一种基于约束的事务工作流并发控制方法①

刘 慧[1,2]，王 宁[1]，刘元元[1]，韩晓琼[3]

[1](中国科学院沈阳计算技术研究所，沈阳 110168)
[2](中国科学院研究生院，北京 100049)
[3](北京大学 信息科学技术学院，北京 100871)

摘 要：在辽河流域水环境管理中，许多业务流程操作都满足事务的特性。传统工作流系统中并发控制的实现主要是依据对共享数据项的存取控制，当用来处理事务工作流时会影响系统的效率，因为有的业务流程可能持续的时间比较长，这样它对共享数据项的占有时间就会很长而导致其他流程不能继续流转。为解决这一问题，借助类似于处理器中指令流水的思想研究了一种事务工作流的并发控制方法，并在此基础上建立了总体设计模型，借助并发控制器和任务管理器来高效调度工作流。在保证工作流执行正确的前提下，引入约束的概念来实现事务工作流的并发控制，从而提高系统的性能和效率。

关键词：事务；事务工作流；并发控制；约束

Concurrency Control Method Based on Constraint in Transactional Workflow

LIU Hui[1,2], WANG Ning[1], LIU Yuan-Yuan[1], HAN Xiao-Qiong[3]

[1](Shenyang Institue of Computing Technology, Chinese Academy of Sciences, Shenyang 110168, China)
[2](Graduate University, Chinese Academy of Sciences, Beijing 100049, China)
[3](School of Electronics Engineering and Computer Science, Peking University, Beijing 100871, China)

Abstract: Many business process operations are transactional workflow in LiaoHe River water environmental management system. In the traditional workflow system, concurrency control mainly based on the realization of accessing control on the shared data, using it process the transactional workflow would affect the efficiency of the system when some business process operations sustained for a long time. In this case, these processes hold the shared data for a long time, so other processes must wait. To solve the problem, this paper uses a similar thought of instructions pipelining in the processor to research a method of concurrency control on transactional workflow, and based on it the general design model is given. The workflows are highly scheduled wth the help of concurrency controler and the task manager. In the premise of the correct execution of workflow, the concept of constraint is introduced for the purpose of the concurrency control of transactional workflow, thereby the performance and efficiency of the system has improved.

Key words: transaction; transactional workflow; concurrency control; constraint

事务是起源于数据库系统的一个概念，在数据库系统中事务的并发控制技术主要是锁机制和补偿机制。工作流是为实现业务过程自动化的一系列活动的集合，从某种角度看有类似于事务的特性，但事务的并发控制技术并不能完全适用于工作流系统，因为并不是所有的工作流实例都满足事务的特征；特别是在当前松耦合的分布环境下，锁机制几乎不可能实现，同时补偿事务也与其他事务一样可能失败[1]。为解决这一问题，Georgia 大学的 Amit Sheth 等人提出了事务工作流的概念。目前,适合长事务并发控制方法的研究主要有利它锁（Altruistic）协议、基于 NT/PV 模型的方法以及基于语义的处理方法[2]。事务工作流的研究致力于保证工作流系统在存在并发访问共享资源且其中某些活动需要当做一个整体对待情况下的正确性。

① 基金项目:国家水体污染控制与治理科技重大专项(2009ZX07528-006-05)
收稿时间:2011-04-18;收到修改稿时间:2011-05-14

1 相关研究情况

在工作流系统中怎样控制工作流实例的并发执行以避免出现读脏数据、丢失修改、不可重复读等现象是人们在并发控制领域内的研究重点。目前已有数十种工作流产品，并且每种工作流产品的并发控制机制都有所不同，但还没有一种工作流的并发控制技术像传统数据库两阶段封锁协议那样成熟；现有的工作流并发控制大多是基于对共享数据项的存取控制来实现[3]。如果在工作流系统中采用传统的数据库并发技术，按照事务的观点来调度工作流实例（即严格遵守 ACID），不仅会带来工作流运行效率低下，而且由于并不是每个工作流实例都具有事务特性，因此可能会出现不适合性；如果适当放松 ACID 特性，执行的正确性又不能保证。

文献[4]中，利用的是数据库本身的并发控制能力来实现对工作流的并发控制，用存储过程将对工作流的并发控制转换为对数据的并发控制来实现。通过设定约束和撤销约束来合理调度并发事务，避免并发事务之间的相互干扰造成的数据不一致性。但该方法过分依赖某种特定数据库管理系统本身的并发控制能力，也正因为如此，此方法对于一些不满足事务特性的工作流并不适用。

文献[5]在建模阶段采用业务流程和事务需求分开定制的方式，通过设置隔离域的隔离属性来控制各流程实例并发活动的执行，提出了采用隔离域的概念来实现并发控制；并且针对事务工作流可能长时间执行的特点放松了传统的隔离性要求，提出了松弛隔离性的概念。但工作流系统在运行过程中对数据的访问是多样的，可能对于这些数据的访问控制权限也是不同的；另外，访问这些数据的执行者也可能不同，因此对应的隔离级别也应该不同。这些都是在流程设计阶段应该考虑的问题，也就加大了流程设计的难度。

文献[6]通过扩展同名共享锁和同名互斥锁的概念给出了一种基于锁操作的面向复杂应用的工作流并发控制机制，充分考虑了同一工作流模式不同实例间的并发和不同工作流模式不同实例间的并发，适合于分布式工作流系统；但本质上并发控制仍然是基于对共享数据项的操作来实现的。

本文主要研究事务工作流的并发问题，借助指令流水思想通过引入资源约束、数据约束、任务约束的概念来实现基于约束的事务工作流的并发控制。可以看到，无论各工作流实例之间有怎样的相关（约束），工作流实例都可以直接被调度而不用等待，这就大大提高了系统效率。

2 问题分析

在一个实际应用环境中,往往会有多个工作流模式[7]存在，而在某一时刻，这些工作流模式可能会有多个实例在运行，它们可能在以不同的速度向前推进同时要以排他的方式存取某个不具有并发控制的共享资源（如内存、文件等）。一个事务工作流的执行必须将系统从一个一致性状态转换到另一个一致性状态[8]。如何协调这些排他执行需求而使每个实例的执行结果不受其他实例的影响，也不会因为资源匮乏而无法继续，这就是系统中并发控制需要解决的问题。

2.1 事务工作流的相关性引入

在处理器设计中，为达到指令的多发射与流水性，需要借助一些手段来消除指令之间的相关从而达到指令并行的目的。事务工作流之间也有类似的相关关系，譬如两个流程之间的合作关系等，为达到事物工作流并行的目的在这里我们借助类似于指令流水的思想来实现事务工作流之间的并发控制。

2.2 约束的引入

2.2.1 资源约束的引入

两个工作流实例 wf1 与 wf2 在执行期间都要对资源 A 进行访问，而资源 A 又并不具有并发控制的能力，此时需要用资源约束的方式来实现工作流之间因为资源相关的并发控制。

2.2.2 数据约束的引入

两个工作流实例 wf1 与 wf2 在执行期间，wf2 在某一步的操作需要用到 wf1 在它之前或者之后产生的某个数据结果，这样就形成了二者之间的数据相关，需要通过数据约束的方式来实现它们之间的并发控制。

2.2.3 任务约束的引入

两个工作流实例 wf1 与 wf2，它们之间存在严格的先后执行关系，否则将出现错误的结果。例如 wf2 必须在 wf1 完成之后才能开始执行，这样就形成了工作流之间的任务相关，需要通过任务约束的方式来实现此时它们之间的并发控制。

3 并发控制的设计

通过以上相关性的分析，只要两个工作流实例之

间没有相关，那么它们之间就是可以并发执行的，因此并发控制的主要工作就是如何消除这些相关或者如何处理相关来达到实现工作流的并发执行的目的。考虑到实际的业务系统需求，本文中这种并发控制主要通过任务管理器和并发控制器一起完成。

图 1 是并发控制部分的结构图，此部分主要包括工作流调度队列、任务管理器、并发控制器等主要模块；在此忽略了其他一些部分，例如异常处理部分等。并发控制器主要包含多个工作流执行保留站，保留站用来存储正在执行的工作流实例的相关信息，并负责与数据库的交互来控制工作流之间的资源相关和数据相关的处理；任务管理器通过和并发控制器的通信负责获取各业务系统需要的数据。

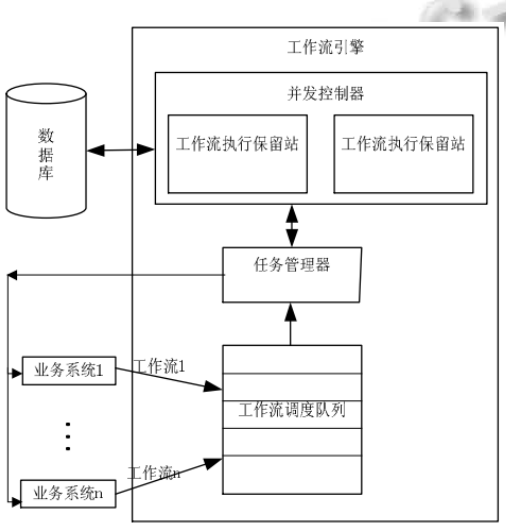

图 1 并发控制部分的结构模型

3.1 并发控制器的设计

并发控制器负责跟数据库直接打交道，数据库中预先存有冲突表等信息，例如共享资源使用情况表等，主要包括各资源的使用情况。在工作流执行过程中，有用到具体资源的地方，需要修改资源冲突表，多工作流实例正常访问数据库可以利用 DBMS 本身的并发控制能力，当发现资源使用冲突时（如资源不足）利用流程定义的异常处理来处理异常，这样可以实现各流程之间的资源约束；同时并发控制器也和任务管理器交互，这一点在任务管理器中说明。为尽可能提高系统的性能效率，本文中并发控制器的设计原则是：在保证当前正在执行的工作流实例正确执行的同时，最大限度的允许其他工作流实例的执行。因此，保留站数量的设定可以根据实际应用来确定。下面是保留站的结构：

NO	flag	st	value

图 2 保留站结构图

其中，第一部分 NO 标示保留站的编号；第二部分 flag 是一个标志域，它标示保留站被占用与否，规定 true 表示被占用；第三部分 st 是一个状态域；第四部分 value 是一个工作流执行时需要的输入。我们规定如果 st 的值是 0，那么表示 value 的值是可用的，此工作流是可以执行的；否则此工作流不满足执行条件，需要等待第 NO 号标示的保留站的工作流执行完成后的值，得到值后将 st 置为 0 并开始工作流的执行。因为都是事务工作流，执行中途不可中断。因此，可以通过保留站实现工作流之间的数据约束，而保留站关于工作流的执行规则正好实现了任务约束，因为如果有任务约束存在，那么某一工作流必须等待它的前提任务执行完成后才能得到它本身执行所需要的条件和值。

当某一保留站的工作流执行完毕，立刻与数据库和任务管理器通信，将对共享资源的使用还原，这样就保证了每个并发执行的流程拥有共享资源的时间最短；同时将执行结果写回任务管理器。当这些工作都正常完成以后，清空保留站并修改 flag，以便接收任务管理器传递的下一个工作流实例。

3.2 任务管理器的设计

允许工作流并行为了更好的提高系统性能，因此对每一个已经调度了的工作流，不管其执行条件满足与否任务管理器都直接将其送往保留站而不用在起始阶段就开始等待，因为在保留站那里将对其是否立即执行作出判断，这样就明显提高了系统处理效率。任务管理器负责和业务系统的数据交互，因为这一部分数据是业务系统直接需要的数据，因此可以不必写进数据库而直接返回给业务系统，从而大大提高了数据交换的效率。各业务系统通过自己的工作流 ID 来匹配任务管理器中的相应项，来决定数据值是否可取。任务管理器结构如下：

ID	st	value

图 3 任务管理器结构图

主要包含 3 部分，其中 ID 为工作流编号域；st 是结果状态域；value 是结果值域。当并发控制器的工作流执行保留站执行完某一工作流后在与任务管理器交互通信时根据当前工作流实例将工作流 ID 传回来，我们规定 st 为 0 表示结果值 value 可用，否则需要等待。这样，各业务系统便可以保证最终得到的数据是正确的，而不会出现读‘脏数据’的现象。

在解决资源约束问题时为了不过多的增加工作流程的设计难度应尽可能的结合实际项目利用现有方案来实现，譬如已经有利用 DBMS 所提供的服务来实现的[3]，在这里我们也是采取的这种方式，将对共享资源访问的地方交给 DBMS 去处理；数据约束可以通过工作流执行保留站来实行；任务约束通过工作流在保留站的执行规则来实现；这三部分的约束一起实现事务工作流的约束，从而消除事务工作流之间的相关而达到工作流并行的目的，至此并发控制也就实现了。

3.3 一个例子

有 3 个事务工作流实例 wf1，wf2，wf3 分别如下：

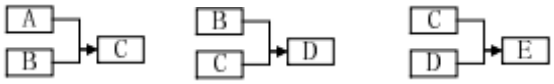

图 4　3 个事务工作流的实例

其中，wf1 与 wf2 都需要访问资源 B 形成资源相关；wf2 的输入依赖于 wf1 的输出结果 C，这样二者形成了数据相关；而 wf3 的输入 C 和 D 都必须等到 wf1 和 wf2 执行完成后才能得到，即形成了任务相关。在业务系统将这些工作流实例提交后，任务管理器首先从工作流调度队列中将这些工作流送到工作流执行保留站去执行，并假设 wf1、 wf2、 wf3 所在的保留站分别是 1、2、3 号。wf2 与 wf3 的保留站状态分别如下：

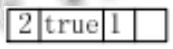

图 5　wf2 的保留站

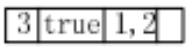

图 6　wf3 的保留站

由于 st 状态域均不为 0，因此不满足执行的条件，要等待相应 NO 号的保留站的结果；wf1 的保留站状态如下：

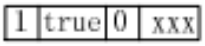

图 7　wf1 的保留站

在 wf1 获得资源 B 后就满足执行条件开始执行，一旦 wf1 执行完毕，2 号保留站就能得到相应的值而开始执行，与之类似，3 号保留站也能顺利执行。当某一保留站执行完毕，就清理保留站现场同时通知任务管理器，将工作流实例的 ID 和相关值传回以便业务系统访问取得需要的数据。

因此，不管工作流实例之间有怎样的相关，都可以直接先调度工作流而不用在开始阶段就等待，这样的并发有助于系统的处理效率。

3.4 系统应用

在辽河流域水环境风险评估与预警平台建设及示范研究课题中，工作流系统作为整个项目里的一个业务支撑平台存在，其上是业务系统，如污染源管理系统等。在实际应用中，各业务系统的许多操作流程都属于事务性工作流。初期项目成果验证，该工作流系统运行良好，对实际业务流程的处理很稳定。

4　结语

本文通过对工作流系统中已有并发控制的研究，参考处理器设计中指令流水的思想分析了工作流并发时的相关性，进而提出了三种约束的概念来实现事务工作流的并发控制；不仅实现了事务工作流的高效调度而且避免了一些丢失修改、读脏数据等典型的错误现象，达到了并发控制的目的。

参考文献

1 张民,郭玉彬,李西明,蒋郁.工作流正确性问题综述.计算机应用研究,2009,26(5):1645−1648.
2 郝丽波,李建华,夏明伟.工作流事务性研究综述.计算机工程与设计,2007,28(13):3209−3212.
3 杨威.分布式工作流管理系统并发控制的研究与实现.大连:大连理工大学,2007.
4 陈根浪,王泽兵,冯雁.用 SQL 实现工作流的并发控制.计算机工程与设计,2003,24(2):67−73.
5 郝丽波,李建华.基于隔离域的事务工作流并发控制.计算机工程与设计,2008,29(1):199−202.
6 宋宝燕,于戈,葛薇,王国仁.支持复杂应用的工作流并发控制机制.东北大学学报(自然科学版),2002,23(1): 12−15.
7 范玉顺.工作流管理技术基础.北京:清华大学出版社,2000.143−153.
8 任怡,吴泉源,戴华东,吴庆波.一种基于 QoS 的事务工作流并发调度算法.电子学报,2007,35(4):621−624.