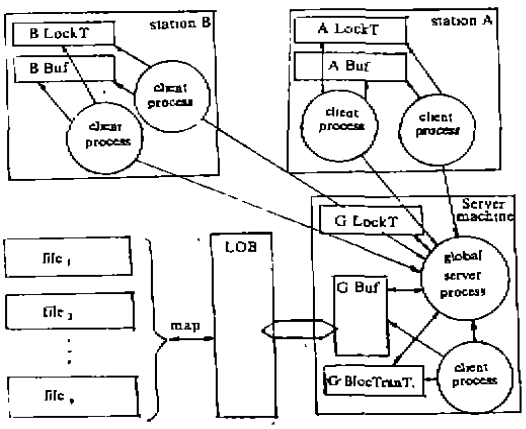

图 4　MIDS 的 Client/Server 体系结构

4. 结束语

我们在着手 MIDS 的设计之前，分析和解剖了许多著明的同类系统，如 O_2[14]，ORION[16]、ODE[13]、Vbase、HiPAC、Bubba、Objectstore[18]等，MIDS 之设计充分借鉴了各家之长，同时有自己的一系列鲜明特点。MIDS 的数据模型的最大特征是对象、框架和规则高度统一；MIDS 系统的开发方法的特点是面向对象和“自举式[11]”；MIDS 体系结构[2]的特征是：页服务器，全局的单服务器进程，分布的多客户进程；MIDS 存储结构[9]的特点是：单级存储，虚存之上的虚存（Virtul over virtual memory）和一体化对象管理[17]。

（参考文献共 18 篇略）

维普资讯 http://www.cqvip.com

在数据模型化（data modeling）中，为“关系”概念确立一个适宜的位置，是MIDS数据模型化哲学[3]的一个出发点。MIDS哲学既不绝对地认为“世界仅由对象组成”，也不绝对地认为现实世界的任何一个关系实体，在模型化世界中都必须通过另一个与对象模型平行的“关系”模型才能模拟。

MIDS哲学的基本点是：把对象之间的关系划分为二个范畴，四种基本的对象关系[3]IKO，IIO，IPO和MS通讯关系属于第一范畴，它们是系统的内定义机制；对象间的引用关系属于IPO的一种，直接通过UID和指针便可摸拟；除此以外的对象关系属于第二范畴，第二范畴关系通常必须借助于一个专门定义的“relationship”类的对象去模拟，这个“relationship”类与普通的对象类无本质差别。

对于MIDS/BUAA来说，坚持MIDS哲学有两个重要意义：第一，MIDS最终要向用户提供一个智能化的面向对象开发环境HLTools，并要求用户坚持以MIDS哲学和按OO风范去开发应用；其次，MIDS/BUAA本身之开发也将坚持以MIDS哲学和按OO风范型进行。OO开发的一个要点是坚持“对象化”观点，即用对象观点去分析问题，和用对象模型去（映射）组织解。在MIDS的设计开发中，我们将框架、规则、事务、查询类、方法、应用等重要概念和实体都当作MIDS对象来看待。

2.2 MIDS的对象数据模型

多媒体信息处理要求系统必须提供处理复杂对象（特别是Text对象和Bitmap对象[4,5]）的能力；当代人工智能离不开知识和推理，而推理又多是基于规则的推理，框架表示是多种知识表示法中较通用和重要的一种。因此，尽管IDB依赖于前述的五项重要技术，MIDS设计中的第一个核心问题则是寻求能够将对象、框架和规则进行统一的数据模型——MIDS对象模型。

我们为MIDS/BUAA下的一个简化定义是：MIDS/BUAA是一个通用的OODBMS，其特殊性表现在它的通用对象模型是经典对象模型、规则和框架知识表示的统一模型。

定义1 一个MIDS**对象**是一个可以标识的实体的模型，有状态，主动或被动。主动意味着行为主体，被动意味着行为受体。一个MIDS对象必须具有：

1）一个系统定义的全局**唯一性标识符**UID；

2）一组属性，这组属性称之为该对象的状态特性，属性可使一个对象与其它对象发生引用关系，因而确定该对象所处的状态。与经典对象概念有所不同，每个MIDS对象的每个属性，除具有一个值侧面外，还具有另外三个可选的侧面：缺省值侧面、If-Needed侧面和If-Added侧面。在实现上，后两个侧面通常被定义成两个专门的方法，显然，若用C++的术语，具有后两侧面的属性必是非公共的。

3）一组方法（method），这组方法称之为该对象的**行为特性**。直观而言，每个方法是一个程序，可以完成一定责任（responsability），如访问和操作该对象，从而使该对象可能发生状态改变，服务和操作常作为方法的同义词。

4）一组**约束**（可选），对象所取的状态不是任意的，约束通常是一个谓词、规则或触发器，用于限定该对象的合法状态。对对象的约束常表现为对对象某些属性的约束。复杂约束在MIDS中被定义成一个单独的rule/trigger对象。

每个对象对应一个直接描述它们的类（**直接类**），且只有一个**直接类**，**间接类**（superclass）的个数可以不限。每个对象类的外延是一个对象集合，该集合中的每个对象均具有相同的状态特性和行为特性，对其它对象表现出相同的语义。

MIDS规定只有**持久对象**才具有UID，并且持久对象的UID是在该对象第一次入库时由MIDS的SMS模块赋予的，这个UID在

维普资讯 http://www.cqvip.com

该对象生存期中始终不变，这一规定与MIDS的存储策略[6]有关。

我们规定UID这一术语仅用于持久对象，但这不意味着挥发对象就没有标识。为了有所区别，我们将用Oid、Rid、Tid、Qid和Fid分别标识普通挥发对象、挥发规则对象、挥发事务对象、挥发查询对象和挥发框架对象。此处，"挥发"一词指不入库或尚未入库。每个持久对象都要经历一段"挥发期"，渡过挥发期的对象方可入库，入库后就获得UID（"绿卡"）。对象UID与它们在挥发期的标识一般是不同的（UID≠Oid）。普通对象在挥发期的标识同C++支持的标识机制是一致的。

规则作为方法的不适宜性已为文[7]所指出，规则作为对象的可行性已被实证[8]，规则作为对象的较好模型是ECA模型[8]，MIDS的规则对象模型亦属ECA模型之列。

MIDS对象模型与Form对象模型[9]相似，每个MIDS对象在不同的上下文中表现出不同的形式（和作用）：

- 在推理机中，它是一条规则或一个触发器；
- 在专家系统环境中，它是一个框架；
- 在面向对象系统中，它是一个普通对象；
- 在Hypertext系统中，它是一个正文段；
- 在Hypermedia的系统中，它是一个图表、一个图象，或一个声音。

MIDS对象中的每个属性在不同上下文中也表现出不同形式和作用：

- 它是框架中的槽；
- 普通对象中的属性；
- 正文段中的键字；
- 和图表、图象中的标号和标记。

概括（Generalization）和聚集（Aggregation）是OO风范非常强调的两种抽象机制。概括与IKO关系相对应，聚集与IPO关系相对应。下面给出几个与聚集有关的对象概念：

定义2 若对象A是对象B的一部分（即A与B之间具有IPO关系，记作IPO(A，B)），则将A称为B的**成分对象**，而将B称为A的宿主（enclosing）对象或**复合对象**[3,10]。

每个MIDS复合对象至少具有一个成分对象。复合对象与**大对象**是两个不同的概念，复合对象未必都是大对象，大对象也未必都是复合对象。MIDS把大小超过一个页面尺寸的对象称为大对象。

复杂对象是复合对象的另一个常用的同义词。

成分对象和宿主对象之间的IPO关系，在实现上反映为一种被引用关系。

成分/宿主对象间的被引用关系有下列两组基本性质：

1）依赖性质。a）**依赖性**：是指成分对象的存在依赖于宿主对象，若宿主对象挥发（被删除），成分对象亦随之而挥发；b）**独立性**：是指成分对象的存在与宿主对象是否存在无关；

2）共享性质。c）**共享性**：是指一个对象可以同时成为多个对象的成分对象，即可以同时与多个对象发生IPO关联；d）**排它性**：是指一个对象不可能同时与两个以上的对象具有IPO关联。

以上两组性质可以形成四种组合性质：(a，c)，(a，d)，(b，c)和(b，d)。在MIDS中，IPO关系的这种组合性质是以相关属性上的约束的形式被描述在类定义之中。

MIDS与O_2[4]系统一样，支持三种基本的对象构造子：Tuple，List和Set。可以认为这三个构造子对于绝大多数应用是完备的。

2.3 MIDS的持久性策略与库模式

库模式的规定与数据模型和持久性策略有关。MIDS的持久性策略属于严格的正交持久性[11]，由下列四条持久性规则保证：

持久性规则1（外延规则）：定义时被指明具有外延的类（带有"extension"说明[12]），其实例均为持久对象（即生成时自动入库）。

持久性规则2（命名规则）：任何被显式

维普资讯 http://www.cqvip.com

命名了的对象（具有持久性根）均是持久对象。

持久性规则 3（可达性规则）：被任一持久对象直接或间接引用到的对象或值（与持久对象之间具有 IPO 关联）亦自动持久。

显然值的持久依赖于与之发生 IPO 关联的宿主对象是否持久。

持久性规则 4（挥发规则）：如果一个持久对象的所有引用者（包括其持久性根的引用）均已挥发（reference count＝＝0），则该对象亦自动挥发。

数据库模式的作用在于定义欲存入库中之持久对象，并帮助 DBMS 管理和维护（提供元信息）这些持久对象，因之 MIDS 库模式应能直接反映 MIDS 数据模型的特征和持久性策略。

定义 3 在 MIDS 中库模式由下列三部分组成：

1）所有数据库类（每个持久对象的直接类和所有的间接（超）类都称为 DBClass）构成 MIDS 库模式的基本内容；

2）被每个 DBClass 在定义中直接或间接引用（通过 IPO 关联）到的对象类和值类型是库模式的必然内容；

3）持久性名字表 GNT（Global Name Table）。MIDS 把 GNT 中的每个入口均称为一个持久性根(persistent root)，存在于库中的任何持久对象或值都直接/间接地挂在某个持久性根上（Name≠UID），即通过 GNT，可以查找（追踪）出库中的任何持久性成分。

3. 值与对象[13]

值是一种抽象，是与时间无关的，不可改变的，和不可例示的；值本身不可被计数，谈这个 2，那个 2，或一个 3，两个 3，都是没有意义的，因而，谈及对值的修改、共享和例示亦均无意义[13]。

对象映射的是现实世界中的实体，因而存在于时间之中，有其自身的状态，可以被改变，可以被例示，因而也能被创建，销毁和共享[13]。

对象的唯一性取决于它的外部关系，而与其内部关系和性质无关，例如张三作为"Person"类的一个对象，其年龄，体重，甚至他的思想等，这些内在的关系和性质均可改变，对外则表现为状态之改变，但张三这个人永远是他这个人（名字亦可变），其作为人对象的唯一性是不变的。于是我们可以说对象是"外向型的"，而值是"内向型的"。对对象而言，封装性和标识（identity≠name）是两个本质特性[1]。

值恰与此相反，它完全为其内在的关系和性质所决定，例如一个集合型值完全被该集合的元素所决定。对值而言，抽象和自标识是它的本质特性[1]。

计算可以被看成是模拟。数据模型化要求程序员应该清楚他们将欲模型化什么，然后选用合适的结构。如果他们将欲模型化一个抽象，就应该使用值；如果他们将欲模型化一个实体，或是某个存在于时间域中的东西，则应使用对象。这表明程序设计语言应当同时支持值与对象，以及按适宜的方法使用它们的手段。事实上，类似的思想已经被反映在许多 OOS 或 OODBS 系统中，如 O$_2$[14]，Orion[14]，Exodus[15]，Advance[16]，和新版本的 FAD[17]之中。

纯面向对象（无值的概念）的数据库系统具有下列一些严重缺陷：

（1）每当用户仅仅需要一个复杂值时，他却不得不定义一个新的类，这便导致类网格的不必要增长，而类网格理应是仅含有这样一些类：它们的作用是描述那些可被不同的软件模块所共享的数据。类网格是不应该被那些仅用于描述非共享值的类所"污染"的[4]。

（2）若不支持值的概念，不仅数据操纵语言的性能要受到影响，而且在维护临时结果和删除重复时亦会有困难。这是由于对象的封装性，消息发送成了数据操纵的唯一途径[1]。

（3）元组、表和集合是三种最常规的对

维普资讯 http://www.cqvip.com

象结构，对于如此构造的复合对象，其成分对象只能以它们的 UID 出现和被引用在复合对象中，因而复合对象的结构事实上也是一个“平坦”结构，这在本质上无异于RDBS的“平坦”关系之概念，因而若不对值，特别是复杂值的使用提供必要的支持，“纯”OODBS 也要面临与 RDBS 一样的局限性。

因此，我们在进行 MIDS 对象模型的设计时，充分考虑了值概念的作用和重要性，并给之予以适当地位。对象是类的实例，值是型（type）之实例，MIDS 的数据模型对这两种概念均支持。

在 MIDS 中对象概念和值概念不是平等的，对象是“第一性”的，值是“第二性”的，这是指值的持久性依赖于对象的持久性，没有独立存在的持久的值，而只有与某个持久对象间具有“IPO”关联的那些值也因之变得持久化（入库）。这已经反映在 MIDS 持久性策略的可达性规则中。

4. 规则作为对象

MIDS 把框架模型和对象模型相统一，即框架作为扩充了对象模型，从中获得了知识表示和知识处理的能力；又从规则（和触发器）与对象的统一中获得了主动性和智能。主动性的背景是触发功能，智能背景的核心是规则推理。

4.1 规则对象的定义

定义 4 ECA（Event-Conditon-Action）模型的语义：当事件业已发生，对条件求值，若条件被满足（求值为真），则执行相应的动作。

规则对象（简称规则）作为对象，也同样应符合 MIDS 对象的一般定义，即含有四元素：UID、状态特性（一组属性）、行为特性（一组方法）和（可选的）约束。

4.1.1 状态特性 除了具有普通对象具有的那些通常意义下的状态特性外，规则对象特有的属性是：

1）Event：描述可触发该规则的一集事件（也称做事件模式）。Event 属性还可含有一组形参，当事件被探知后，事件形参便与一组实参束定在一起。事件的发生往往通过一个事件信号来通告（其中也含与之相关的实参之值，用于参数束定）。

2）Condition：这是一组查询（或谓词公式），一旦该规则被某个事件成功触发（即使之点火），则对该属性所描述的条件进行验证（求值）。

3）Action：该属性描述规则被点火成功（条件满足）后应执行的动作（一组操作或方法）。

4）E-C-Coupling：这一属性称之为“事件-条件”耦合方式，用来描述相对于（触发者）事件发生所在的事务，何时对该规则之条件进行求值。为了管理规则和长事务，MIDS 支持嵌套事务模型。

5）C-A-Coupling：这一属性称之为“条件-动作”耦合方式，用来描述相对于对该规则之条件进行求值的事务而言，何时调度（执行）该规则之动作。

E-C-Coupling 和 C-A-Coupling 属性有三个不同的取值：immediate（即时），separate（分离）和 defered（延迟）。其语义分别是：

（1）Immediate：对 E-C-Coupling 是指，一旦触发事件发生，则立即在同一事务中对条件求值（即条件求值要抢先于对原事务剩余步的处理而进行）；对于 C-A-Coupling 属性，其语义与之相似。

（2）Separate：对于 E-C-Coupling 是指，当触发事件发生后，便在另一个独立事务中对条件求值，由于事务处理必须接受 TMS 之调度，因而按 separate 耦合方式对条件求值的即时性要比 immediate 方式差；对 C-A-Coupling 情况相仿。

（3）Deferred：对于 E-C-Coupling 是指，当触发事件发生后，条件求值仍在同一事务中进行，但要等待原事务步都结束后方才开始，deferred 方式还可带有一时间参数 t，这时 deferred（t）是指相对于触发事件发生的那个时刻，t 秒钟之后（原事务步未必均处理完），开始条件求值，注意，t 参数也可以是能返回时

维普资讯 http://www.cqvip.com

间值的方法激活；对于 C-A-Coupling 属性，情形是相似的。

4.1.2 行为特性　在 MIDS 中，规则作为普通的数据库对象，可接受所有普通数据库对象所共有的各种操作（方法）：创建，修改和删除等。除此之外，规则对象尚具有一系列规则特有的操作，这些规则特有的操作影响规则处理，构成规则管理器的主要内容：

1) Fire（点火）：按 E-C-Coupling 耦合方式验证（求值）规则条件，若被满足，则再按 C-A-Coupling 方式，执行规则动作。

2) Disable（失能）：使得规则对一定的事件或事件集，失去自动点火之能力（即使事件业已发生，本该进行条件求值，继而可能执行其动作的规则，现不做任何反应）。

3) Enable（赋能）：这是 Disable 操作的逆操作，重新赋予规则自动点火的能力。

作为普通数据库对象的规则，不仅接受普通对象共有的操作，也要遵循 MIDS 的事务语义之要求。"点火"操作需申请"读"锁，而"创建"、"修改"、"删除"、"失能"和"赋能"操作均需要"写"锁。

需要指出的是，在 MIDS 中触发和点火的含意是不一样的。对于确实已经发生了的事件，若未被系统检测到，则认为该事件没有触发作用；若被系统检测到并且通告规则管理器 RM（见图 1），则认为该事件已成功地完成了触发作用，至于是否真会引起某些规则的点火，则取决于 RM 的"态度"。显然，触发的含义与前面定义的点火是很不相同的。

4.2　规则点火与事件

定义 5　源事件：在 MIDS 中确已发生的事件，它们通过 RM 触发某些规则对象，但却未必真能使之点火成功，这一方面取决于 RM 是否允许点火，另一方面取决于规则条件是否被满足。

定义 6　形式事件：形式事件是指规则对象中的 Event 属性所描述的一个事件模式，它可覆盖一集事件。

定义 7　正规点火：包含两个步骤，第一步（触发步）：由事件检测器检测到源事件的发生并将之通告给规则管理器 RM；第二步（点火步）：规则管理器利用检测器通告来的信号与所有相关规则的形式事件进行匹配，以决定哪些规则被触发（有待进一步点火），然后请求事务管理子系统 TMS 对匹配成功的规则，按它们的 E-C 和 C-A 耦合方式调度它们的条件求值和动作执行。

一个源事件可能触发某些规则，这些规则的成功点火继而又会触发另外一些规则，…，这种链锁反应式的触发，可形成一些触发链，一条触发链对应着推理机中的一个推理，这就是 MIDS 中支持推理机制的背景。

正规点火凭借事件检测器的检测和规则管理器的匹配建立触发链，触发者和被触发者之间的联系不是必然的，而是间接关系（与系统当时的状态有关）。如果触发者和被触发者的联系是必然的、确定的，则可利用 MIDS 提供的另一种更高效的点火机制，这就是下面的定义 8。

定义 8　直接点火：直接点火是指源事件绕过事件检测器的检测和规则管理器的匹配，直接触发某些规则并使之点火，即直接请求 TMS 调度这些规则的条件求值和动作执行。

直接点火适于为那些有着必然因果关系的对象建立因果链。在 MIDS 中使用直接点火的场合是：属性的 If-Added 和 If-Needed 侧面，和某些方法执行。

4.3　规则与事务

规则作为普通的数据库对象，其条件求值和动作执行中包含库操作是天经地义的，MIDS 系统通过对事务的管理和调度来实现对数据库操作的管理。一个源事件可能引发若干条规则相继点火，一次成功的规则点火意味着对条件求值和动作执行分别指派两个事务，并按指定的耦合方式加以调度。这些事务之间除了"兄弟"关系外，"父子"关系是另一种较普遍的关系。"兄弟"关系反映着事务间的并发关系，"父子"关系则要求系统

维普资讯 http://www.cqvip.com

能够支持嵌套的事务模型[8]。

4.3.1 MIDS的嵌套事务模型 在嵌套事务模型中[8]，有两种事务：顶层事务（父事务）和被嵌套的事务（子事务）。子事务完全包含在父事务中。被嵌套事务的父事务或者是一个顶层事务，或者是另一个被嵌套事务。MIDS事务模型假定一个父事务可以包含若干个子事务，兄弟事务间可以并发执行，当子事务执行时，其父事务被迫挂起等待。

MIDS的顶层事务与常规的数据库事务一样，也要求原子性，可串行化和永久化。子事务是原子的，并发执行的兄弟事务是可串行化的，子事务必须待到它和它的所有祖辈事务均正常提交后，才能永久性生效。一个事务的流产将导致它及其所有后裔事务均失效（瀑布式流产）。

很显然，MIDS的嵌套事务模型与复合对象模型完全一致，一个顶层事务可以被当成一个复合对象加以实现，这正是TMS的实现策略。这种策略的好处是，OMS中的持久性机制可被方便地用于实现TMS对长事务的支持。

4.3.2 耦合方式和事务 一条规则被点火，TMS在RM的请求下便为之创建一个用于对其条件进行求值的事务（称“条件求值事务”）。如果E-C耦合取immediate或deferred方式，则条件求值事务就作为触发（者）事务的子事务。子事务执行时，父事务被挂起。对于即时耦合方式，条件求值子事务是在触发事件发生的那一时刻被创建和开始执行的。对于延迟耦合方式，条件求值子事务是在父事务即将提交之前被创建并开始执行。若E-C耦合是separate方式，条件求值事务将被作为一个独立的顶层事务被创建，并与触发事务并发执行。

一旦规则条件被满足，TMS还要创建另一个事务并在其中执行规则的动作（称“动作执行事务”）。如同条件求值事务的特性取决于E-C耦合方式一样，动作执行事务的特性也取决于规则的C-A耦合方式，这里不再重复。

如果一个源事件的触发能使多个规则点火，TMS便为每条规则创建自己的一个条件求值事务。动作执行事务的情况与此相仿。

4.4 规则与体系结构

前面已经介绍了MIDS规则对象的ECA模型和它的嵌套事务模型。不难看出，ECA模型的实现要求MIDS必须在体系结构上提供下列支持：

（1）检测并报告源事件的发生——事件检测器。

（2）把事件信号与所有相关规则的事件模式进行匹配，确定被触发（待点火）的规则——规则管理器RM。

（3）对规则条件进行求值——TMS。

（4）执行规则动作——TMS。

在MIDS中，这些支持均由对象管理子系统OMS（请参见本期p35图3）提供。OMS中没有事件检测器。这是因为MIDS允许的事件源的范围较广，事件检测的任务被分布在其它有关的模块之中，由它们兼任。即把“检测”部分地变成“汇报”，这符合MIDS的总体设计，尤其对事件检测的效率有很大改善。

匹配的效率低下是许多主动数据库和推理系统的致命弱点。MIDS本身是一个数据库系统，可方便地利用数据库系统强有力的索引机制开发对规则较适宜的索引支持，以此获得事件匹配的高效率。

5. 结束语

讨论MIDS的对象数据模型是本文的中心议题。与同类系统相比，MIDS对值和规则（对象）的支持有独到之处，本文对值、规则和与规则密切相关的嵌套事务模型和对象管理子系统OMS作了较多介绍。

概括起来，MIDS对象数据模型的特点是：

• OO：支持OO风范的各种基本机制，如对象、类、继承，多态性和消息发送；具有较强的对象模型能力，特别是对多媒体信息

维普资讯 http://www.cqvip.com

分布式 数据库 系统设计

计算机科学 1994Vol. 21 №. 2

43-49

分布式数据库系统设计中的问题

TP311.13

郭 江

（北京航空航天大学软件工程研究所 北京 100083）

摘要 本文围绕着网络的规模、分布式的设计、查询处理、事务处理、支持分布式的操作系统、多数据库系统（MDBMS）等问题进行了探讨。如何理解和处理这些问题呢？本文从设计的角度出发，较为详细地讨论了这些问题对分布式数据库设计的影响以及设计时决策的常见选择方式。

关键词 多数据库系统 事务处理

现在，无论是商业数据处理，还是计算机工程环境（SEE）都迫切需要分布式数据库系统的支持。但是就目前的技术状况来说，关键的问题是怎样使分布式数据库成为产品。因此下面这些问题的实现决策就变得很重要了：(1) 网络的规模；(2) 分布式的设计；(3) 查询处理；(4) 事务处理；(5) 支持分布式数据库的操作系统；(6) 多数据库系统。这些问题的实现决策不仅影响着分布式数据库技术的发展，而且还影响着分布式数据库技术向产品的转化。

1. 网络的规模问题

目前，数据库界的人们并没完全理解所有实现分布式 DBMS 所做设计选择的含义。特别是一些协议和算法的规模会随着系统地理上的分布[2]和系统成份的增加[3]而增加。

一个值得注意的方面是：分布式事务处理机制在分布式数据库系统中是建立在广域网基础之上的。这就涉及到两阶段上锁和两阶段提交协议的问题。更重要的是与这些协议有关的一些问题，而且在一个较慢的广域网上实现起来可能比较困难。

规模问题仅是一个更为一般的问题的一个部分：没有很好地处理网络的体系结构和协议在分布式 DBMS 执行中所扮演的角色。在几乎所有的执行性能研究中，都是以一个假设的、非常简单的网络代价模型为基础。有时简单地使用一个固定延迟而根本不考虑网络特征，如负载、信息量、网络大小等。例如，考虑一个运行在 Ethernet 类型局域网上的分布式 DBMS，信息延迟会随着网络负载的增加而增加，而且并没有施加限制。因此，在 Ethernet 网上实现的分布式 DBMS 的执行性能模型就不能使用一个不变的网络延迟或延迟函数而不考虑网络的负载。一般说来，目前的情况是：还没有很好地理解不同局域网络体系结构中所提供的算法和协议的执行情况，就随意将它们从局域网移植到了广域网中。

处理规模问题的一个比较合适的方法是去开发出一个一般的、强有力的执行模型和测量工具以及相关的方法学。尽管已为集中式的 DBMS 作了很多这样的工作，但还没有扩充到分布式的 DBMS 上。

现在的分布式 DBMS 的执行性能研究通常是以简单的模型、人工的网络负载等假设为基础，或者是只考虑了仅有的几种特殊的算法。为了使研究具有一般化就需要作进一步的努力，但这并不是说不要分清主次矛盾。目前，一些系统的权衡已经处理得比较好了。例如，SDD-1 系统的查询处理就能在较慢的广域网上有效地执行分布式操作。最近的一些研究正在考虑查询处理器在较快的局域网上进行优化的问题。但是这样的一些权衡不能仅考虑定性的方面，还需要对执行模型进行定量的研究。

2. 分布式的设计

分布式数据库系统的设计方法因系统体系结构

的强有力支持；

• 基于代理者[18]的对象标识；

• 支持值：MIDS 对值概念有自己独特的理解和支持；

• 强类型的；

• Object，Frame 和 Rule 相统一的对象数据模型；

• 智能的：支持知识的表示和处理，以及基于规则的推理；

• 主动的：Rule 与 Trigger 统一，支持触发机制。

（参考文献共 18 篇略）