江西财经大学学术文库

嵌入式实时数据库系统的事务模型及其处理技术

夏家莉／著

嵌入式数据库系统的兴起可以归功于便携式计算设备的开发兴起。由于这些设备具有通讯能力，用户不仅用它们存储设备本身产生的大量数据，而且需要从中心企业下载资料以便脱线处理，因此，需要这些设备具备成熟的数据管理能力，所需功能常常如此复杂以至于平坦的文件系统不足以处理和操纵这些数据，这就促进了对嵌入式数据库的需求。

QIANRUSHI SHISHI SHUJUKU XITONG DE SHIWUMOXING JIQI CHULI JISHU

经济管理出版社
ECONOMY & MANAGEMENT PUBLISHING HOUSE

江西财经大学学术文库

嵌入式实时数据库系统的事务模型及其处理技术

夏家莉 著

经济管理出版社

图书在版编目（CIP）数据

嵌入式实时数据库系统的事务模型及其处理技术/夏家莉著. —北京：经济管理出版社，2005
ISBN 7-80207-263-8

Ⅰ.嵌… Ⅱ.夏… Ⅲ.数据库系统
Ⅳ.TP311.13

中国版本图书馆 CIP 数据核字(2005)第 023965 号

出版发行：经济管理出版社
北京市海淀区北蜂窝 8 号中雅大厦 11 层
电话：(010) 51915602　邮编：100038
印刷：北京晨旭印刷厂　　经销：新华书店

责任编辑：卢小生
技术编辑：杨　玲
责任校对：静　洁

787mm×1092mm/16　　10.75 印张　　184 千字
2004 年 12 月第 1 版　　2004 年 12 月第 1 次印刷
印数：1—6000 册　　定价：20.00 元
书号：ISBN 7-80207-263-8/F·252

序

公元676年，唐代诗人王勃去交趾探父，途径洪州（今江西南昌），在都督阎伯屿的盛宴上写下了千古名篇《滕王阁序》，其中有“物华天宝，龙光射斗牛之墟。人杰地灵，徐孺下陈蕃之榻”赞誉江西的佳句。千年的历史证明，王勃之言并非客套的溢美之词，而是据实之言。从盛唐至宋，江西经济文化十分发达，唐宋八大家江西占了三家。从长江经鄱阳湖赣江，翻越大庾岭经梅关入广东，商旅不绝，十分繁荣。近代的江西，志士仁人辈出，尤其是土地革命时期，江西籍的革命烈士（有姓名记录者）达几十万人。如今，中国改革开放，又逢盛世，江西经济、文化和社会发展又临新机遇。京九铁路开通，大大改善了江西的区位条件，当初“襟三江而带五湖，控蛮荆而引瓯越”的优势重现，“雄州雾列，俊彩星驰”胜景盛况再来，江西这块交织着历史文化沉淀和革命传统的红土地，将展现新的风采。

江西财经大学是一所以经济、管理类学科为主，理、工、法、文、农、艺术学科兼有的新型多科性经济类大学，始建于1923年。学校的目标是建设成为“在江西有优势，在华东有特色，在全国有影响”的第一流高等财经学府。作为学校发展规划的一部分，我们编辑出版《江西财经大学学术文库》（简称《学术文库》），以期展示和检阅我校教师的科研成果，推动我校科研工作和学科建设迈上新的台阶。《学术文库》将收录我校专家学者多年潜心研究获得的学术成果，国家社科基金项目的优秀研究报告，尤其是青年学者的力作。它们的内容涉及经济、管理等学科前沿的许多重大问题，其中不乏对社会主义市场经济建设实践中现实问题的研究成果。《江西财经大学学术文库》将成为我国经济、管理科学百花园中的一支奇葩。

在此，我特别要感谢经济管理出版社卢小生先生和各位编辑，由于他们对经济、管理科学学术研究的深刻理解和大力支持，我校的《学术文库》才得以顺利问世。

史忠良

2004年9月1日

目　录

1

绪 论

嵌入式数据库系统的兴起可以归功于便携式计算设备的开发兴起。由于这些设备具有通讯能力，用户不仅用它们存储设备本身产生的大量数据，而且需要从中心企业下载资料以便脱线处理，因此，需要这些设备具备成熟的数据管理能力，所需功能常常如此复杂以至于平坦的文件系统不足以处理和操纵这些数据，这就促进了对嵌入式数据库的需求。

嵌入式数据库系统还可以嵌入到大型软件系统或设备中，存储和处理来自于所在设备和其他地方的数据，可以访问监视器、进行诊断以及进行其他工作。例如，“汽车制造商用嵌入式数据库获取关于汽车磨损、毁坏、行程里数和性能等的数据”，“化学植物公司的处理控制系统可以用嵌入式数据库记录通过探测器收集到的日志，这些数据可以下载到大型中心数据库以便改善处理过程”。这样的设备还包括具备上互联网（Internet）功能的智能手机（个人数字助理）、售货机等；智能装置和嵌入式系统也开始用嵌入式数据库，包括一些路由器和 hub；嵌入式数据库系统不仅仅存在于时髦的移动设备或固定的灵巧设备中，也可以将嵌入式数据库系统用于成熟的应用环境，这些环境的资源对于今天的完整数据库管理包和需要很多资源的操作系统来说显得贫乏。总之，嵌入式数据库系统可应用于航天、航海、军事武器、工业控制、机器人等各行业以及汽车、家用电器等生活设施中，具有广泛的应用前景。

嵌入式数据库系统的设计目的，是在最小的干涉和最小的系统影响下进行数据存储和恢复。由于嵌入式数据库系统常常需要对环境做出实时反应，此概念建立在实时和近似实时的嵌入式计算中，应准确地称之为嵌入式实时数据库系统。

嵌入式实时数据库系统的研究尚处于初级阶段，目前未见成熟的产品，市面上有一些嵌入式数据库系统和实时数据库系统的产品，但是，嵌入式数据库

系统基本上是传统商业数据库系统的剪裁，实时数据库系统在模型和并发控制机制上也并无新意，它们沿用了普通数据库系统的一些控制机制，或加以部分改造。

现在的问题是，一方面，目前的嵌入式数据库系统不具备实时性，但在实际应用中，常常要求系统对外部环境做出实时反应；另一方面，目前的实时数据库系统又由于难以保证高的成功率而不能适应嵌入式环境，因为嵌入式实时数据库系统的基本目标之一就是具有高的成功率从而具备高的可靠性，除此之外，嵌入式实时数据库系统还应该具备一定的主动性，能捕获特殊事件并做出相应反应，因此，严格地说，目前还没有真正意义上的嵌入式实时数据库系统。

嵌入式实时数据库系统同时具备实时性、主动性和嵌入式的特点，并采用了内存数据库的技术，它集成了实时数据库系统、内存数据库系统和主动数据库系统的处理技术。

1.1 嵌入式数据库的现状

嵌入式数据库市场一般不是直接面向用户而是面向设备制造商，设备制造商将嵌入式数据库系统嵌入到设备和应用中，在应用初期，嵌入式数据库主要用于提供对大型、数据密集性应用的数据管理，比如，Intuit's Quicken 金融管理软件。如今，设备中的嵌入式数据库可以与中心数据库通讯，具有与其他设备同步的功能（移动数据库）。

将嵌入式数据库产品分类成表 1.1 所示的关系型 C/S 产品、表 1.2 所示的面向对象 C/S 产品，以及表 1.3 所示的嵌入式程序库（它不需要单独的服务器处理）。这些产品各有优劣，因为不同的原因而流行，表 1.3 中只有 gdbm 不支持事务和一定程度的并发存取。

如表 1.1 所示的厂商提供 C/S 关系型系统。微软公司、Oracle 公司、Sybase 公司以及 Informix 公司研制并出售高级终端企业数据库系统和企业数据库引擎。

对于嵌入式系统开发者而言，C/S 关系型产品十分流行，因为程序员们已经熟悉了 SQL 和关系数据库设计，缺点是客户机和服务器之间的通讯要花费额外的代价，并且为在嵌入式系统中安装、运行和维护一个独立的服务处理器增加了复杂度。

表 1.1 关系型 C/S 产品

厂商	产品	特点
Centura Software	Velocis	运行于桌面、服务器和嵌入式 OS。支持数据管理的 SQL 和编程接口。
Empress	Empress RDBMS	运行于主要的嵌入式 OS，支持数据管理的 SQL 和编程接口。
Microsoft	MSDE	与 Microsoft SQL Sever DB engin 兼容，支持的并发度有限，处理的数据量较小。只能运行于微软操作系统。
Oracle	Oracle 8i	运行于 Linux。
Perasive	Perasive. SQL 2000	适应于嵌入式 OS，也能运行于桌面和服务器 OS 上。支持简单的数据操作接口，可以根据嵌入式需求裁减。
Polyhedra	Polyhedra	支持 SQL 的 C/S 数据库引擎，运行于桌面、服务器和嵌入式 OS。
Solid	Solid Embedded	在多种服务器和嵌入式 OS 上提供数据库管理，通过 SQL 提供数据存取，具有 ODBC 和 JDBC 接口。
Sybase	SQL Anywhere	一个可以根据应用裁减的 SQL 引擎。特别强调与运行于服务器上的 Sybase 企业数据库同步。
TimesTen	TimesTen	通过 SQL、ODBC、JDBC 提供标准的关系数据存取，设计使用于大内存系统，主要运行于桌面和服务器系统。

表 1.2 面向对象的 C/S 产品

厂商	产品	特点
Objectivety	Objectivety/DB	对 C++、JAVA 稳定的对象存储。支持桌面和服务器 OS，但不提供嵌入式 OS 支持处理模型比其他厂商的产品复杂。
Persistence	PowerTier	与许多厂商生产的流行数据库接口的对象存储。支持企业 JAVA Bean 和 C++。客户软件运行于桌面和服务器 OS 上，服务器平台由数据库服务引擎的厂商决定。
POET Software	POET Object	对象库允许 C++ 和 JAVA 应用于各种后台数据库服务器，包括 POET，面向对象数据和 Oracle 以及其他厂商的关系型数据库，提供稳定的对象存储，不支持嵌入式 OS。

表 1.2 所示的产品似乎很适合嵌入式市场，但很少有人考虑，因为开发者为 Unix 设计了流行的面向对象数据库并且使它们建立在关于内存管理系统和通讯处理的有关假设上，难以与嵌入式 OS 接口。然而，可以证明它们可用于基于 Linux 的系统，比如，FreeBSD 可以用于嵌入式设备。面向对象数据库系

统之所以流行，是因为它们集成了C++和JAVA程序语言，并且对应用程序员隐藏了数据库设计的复杂性，主要缺点是C/S通讯负担和很少的开发者支持嵌入式OS。

表1.3 嵌入式程序库

厂商	产品	特点
Centura Software	RDM. db. linux	作为源代码发布，rdm编译到与应用相连接的一个库中，运行于嵌入式OS上，公司提供RDM的linux接口db. linux。
FairCom	c－tree Plus	作为源代码发布，直接编译到应用的地址空间。支持QNX和LynxOS，但用户可以将源代码直接接口（port）到其他嵌入式OS。
Free Software	Foundation Gdbm	提供简单的编程接口到数据记录，不支持并行的读写和事务，遵从Library GeneralPublic License，以源代码的形式发布，没有Free Software Foundation对嵌入式OS接口的明确支持。不管怎样，很多接口可用，用户可以自由地将该软件接口到新的OS。
Informix	C－IS AM	可嵌入的数据管理库，提供嵌入式系统要求的服务，该库只运行于桌面和服务器OS。
Sleepycat Software	Berkeley DB	为数据库提供简单的编程接口，可以根据嵌入式系统的要求配置需要的和不需要的部件。以源代码的形式发布。运行于桌面和服务器以及某些嵌入式OS上。

表1.3所示的为嵌入式程序库，它的系统直接链接到应用的地址空间，提供简单的语言级API而不是操纵数据的SQL。嵌入式程序库的主要优点是：

①快速执行，因为数据库操作不需要与隔离的服务器通讯。

②提高了可靠性，因为极少的部件运行在嵌入式系统上。

嵌入式程序库的明显缺点是：要求开发者精通非标准化的编程接口。

1.2 实时数据库系统的发展

实时数据库乃指事务和数据都可具备显式定时限制的数据库管理系统，系统的正确执行既要满足逻辑约束又要满足时间约束。同时满足这两种一致性要

求的实时数据库事务的处理与传统的数据库相比具有更大的困难，事务处理中的众多不可预测因数的存在，以及时间约束与逻辑约束的并存导致实时事务的模型、调度策略和并发控制技术成为研究热点。

1.2.1 模型

关于实时数据库事务模型的演进成果较多，其中有代表性的有如下一些：

Kim 建立了一个实时数据库事务模型，它包含硬实时事务和软实时事务，维护数据的时态和逻辑一致性，支持多担保级别。在这个模型下，它设计出一个集成的事务处理方案，提供实时数据库系统的预测能力和一致性，这样，系统中的每个应用被确认取得它们各自的性能目标（担保级），并且维持一致性需求。模拟实验表明：高确认级需要更多的系统资源，代价高于非确认事务。

Braoudakis 采取不同的方法，将事务与一个价值函数关联。该价值函数表示时间约束以及所有的重要的系统使命，这个模型可以说明非实时事务、软实时事务、固实时事务和硬实时事务。此模型的特点是用一致的方法处理不同类型的事务，事务价值以及价值函数的观点被应用于两个实时系统以及实时数据库系统中，任务的价值在接纳该任务时被估算出来，根据任务的价值决定是否拒绝或移走一个已经被确认的任务；只要没有更高价值（严格的）的冲突任务到达，被系统接纳的任务有条件地确保完成其执行。

Zhou，Rundensteiner 和 Shin 将面向对象的观点结合到实时数据库系统中，提出了 ROMPP——一个实时的对象模型，其中用面向对象的框架探讨了时态和逻辑的一致性和正确性。

1.2.2 实时并发控制策略

在实际应用环境中，许多事务都带有时间特性，需要对特定事件做出定时反应，如：定时采样、在特殊的时间点或时间段检测水库的水位及闸门的位置等，这些工作是传统数据库系统难以胜任的，需要实时数据库系统（RTDBS）的支持。随着计算机速度越来越快，功能越来越强，使用范围越来越广，实时数据库系统越来越重要。例如，它们用于股票交易程序、电话转换系统、虚拟环境系统、网络管理、自动工厂管理、指挥和控制系统中。实时数据库系统是事务或数据带有显示的定时限制的数据库系统，其中带有显示时间限制的事务是实时事务，其典型的形式就是事务截止期。对于具有硬截止期的实时事务，必须在其截止期之前完成，否则其价值为零，甚至导致灾难。对于具有软截止

期的实时事务，如果它在其截止期之前不能完成，则价值下降，但系统通常会允许它继续执行。

由于具有时间约束，我们不能沿用传统数据库系统的并发控制协议，因为它们强调所有事务具有平等的地位和相同的调度机会。对于实时数据库系统而言，为了使尽可能多的事务满足它们的截止期，应该将事务的时间信息加入到调度中来，事务越紧急应越早地投入运行。当一个事务正在运行时，如果新到了一个更为紧急的事务，该已执行事务应该让出处理机及对所有资源的控制权，使新到的紧急事务先执行。

具体来说，实时数据库系统中事务的并发执行具有以下特点：

①并发执行的事务具有时间约束和依赖性。实时数据库系统中的事务具有显示的时间约束，当它们并发执行时，事务之间可能存在与时间有关的依赖关系，比如，事务 T_1 必须在事务 T_2 开始执行之前（或提交前/后）开始/提交；一个事务在运行过程中可能触发其他的子事务，被触发事务与此触发事务并发执行，当触发事务完成时，可能不能立即提交，需要等到被它触发的事务也完成以便一起提交；等等。在传统数据库系统中，并发执行的事务间也存在关联，但主要是由于事务间通讯产生的，与我们所指的关联明显具有区别。

②必须满足硬实时事务的截止期要求，尽可能满足软实时事务的截止期要求。在实时数据库系统中，按截止期的特性将实时事务分做硬实时事务和软实时事务，系统必须保证硬实时事务的截止期，否则，该事务在截止期后可能对系统产生危害；同时，虽然软实时事务超过截止期后不会对系统产生危害，但该事务的价值也会急剧下降。由此，并发控制在保证数据一致性的前提下，为了满足事务的截止期要求，应区别对待不同特性的实时事务。

③一个正在运行的事务可以被更为紧急的事务抢占系统资源和 CPU 控制权。如上所述，当一个紧急事务到达时，为了处理该紧急事务，系统使正在执行（还未完成）的事务夭折，执行该紧急事务。

④由于必须保证事务的原子性，被抢占的事务夭折重启。实时数据库系统的事务与传统数据库系统中的事务相比，具有更为复杂的结构和更强的描述能力，它已不再是平淡的操作序列，典型的事务模型具有分裂结构、嵌套结构等，似乎不必维持事务的原子性。其实不然，从事务的整体角度上来说，由于事务可以嵌套（或分裂）甚至触发多个子事务，其原子性被破坏了，但对于组成该复杂事务的基本事务（子事务）来说，依然应保持原子性，因此，当一个基本事务被抢占后，必须重启。

⑤系统宁愿要及时的部分正确的结果，也不要过时的精确的结果。在资源受限的实时数据库系统中，并发事务竞争系统资源，当系统超载时，必然存在某些事务不能在其截止期内完成，对于某些实时应用环境，在截止期之后的精确结果是无效的。那么，在截止期内，即使得不到精确结果，如果能得到近似结果也比根本没有结果要好得多。比如，在交通管理系统中，为了合理调度汽车，保持各条马路负载的均衡，必须实时探测各条马路上的当前流量，如果不能在限定时间内得到查询结果，常常取一个模糊值作为调度的依据。

⑥事务处理的正确性不仅依赖于逻辑结果，而且依赖于逻辑结果产生的时间。在更多的情况下，用户需要得到精确的实时的计算结果，事务超过截止期后得到的结果可能和截止期前的系统状态相去甚远，其正确性得不到保证。比如，要求事务在3秒内计算机场上的飞机架次，如果该事务在4秒后完成，有可能在最后1秒内机场上增加了1架飞机。

⑦数据一致性标准有所放松。传统的数据库一致性标准是可串行化，并发控制的目的是保证数据的逻辑一致性，要求事务是平坦的，具有ACID特性。由于实时数据库系统必须满足数据的时态一致性和逻辑一致性，则此标准过于严格，因为在并发控制算法中，由于事务的重启和/或受阻可能导致无法预计的延迟。为了使尽可能多的事务满足截止期，需要放松正确性标准以满足或兼顾时间一致性要求，早期的努力集中于放松截止期的语义（主要是软实时事务和固实时事务，不能应用于硬实时事务）或事务的ACID特性（特别在可串行性方面）。比如，Kuo和Mok提出了建立在基于语义的正确性标准之上的并发控制策略。

一般做法是，在传统的乐观或悲观并发控制基础上，加以改造以适应实时并发控制的需要。典型的并发控制正确性标准有：

·ε—可串行化并发控制。ε—可串行化（Epsilon－Serializability）是事务处理中的一种正确性标准，它是一种更一般的误差受限的串行化。它允许不一致性数据被读取，并控制不一致性的程度，使误差量限制在一个指定范围，而使数据最终能达到一个一致性状态。ε—可串行化需要数据库的数据空间是一可度量空间，使用ε--可串行化放松逻辑一致性要求以期满足时间一致性要求。

·Δ—可串行性并发控制。Δ—可串行性是基于“相似性”概念的正规可串行性的扩展。相似性概念是指在时间和精确性上有略微差别的数据可以彼此替换地作为事务的可读数据。“相似”是内在地依赖于应用的概念，不同的事务对同样的数据对象可以有不同的相似性要求，两个值有多大的“微细差别”才

能算相似，应依特定的应用而定。“相似”是一种二元关系，它是自反的、对称的，但不一定是可传递的。

·基于语义的并发控制。可串行化不是维护逻辑一致性的惟一方法，还可以依据事务在数据库中的操作语义将事务分类，使用兼容集而非可串行化作为正确性标准。在 Rhode Island 大学开发的 RTSORAC 系统中使用了基于语义的并发控制技术。

⑧评价并发控制协议性能的标准不是反应时间和系统吞吐率，而是满足事务截止期的比率，即系统的成功率。由于事务需要满足时间约束，其正确性标准也有别于传统的数据库系统，系统不能一味地追求吞吐率，必须将重点转移到满足事务截止期上，只有成功的事务才是正确的事务。

总之，传统的并发控制协议不能满足调度实时事务的要求，需要开发新的适合于实时数据库系统的并发控制协议。

(1) 基于锁的并发控制协议

基于锁的并发控制协议主要有以下几种：

①基于优先级的两段锁协议 2PL－HP。2PL 是经典的传统数据库系统的并发控制协议，事务在读写数据对象之前，必须对此对象上锁，事务保持这些锁直到该事务完成（提交）才释放，如果锁请求被拒绝，则等待，直到持有锁的事务释放这些锁。对于实时数据库系统，两段锁需要使用基于优先级的冲突解决机制来保证高优先级事务不被低优先级事务阻塞。在高优先级机制中，所有解决数据冲突的方法倾向于高优先级事务。当一个事务以冲突方式申请被另一事务持有的锁时，优先级较低的持有者重启，优先级较高的申请者被授予锁；如果申请者的优先级低，它等待优先级较高的锁持有者释放锁。另外，只有当新的读锁申请者的优先级高于所有等待写锁操作时，它才可以加入一组读锁持有者，该协议被称为 2PL－HP。

高优先级机制和传统的循环等待机制相似，它们都加入了两段锁来防止死锁。惟一的差别是高优先级机制使用由事务时间限制决定的优先级顺序来进行冲突处理，而循环等待机制使用由事务到达时间决定的时标顺序。很显然，如果优先级分派机制赋予事务惟一的优先级值，并且不动态改变并发事务的相对顺序，高优先级可作为一种死锁预防机制。

②2PL－WP。2PL－WP（等待升级）使用优先级继承机制来解决冲突。优先级继承是为了解决优先级倒置问题。使用这一策略，当优先级倒置发生时，持有锁的低优先级事务将以在等待锁的事务中的最高优先级执行，直到它

释放锁。由于优先级的升级，它能够比没有优先级升级时执行得快，这样释放锁就很快。结果是，高优先级事务的阻塞时间可能降低。

然而，使用 2PL－WP 时事务阻塞的时间仍然不确定，因为数据冲突很高时，优先级继承实际上导致系统中大部分或所有的事务以同一优先级执行。在这种情形下，实时数据库系统的行为显著地降低到传统的数据库。实时并发控制算法的实验性能评价确认在高的数据冲突的情形下，2PL－WP 的性能降得很快。Huag 研究指明，综合高优先级和等待提升机制解决冲突具有较优的性能。它通常使用高优先级策略操作，只有锁持有事务接近结束时强迫锁申请事务等待，并且由于等待而出现优先级倒置时使用优先级继承。

③2PL－PC。优先级顶是解决优先级倒置的另一种方法。集成实时数据库系统中的优先级顶和封锁机制产生了 2PL－PC。这种机制对于硬实时环境很有前途，因为它阻止死锁形成并且严格限制了优先级倒置产生的事务阻塞时间。然而，这种机制需要被每一事务存取的数据对象的优先级知识。这种情形在许多数据库应用中不适用，因为在很多情形下事务执行是数据依赖的，从并发度的角度来说，这一机制很保守。

④RPL。实时锁协议 RPL 使用上锁和动态调整串行化顺序来解决优先级冲突。串行化顺序调整的基本思路是推迟确定事务之间最后的串行化顺序，这有利于高优先级事务动态调整临时性的串行化顺序，这种机制能防止因事务的优先级顺序和串行化顺序不匹配而导致的阻塞和夭折。在 RTL 中，为了实现串行化顺序的动态调整，事务的执行和 OCC 中一样是分阶段的，第一阶段为读段，事务从数据库读数据并且写到局部工作区，然而，与 OCC（冲突仅在确认阶段解决）不同的是：RTL 在读阶段使用事务优先级解决冲突。在 RTL 的写阶段，决定了最后的串行化顺序，并且将更新永久地写到数据库。事务读阶段所依赖的控制和局部工作区的使用也提供了潜在的高并发度。

⑤基于资源预报的并发控制协议。öZGÜR Ulusoy 提出的并发控制协议强调资源预报，每个被系统接纳的事务向系统预报自己所需的资源，系统按事务的优先级提供服务。对于一个新到的事务，如果系统不能满足其资源要求，该事务必须等待，直到优先级高的事务释放了所需资源系统才将该事务投入运行，此方法没有死锁的可能。

（2）乐观并发控制协议

在乐观并发控制协议中，事务允许不受阻碍的执行直到它们到达提交点再对它们进行确认，这样，事务的执行包括读、确认、写三个阶段。关键是确认

阶段，确认基本上分为向后确认和向前确认两种方法。在向后确认机制中，确认处理针对已提交事务；在向前确认中，事务的确认针对当前运行的事务。

①OCC－FV。如上所述，在实时数据库系统中，数据冲突的解决应该有利于高优先级事务，然而，在向后确认中，没有办法在串行化处理中考虑事务的优先级，因为它处理已经提交的事务，这样向后确认机制不能应用到实时数据库系统。向前确认解决冲突较灵活，正被确认的事务或者冲突的动态事务可能重启，所以它有利于实时数据库系统，而且，向前机制通常比向后机制探测和解决冲突早，资源和时间浪费较少。我们把使用向前确认的乐观算法称为OCC－FV。

需要指出的一点是，与2PL－HP不同，OCC－FV不使用事务优先级信息解决数据冲突，这样，在OCC－FV中，高优先级事务由于低优先级事务提交可能需要重启。已经提出和研究了几种用优先级等待或夭折机制将优先级信息加到OCC－FV中的方法。

②OPT－SACRIFICE。OPT－SACRIFICE是一个乐观协议，它使用优先级驱动夭折来解决冲突。在这个协议中，一个事务到达确认阶段时，如果一个或多个冲突事务比确认的事务优先级高，它就夭折；否则，它就提交并且所有冲突的事务立即重启。该协议以这样的方式使用事务的优先级：确认事务为了与它冲突的高优先级事务而牺牲自己。在这种机制中，采用向后确认的乐观协议，在确认事务大部分执行后，可以结束夭折（排除夭折），因为冲突的高优先级事务仍然在读段。这种机制的另外一个问题是无效牺牲。所谓无效牺牲是指一个事务为之牺牲的是后来被删除的事务。

③OPT－WAIT。在OPT－WAIT机制中，当一个事务到达确认段时，如果它的优先级在所有的冲突事务中不是最高的，它就等待具有较高优先级的事务完成，这一协议使较高优先级的事务首先满足截止期。一个确认事务等待时，可能由于具有较高优先级事务之一的提交而重启，因为优先权给了高优先级事务，这一机制保持了实时冲突处理最初的目标。值得注意的是，该机制中没有无效重启，因为所有的重启是根据要求而做的。

如果一个事务等待一段时间后最终提交了，它导致具有较低优先级的事务在后来某时间重启，从而降低了这些事务满足截止期的机会，这就浪费了资源；随着导致瀑布夭折的阻塞链的形成，这一问题更严重。另外，一个确认事务等待时，新的冲突可能发生，从而使冲突事务的数目和夭折的机会增加。

④WAIT－50。在WAIT－50机制中，只要与之冲突的事务有一半优先级