计算机科学2003Vol.30№.5

事务处理技术综述*)

陈小芳 丁 柯 金蓓弘
(中国科学院软件研究所 计算机科学重点实验室 北京100080)
(中国科学院软件研究所 软件工程技术中心 北京100080)

Transaction Processing Technology Survey

CHEN Xiao-Fang DING Ke JIN Bei-Hong
(Key Laboratory of Computer Science, Institute of Software, The Chinese Academy of Sciences, Beijing 100080, China)
(Software Engineering Technology Center, Institute of Software, The Chinese Academy of Sciences, Beijing 100080, China)
Email:fanger@otcaix.iscas.ac.cn

Abstract The concept of transaction originates from database research. It is a key technology to ensure the reliability and consistency of information. The flat transaction model is not suitable to current complex and distributed applications since it lacks the semantic support for complex applications. In this paper, we survey two approaches to solve the problems of the flat transaction model: extended transaction models and transactional workflows. We then present the transaction specifications and major commercial products of transaction processing technologies. Finally, we prospect the trend of these technologies.

Keywords Transaction, Extended transaction model, Transactional workflow

1 事务处理技术

事务处理技术是保证信息可靠性和一致性的重要技术。事务是具有ACID(atomicity,consistency,isolationanddurability)特性的原子操作序列,它的概念最早来源于数据库管理系统,用来保证应用程序对数据库访问的一致性和可靠性。在早期应用中,商用DBMS系统内部集成的事务管理器提供应用所需的事务处理功能。随着网络技术的发展以及应用需求的变化,以往集中式应用演化发展为网络分布应用,数据和处理分布在不同的计算机上。此时事务管理功能由专门的中间件(例如事务监控器)提供,事务处理技术也发展为分布式事务处理。

事务处理技术的核心问题是并发控制和恢复处理(CC&R)[1]。并发控制目的在于保证多个并发执行事务的最终执行效果等价于某种串行化执行的效果,即保证执行的串行等价性(serial equivalence)。目前有三种常用并发控制方法:锁机制、时间戳机制[2]和乐观并发控制[3,4],实用系统(例如DBMS)一般利用锁机制。另一方面,恢复处理用来解决事务执行过程中可能出现的各种故障。

在引入网络分布计算环境后,事务处理需要解决分布性带来的问题。首先,事务作用域不再集中于单一节点,而是分布在各网络节点上。事务性远程过程调用(Transactional RPC)可用来传递事务作用域。另外,分布事务处理牵涉更多的不同类型的共享资源,它们通常由资源管理器提供。为了在分布情况下保持事务ACID特性,分布事务管理器通常使用原子提交协议(例如两阶段提交协议)来提交整个分布事务。

近年来,事务处理技术和网络分布计算技术有了长足进步。客户/服务器模式、数据库系统、组件、群件系统等技术的发展对事务处理技术和网络分布计算技术影响很大。另外,日趋成熟的组件技术替代了基于RPC的分布计算平台。对应用而言,事务处理技术是保证可靠性的至关重要的基础设施(infrastructure),大量的分布应用必须依靠平台提供的事务服务才能正确和可靠地运行。

工作流应用、电子商务应用以及企业应用集成(EAI)是近年来兴起的分布应用,它们都需要事务服务的支持。工作流技术自80年代初期出现至今得到了迅速发展和应用。它一方面比较好地实现了企业(或组织)的业务流程自动化,另一方面,可以在大规模的异构、分布环境下支持相关任务的高效协同运行,非常符合现代快速多变的经济信息化和全球化的要求。

2 扩展事务模型

平面事务模型的应用控制只有一层,它比较适用于短事务,能有效地实现ACID特性。在处理长时间事务时,平面事务模型存在较为严重的性能问题,而且它无法对特定领域中的应用语义进行描述[5,6]。随着网络分布计算的普及,以及电子商务应用的发展,人们越来越多地需要使用长时间事务。为了解决平面事务模型的不足,研究者提出了各种扩展事务模型(ETM)[7]。它们在平面事务的基础上加入特定的应用语义,基于固定的控制和数据依赖,在不同方面放宽了对ACID特性的要求,以解决特定领域的问题。常见的扩展事务模型包括嵌套事务模型(Nested Transactions)、多层事务模型(Multi-Level Transactions)、Sagas、分支汇合事务模型(Split and Join Transactions)和柔性事务模型(Flexible Tran-

*)本文研究得到国家863基金项目(2001AA414020)、国家973基金项目(G1999035807)和国家自然科学基金重点项目(69833030)的资助。陈小芳 硕士生,主要研究领域为软件工程、分布事务处理技术。丁 柯 博士生,主要研究领域为分布式计算、分布事务处理技术。金蓓弘 博士,副研究员,主要研究领域为分布式计算。

sactions)等。

2.1 扩展事务模型举例

(1)嵌套事务模型[7,8] 是E. Moss在1985年提出的一种扩展事务模型,它允许在一个事务内部启动新的事务(子事务),父事务可以创建任意多个子事务,子事务也可以递归地包含自己的子事务,从而形成一棵事务树。子事务只有在父事务启动后才能启动,父事务在所有子事务提交后才能提交,父事务中止时要求所有子事务均回滚。

嵌套事务模型的优点是子事务之间可以并发执行,从而缩短了系统的响应时间,另外利用这种模型可以增强系统的模块化,用独立发展起来的模块实现了简单并且安全的结构重组。嵌套事务模型的缺点是很可能在一个事务中形成自身的死锁,并且子事务无法看到其兄弟事务的更新结果。

(2)多层事务模型[9,10] 是为了适应数据库系统向模块化和面向对象的发展而提出的,它支持松弛的ACID特性,在系统的并发以及性能上有较大提高。多层事务是嵌套事务的一种变形,也称为开放式嵌套(open nested)事务。它刻画了DBMS存储的层次结构,是System R更一般的版本。

在多层事务模型的事务树中,每个节点对应抽象层次化系统中特定层的操作,最高层对应传统事务,最底层对应操作系统的接口,它的每一层有自己独立的日志和并发控制机制。

多层事务模型的并发控制主要研究在层次上反映可交换性操作的冲突关系,以及这些操作的语义。

(3)分支汇合事务模型[11,12] 是由Columbia大学的G. Kaiser和C. Pu等人在研究非封闭(open-ended)活动时提出的。非封闭活动将产生长时间事务,分支汇合事务模型采用动态重构技术,对活动的各个并发事务进行动态修改。一种操作是把一个正在进行的事务分成两个可串行的事务,称为分支事务操作,它们从此可以独立地运行;另一种操作是分支事务的反操作,它把两个正在进行的可串行事务合并成一个事务,连同对资源的访问,称为汇合事务操作。

分支事务产生的新事务与嵌套事务不同,它不是原来事务的子事务,也不存在父事务,两个新事务之间是完全独立的;而汇合事务的结果则是一个普通事务。两种操作的结果对执行的串行化没有影响,只是部分影响了并发的原子性。

(4)Sagas[13] 是由Princeton大学的H. Garcia-Molina等人研究开发的一种扩展事务模型,它第一次提出了补偿事务的概念,事务T的恢复不再使用传统的阴影页或者日志方法,而是通过执行补偿事务C,在应用语义层次上消除事务T的更新效果。一个Saga是一个普通事务序列。对任何一个Saga,系统保证下面两种执行序列:

$T_1, T_2, \cdots, T_n$; 或者

$T_1, T_2, \cdots, T_j, C_j, \cdots, C_2, C_1$

Sagas不再提供严格的ACID特性,而是松弛原子性,放松了事务的隔离特性:Saga中的事务一旦提交即对其他事务可见。Sagas模型的缺点在于它要求每个事务均存在补偿事务,这不符合许多实际应用的情况,另外在系统的数据结构、并发控制以及恢复机制上人们还没有提出规范的可实现方法。

(5)柔性事务模型[14,15] 由Purdue大学研究出的一种松耦合的扩展事务模型,它对操作的原子性和分离性要求较低,主要应用在多数据库环境中。一个柔性事务包含多个子事务,用户可以指定子事务之间内部和外部的依赖性,这些子事务通过precedence和preference两种约束关系组合在一起。

柔性事务模型中子事务按其完成保证(termination guarantee)分成可补偿事务、可重复事务和pivot事务三种类型:可补偿事务是指可以补偿的事务,可重复事务是指执行有限次数后最终提交的事务,pivot事务则是除了这两种类型以外的其他事务。一个良构的(well-formed)柔性事务包含3个有序部分:首先是可补偿事务集合,然后是至多一个pivot事务,最后是一个可重复事务集合。由于柔性事务可以包含可补偿、不可补偿、可重复和不可重复的事务类型,所以它能较好地表示实际应用。

2.2 扩展事务模型的框架和系统

不同的应用领域要求不同的扩展事务模型,但是人们不可能为每一种扩展事务模型实现特定的事务管理器。可行的方法是实现一个可以支持各种ETM的事务管理器,如何处理好通用性和特定性之间的关系是这些系统成功的关键。

(1)ACTA 是一个形式化框架[16],用以描述事务之间的相互关系(事务之间的组成结构)和事务对数据对象的修改。ACTA本身不是一个特定的扩展事务模型,而是用来刻画其他事务模型的框架。ACTA通过一系列的公理、不变式(invariant)来描述特定ETM的可能执行历史,从而可以形式化地定义扩展事务模型的结构和行为。ACTA已被用来描述嵌套事务、Sagas和分支汇合事务。尽管ACTA对于验证事务模型有效,但它作为一个理论框架,不能用来构造一个可用的事务管理器。

(2)ASSET(A System for Supporting Extended Transactions)[17] 定义了若干个事务操作原语,这些原语提供给应用程序来实现事务控制语义。ASSET的事务原语实际上来源于ACTA框架,在某种意义上,可以认为ASSET是ACTA的实现。

ASSET的事务原语包括commit, abort, delegate以及permit,分别用来创建事务和控制事务间的脏数据访问等。文[17]表明ASSET可用来实现多种扩展事务模型。使用ASSET,应用程序的开发者必须自己实现事务模型,这种方法灵活,但同时带来了应用程序开发的负担。ASSET停留在设计想法阶段,没有形成真正的原型系统,因此其实用性不能验证。

(3)ECA方法 E. Anwar等人提出的ECA方法利用存储在主动数据库(active database)中的ECA(event-condition-action)规则来实现扩展事务模型[18]。在实现特定的事务模型时,系统开发者需要了解系统内部的数据结构,然后利用ECA规则访问相关数据结构。由于ECA方法将系统作为白盒对待,因此比ASSET更具灵活性。然而这种灵活性的代价是破坏了系统的封装性和可扩展性。另外ECA方法依赖主动数据库,其通用性和适应性不能验证。

(4)Reflective Transaction Framework(RTF) 其出发点是依靠和扩充现有的商用事务监控器来实现扩展事务模型[19]。该系统应用Open Implementation Approach方法,从而较ASSET和ECA有更好的模块化结构。另外该系统引入了事务适配器(Transaction Adaptors)概念,在商用事务监控器之上实现一个事务适配器来支持扩展事务的特性和语义。RTF将事务编程接口分成三个不同层次,其中底层接口由系统程序员使用,以实现和扩充事务处理语义,高层接口则直接用于应用程序。

3 事务工作流

工作流是一类能够完全或部分执行的经营过程(business

process),根据一系列过程规则,文档、信息或任务能够在不同的执行者之间传递、执行[20]。它首先在办公自动化行业中发展起来,与经营过程的重新构造(reengineer)密切相关[21]。一个经营过程是通过把各种不同的活动联系在一起,确定其数据的流向和相互控制关系而建立起来的,它通常具有持久性,牵涉到异类、分布环境中的许多用户和工具。相对于事务,工作流清晰地给出了某些应用领域的经营过程,具有丰富的应用语义,但它的理论基础比较薄弱,没有给出相关正确性方面的标准,不能保证并发工作流的一致性,在故障出现时不能保证可靠性;同时工作流系统无法控制应用中连续的调用,这与绝大多数基于事务系统的假设完全相反。

某些应用领域要求工作流作为一个整体要么正确提交,要么回滚结束,也就是说要求工作流具有事务的 ACID 属性。为此,不少学者提出事务工作流(transactional workflow)[22,23]概念,将事务处理有选择地引入工作流系统中。他们利用事务模型来保证工作流系统的正确性、数据一致性以及可靠性。事务工作流提供了工作流进程的功能,这些功能如任务协同、支持工作流结构等在通常的事务处理产品中是没有的,同时事务工作流考虑了在有故障和并发执行的环境下如何保证工作流的可靠执行。我们利用事务之间的耦合度以及人机交互的程度对各种扩展事务模型和事务工作流模型进行了分类,形象地表示为图1。

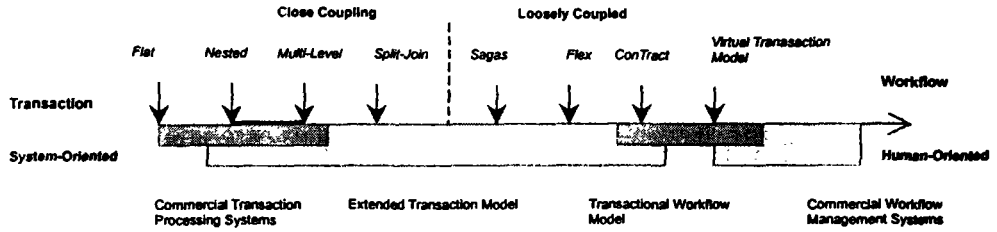

图1 扩展事务模型与事务工作流

3.1 事务工作流模型实例

(1)ConTract 模型[24,25] 是由德 Stuttgart 大学研究开发的一种事务工作流模型,用来定义和控制长时间、复杂的计算。受控制域(spheres of control)概念的影响,它采用了一些事务监控器的机制来管理控制流程,包括队列、上下文数据库等。ConTract 模型的基本思想就是用传统的具有 ACID 特性的短时间事务来构建大型的应用,提供独立于应用的系统服务,它的一个主要优点是把计算过程作为一个整体,具有可靠性和正确性。

ConTract 模型中两个主要的概念是 script 和 step。前者描述了整个事件的控制流及其他执行策略,它类似于一个程序,其中可以定义局部变量,某些部分可以并行执行,也可以用面向应用的同步方法共享某些对象;后者则实现了 script 中出现的基本计算,是整个模型的基本单元。ConTract 模型将 step 的编码与应用的控制流定义 script 分离,同时 script 可以组合若干个 step 为一个具有 ACID 特性的事务,以适应应用的需求。

(2)Virtual Transaction 模型 HP 公司的研究人员 V. Krishnamoorthy 和 M. Shan 提出了用于保证工作流正确性的 Virtual Transaction 模型[26]。Virtual Transaction 是一个比传统事务更高层次的 ACID 实体,它是整个业务流程的保证 ACID 特性的部分执行。Virtual Transaction 之间不能互相重叠,另外重置(reset)弧只能包含在一个 Virtual Transaction 中。Virtual Transaction 的原子性基于补偿机制,可以利用可视化工具来设定 Virtual Transaction 边界;每一个 Virtual Transaction 都可以设定独立的隔离性等级。

3.2 事务工作流的研究现状

与扩展事务模型相比,事务工作流的研究起步较晚,其理论基础比较薄弱。事务工作流建模的核心问题是如何结合应用语义和事务特性。现有文献给出了多种工作流建模的方法,包括基于 Petri 的模型[27]等。H. Schuldt 等人在柔性事务的基础上提出了事务性过程(transactional process)模型[28],并研究了该模型的并发控制和恢复处理问题。

在给出事务工作流模型后,可以对其性质进行研究。文[28]提出了一种验证事务工作流良构性的方法。事务工作流由事务组合形成,其执行满足松弛原子性。由于事务具有不同的可补偿特性和可重复特性,它们之间的组合失配会导致事务工作流的执行不能满足松弛原子性[29]。例如一个不可补偿的事务顺序连接一个不可重复的事务就产生了组合失配。不存在组合失配的事务工作流是良构的,良构的事务工作流能够保证所有执行均满足松弛原子性。

事务工作流的调度算法也是最近的研究热点问题[30]。由于事务工作流的活动粒度较大,并且不同的事务具有不同的完成特性,平面事务的调度算法不能用于事务工作流中[30,31]。H. Schuldt 提出了一种基于有序共享锁(ordered shared lock)的调度算法 Process Locking 用于事务性过程的调度[30],但是该算法为了在调度中可能出现的循环冲突问题,在全局范围内最多只允许一个事务工作流能够执行不可补偿事务,大大限制了工作流执行的并发度。文[29]定义了事务实例和事务类型之间两种不同粒度的冲突关系,并通过预测事务工作流的未来执行,使在未来执行中不会冲突的不可补偿事务能够并发执行,从而提高了并发度。

4 事务处理的规范和产品

4.1 事务处理规范

(1)CORBA OTS OTS(Object Transaction Service)[32]是 OMG 定义的用于事务处理的 CORBA 服务,它在 CORBA 模型的基础上定义了一系列跨越多个 CORBA 对象进行事务处理的接口,同时它在 X/Open DTP 上增加了一些功能,把函数形式的 XA 和 TX 接口替换成了 CORBA IDL 接口,从而 DTP 模型中的对象可以通过 IIOP 上的 CORBA 方法调用来通信。另外 OTS 能与 DTP 模型互操作,使用了事务性对象的应用可以用事务管理器的 TX 接口来进行事务定界。OTS 支持平面事务和嵌套事务两种事务模型,每一个 ORB(Object Request Broker)线程都维护一个事务上下文信息(Transaction Context),这个事务上下文或者为空表明该线程没有参与事务,或者引用当前活动事务。事务上下文可以在事务操作调用过程中被隐含地自动传递,同时也允许作为参数进行显

式传递。一个 OTS 应用由事务客户端、事务对象、可恢复对象、事务服务器和可恢复服务器五种实体组成。

(2)JTS(Java Transaction Service)[33] 是 J2EE 规范中的事务处理规范，它规定了分布式事务管理器的功能，如事务定界、全局事务、事务资源管理、事务同步、事务上下文传递、与其他事务管理器互操作等，并要求实现 CORBA OTS 的 Java 映射。JTS 要求分布式事务管理器对外遵循 JTA(Java Transaction API)[34]规范，而 JTA 分为三部分：为应用程序提供的定义事务边界的高层应用接口，为应用服务器提供的可以控制其中应用的事务边界的高层事务管理器接口，为资源管理器提供的一个工业标准 X/Open XA 的 Java 映射。这样，应用程序可利用 JTA 的高层应用接口，使用 JTS 提供的事务处理能力，应用服务器利用 JTA 的另一个高层接口可在应用服务器中提供事务功能，同时 JTS 利用 JTA 的 XA 接口可与各种资源管理器(如关系数据库，消息队列等)绑定，以实现对数据的各种访问。另一方面，由于 JTS 在低层实现了 CORBA OTS，故利用 CORBA 的 OTS 接口可实现事务管理器之间的互操作和可移植性。

4.2 典型的商用产品

(1)TUXEDO[35] 是早期最广泛应用的事务监控器产品，为应用提供了运行、发展和管理事务的平台，由 BEA 公司推出。该产品最初运行在 Unix 操作系统上，用于小范围的联机事务处理。经过多年的发展，TUXEDO 被不断地扩充和加强，现在它仍是世界上最广泛应用的分布式关键任务应用的事务处理平台之一，可以运行在多种操作系统上。另外 TUXEDO 还是许多 X/Open DTP 标准的基础，包括 DTP 的自身模型、XA、TX 和 XATMI。

TUXEDO 为用户提供了两种接口：基于库函数调用的接口 ATMI(Application to Transaction Manage Interface)和基于语言的接口方式 TxRPC。前者是一组库函数 API，后者将事务处理集成到远程过程调用中。这两组接口都提供了三个主要的事务管理原语：tpbegin()，tpcommit()和 tpabort()。除此之外，TUXEDO 还提供一些重要的服务，如事件服务、消息队列、安全管理以及负载平衡等等。

(2)Microsoft Transaction Server(MTS)[36] 是微软生产的基于组件的事务处理监控产品，与其他事务处理产品不同，它以对象为基础，目的是为了更加简单地生产、部署和管理基于组件的应用。除了提供标准的事务处理监控功能外，MTS 还提供了事务管理、事务远程调用的功能，包含 ORB 和消息中间件的属性。MTS 提供了基于组件的编程环境，支持 COM 组件，许多编程工具如 VB、VC、VJ 以及 Delphi、PowerBuilder 等均对 MTS 有效。

MTS 的应用包括客户端代码、MTS 的对象、资源分配器(dispenser)和资源管理器四个部分，另外 MTS 包含一个可以单独运行的事务管理器 MS DTC(Microsoft Distributed Transaction Coordinator)，它支持微软自身的 API，与 X/Open XA 的事务管理器和资源管理器具有互用性。

(3)WebLogic[37] 是 BEA 公司的另一个主要产品，它以大规模、关键性业务的企业应用为目标，提供了很强的支持企业级应用的能力。通过 J2EE 及 Web Service 的标准 API，结合一系列开发工具，它提供了较好的组件开发、部署和管理环境。WebLogic 在集群技术上具有突出的特点，一组工作在一起的 WebLogic 服务器通过集群技术能提供可靠和可伸缩的应用平台。

WebLogic 提供了负载平衡、失效恢复、缓存与连接池等一系列技术，在对高端应用的支持方面比较完备。另外 WebLogic 能够与 TUXEDO 有效地集成，提供优良的分布式事务处理能力。

结束语 事务处理是保证分布式环境中数据访问可靠性的一种重要技术，传统的事务模型即平面事务模型是最早提出的事务模型，其关键技术的研究已经相当成熟，它也是目前在商业中唯一广泛使用的事务模型，在银行、航空、保险等行业具有很强的应用背景。平面事务模型中的操作具有 ACID 特性，但它对控制流依赖和应用语义支持不足，无法在长时间应用领域进行应用。对平面事务模型进行扩充、加入适当的应用语义是一种很自然的想法，这就是扩展事务模型的来源。目前虽然各个大学和研究机构提出了多种扩展事务模型，但这些模型都没有进入实用领域，其中包含多种原因。一方面，一个扩展事务模型往往只解决某个特定领域的问题，不具有普遍适应性；另一方面，扩展事务模型仍然过多地强调并发控制和恢复机制，忽视了对应用语义的支持；一直缺乏关键应用(killer application)也导致了扩展事务模型不能被推广。

事务工作流结合了事务和通用工作流两者的特点，是当今研究的热门领域，它的研究课题超越了并发控制和恢复机制，更加强调业务流程。随着电子商务、企业应用集成等应用的蓬勃发展，事务工作流具有广泛的应用前景。

参考文献

1 Mohan C. Repeating History Beyond ARIES. In: Proc. of the 25th VLDB Conf. Edinburgh, Scotland, 1999. 1～17

2 Gray J, Reuter A. Transaction Processing: Concepts and Techniques. Morgan Kaufmann Publishers, San Mateo, California, 1993

3 Mohan C. Less Optimism About Optimistic Concurrency Control. In: Proc. of the Intl. Workshop on Research Issues on Data Engineering: Transaction and Query Processing, IEEE Computer Society Press, Feb. 1992. 199～204

4 Adya A, Gruber R, Liskov B, Maheshwari U. Efficient Optimistic Concurrency Control Using Loosely Synchronized Clocks. In: Proc. of the ACM SIGMOD Intl. Conf. on Management of Data, San Jose, CA, May 1995

5 Freytag J, Cristian F, Kaehler B. Masking System Crashes in Database Applications. In: Proc. Intl. Conf. on Very Large Data Bases, Sep. 1987

6 Dayal U, Garcia-Molina H, Hsu M, Kao B. Third Generation TP Monitors: A Database Challenge. In: Proc. ACM Intl. Conf. on Management of Data, May 1993

7 Mohan C. Tutorial: Advanced Transaction Models Survey and Critique. In: ACM SIGMOD Intl. Conf. on Management of Data, Minneapolis, May 1994

8 Moss J E B. Nested Transactions: An Approach to Reliable Distributed Computing. In MIT Press, 1985

9 Weikum G. Principles and Realization Strategies of Multi-Level Transaction Management. In ACM Transactions on Database Systems, March 1991

10 Weikum G, Deacon A, Schaad W, Schek H. Open Nested Transactions in Federated Database Systems. Data Engineering, 1993, 16(2)

11 Kaiser G E, Pu C. Dynamic Restructuring of Transactions. In Database Transaction Models for Advanced Applications, Aug. 1991

12 Pu C, Kaiser G E, Hutchinson N. Split-Transactions for Open-Ended Activities. In Francois Bancilhon, David J. Dewitt eds. In: Proc of the 14th Intl. Conf. on Very Large Databases, Los Angeles, California: Morgan Kafmann, 1987. 26～37

13 Garcia-Molina H, Salem K. SAGAS. In: Proc. of ACM SIGMOD

Conf. on Management of Data, 1987
14 Bukhres O, Elmagarmid A. Kuhn e. Implementation of the Flex Transaction Model. IEEE Data Engineering, 1993,16(2):28～32
15 Elmagarmid A K. Database Transaction Models for Advanced Applications. Morgan Kaufmann Publishers, 1991
16 Chrysanthis P, Rammaritham K. ACTA: A Framework for Specifying and Reasoning about Transaction Structure and Behavior. In:Proc. of ACM SIGMOD Intl. Conf. on Management of Data, New York, 1990
17 Biliris A, Dar S, Gehani N, et al. ASSET: A System for Supporting Extended Transactions. In:Proc. of SIGMOD Intl. Conf. on Management of Data, May 1994. 44～54
18 Anwar E, Chakravarthy S, Viveros M S. An Extensible Approach to Realizing Advanced Transaction Models. In Advanced Transaction Models and Architectures, Sushil Jajodia and Larry Kerschberg eds., Kluwer, 1997
19 Barga R, Pu C. A Practical and Modular Method to Implement extended Transaction Models. In: Proc. of Intl. Conf. on Very Large Data Bases, Zurich, Switzerland, 1995. 206～217
20 罗海滨，范玉顺，吴澄. 工作流技术综述. 软件学报,2000,11(7):899～907
21 Georgakopoulos D, Hornick M. An Overview of Workflow Management: From Process Modeling to Workflow Automation Infrastructure. Distributed and Parallel Database, 1995, 3: 119～153
22 Worah D, Sheth A. What do Advanced Transaction Models Have to Offer for Workflows?. In:Proc. of the Intl. Workshop on Advaned Transaction Models and Architectures (ATMA), Goa, India, Aug. 1996
23 Alonso G, Agrawal D, Abbadi A E, et al. Advanced Transaction Models in Workflow Contexts. In: Proc. of Intl. Conf. on Data Engineering, 1996. 574～581
24 Waechter H, Reuter A. The ConTract Model. In Database Transaction Models for Advanced Applications
25 Reuter A, Schwenkreis F. ConTracts—A Low-Level Mechanism for Building General-Purpose Workflow Management-Systems. In Data Engineering Bulletin, 1995,18(1)
26 Krishnamoorthy V, Shan M. Virtual Transaction Model to Support Workflow Applications. In:Proc. of the 2000 ACM Symposium on Applied Computing, Como, Italy, March 2000. 876～881
27 Aalst W M P. van der. The application of Petri nets to workflow management. Journal of Circuit Systems & Computers, 1998, (1):21～66
28 Schuldt H, Alonso G, Schek H. Concurrency Control and Recovery in Transactional Process Management. In: Proc. of the ACM Symposium on Principles of Database Systems (PODS'99), Philadelphia, Pennsylvania, USA, May/June, 1999. 316～326
29 Ding Ke, Jin Beihong, Wei Jun, Feng Yulin. New Model and Scheduling Protocol for Transactional Workflows. In: Proc. of the 26th Intl. Computer Software and Application Conference, COMPSAC 2002
30 李红臣，史美林，陈信祥. 事务工作流的并发控制算法. 软件学报，增刊 2001. 1～9
31 Schuldt H. Process Locking: A Protocol Based on Ordered Shared Locks for the Execution of Transactional Processes. In:Proc. of the ACM Symposium on Principles of Database Systems (PODS'01), Santa Barbara, California, USA, May, 2001. 289～300
32 Transaction Service Specification Version 1.1, May 2000
33 Cheung S. Java Transaction Service 1.0 Specification, Sun Microsystems Inc., Dec. 1999
34 Cheung S, Matena V. Java Transaction API 1.01 Specification, Sun Microsystems Inc., April, 1999
35 http://www.bea.com/products/tuxedo/index.shtml
36 http://www.microsoft.com/com/tech/MTS.asp
37 http://www.bea.com /products/servers_application.shtml

（上接第5页）

的发展，在 Internet 上运行 VOIP 也是可能的。VOIP 技术在 4G 系统中起了关键的作用。可以这么说，什么时候 VOIP 可以流畅地运行在 Internet 上，那么基于 Internet 的4G 系统就可以正式实施了。

(6)网络管理 网络管理主要包括两个方面：号码管理和计费管理。由于移动终端要在不同的移动接入网和不同的接入服务提供商之间自由漫游，那么移动终端的电话号码必须由统一的电话号码管理中心注册登记，使得电话号码和接入服务分离。计费管理也要由统一的结算中心来管理，以方便用户的使用。当然在发展初期也可以由移动接入服务商采用自己的计费方式独立核算。网络管理还要考虑网络的安全性。因为 Internet 的开放性一方面带来了各方面的好处，另一方面也给黑客的攻击提供了方便，每一个黑客都可以通过计算机和 Internet 直接攻击每一个移动终端。

结论 基于开放的 Internet 的4G 系统打破了以往移动通信系统的封闭式结构，开创了全新的未来。尽管目前的 Internet 还达不到4G 系统所要求的服务质量，但随着 Internet2 等技术的发展并逐渐融入到 Internet，未来的 Internet 必将满足4G 系统的要求。从分布网络结构这个角度来讲，4G 并不是像许多人预测的那样，是10年以后才会出现的事情，现在就可以开始了，现在就可以在 Internet 的基础上，运用蓝牙技术、802.11a/b、GSM、CDMA 等原有的技术进行4G 系统的试验。因为未来的 Internet 必定是在目前的 Internet 基础上逐步发展，并且兼容目前的 Internet，所以基于 Internet 的4G 系统也将随着 Internet 服务质量的提高而日益完善。本文需要强调的一点是，4G 系统和以往移动通信系统的本质区别不在于通信速率，而在于分布式的接入网络。

参考文献

1 Proakis J G 著. Digital Communication(影印版). 北京：电子工业出版社，1998
2 Ohmori S, Yamao Y, Nakajima N. The Future Generations of Mobile Communications Based on Broadband Access Technologies. IEEE Communications Magazine, 2000,38(12):134～142
3 Lemley B. Internet2: A Supercharged new network with true tele-presence puts the needs of science first. DISCOVER, 2002, 23 (5). http://www.discover.com/may_02/gthere.html? article = featinternet2.html
4 Internet II and UCNet. http://www.id.ucsb.edu/detche/library/www/internet2.html
5 Smith J E, Weingarten F W. Research Challenges For The Next Generation Internet. Computing Research Association, May 12-14, 1997. http://www.ngi.gov/pubs/
6 RFC3220: IP Mobility Support for IPv4. C. Perkins, Ed. January 2002
7 RFC2460: Internet Protocol, Version 6 (IPv6) Specification. S. Deering, R. Hinden. December 1998
8 VoIP Howto, v 1.5. June 2, 2002. http://www.tldp.org/HOWTO/VoIP-HOWTO.html