第32卷 第18期 计 算 机 工 程 2006年9月
Vol.32 № 18 Computer Engineering September 2006
·网络与通信· 文章编号：1000—3428(2006)18—0098—02 文献标识码：A 中图分类号：TP393

Web Services 事务非阻塞 2PC 模型的实现与测试

杨海涛[1,2,3]，陈楚潮[4]，林锦堂[4]

(1. 中国科学院计算技术研究所，北京 100080；2. 中国科学院研究生院，北京 100080；
3. 广东省建设信息中心，广州 510500；4. 广州大学信息与机电工程学院，广州 510405)

摘 要：在改进 BTP 2PC with timeouts 的 timeout 设定及相关时序的前提下，提出了一类面向细粒度事务终止特性的非阻塞扩展 2PC 模型（Tx2PCt）。提供了 Tx2PCt 事务处理模型于主要商业开发平台环境（Microsoft .Net）上的测试实例及分析。测试结果支持了 Tx2PCt 事务协议的前提设定的合理性，印证了 Tx2PCt 在事务吞吐量方面的表现与预期相吻合。

关键词：Web services 事务协议模型；测试实例；例外；成功事务吞吐量

Implementation and Test of Non-blocking 2PC Model for Web Services Transaction

YANG Haitao[1,2,3], CHEN Chuchao[4] , LIN Jintang[4]

(1. Computing Technology Institute, CAS, Beijing 100080; 2. Graduate School, CAS, Beijing 100080; 3. Construction Information Center of Guangdong Province, Guangzhou 510500; 4. Information & Electrical Engineering College, Guangzhou University, Guangzhou 510405)

【Abstract】The BTP uses a two-phase outcome coordination protocol with a complete set of timeout parameters (shortly 2PCt) to ensure the overall application achieves a consistent result. With respect to the above situations this paper has posed a category of non-blocking extended two-phase commit (2PC) protocol Tx2PCt (stand for Termination properties-oriented eXtended 2PC with Timeouts) to evolve BTP's 2PCt. It mainly gives some implementation and test examples of Tx2PCt model, which are developed in a major Web services commercial developing platform—Microsoft ASP.NET. The testing results show the rationality of the Tx2PCt model's premises, and justify that the Tx2PCt's behavior is according with the expectation in the aspect of success transaction throughput.

【Key words】 Web services transaction model; Test example; Exception; Throughput of success transaction

目前，Web Services事务的处理协议/模型有许多，但是真正得到计算机业界主流厂商支持的并不多，主要有BTP[1]、WS-C/T和WS-CAF/WS-TXM[2]。这些Web Services事务协议的核心都涉及针对全局事务的两阶段提交协议(2PC)。其中只有BTP能针对 2PC提出较彻底的解决策略及语义说明。尤其是它为事务各种参与者的承诺提供了timeout说明协商机制。这对 2PC协议在Web Services环境下的应用来说是一个非常有现实意义的重要改进。本质上传统 2PC协议具有阻塞性：处于Ready状态的参与者应无期限地等待提交事件的到来，在本文中，这种无期限等待事件来“唤醒”的特性称为协议的“阻塞性”(不同于文献[3])。而当事务参与者在各阶段均有时限限定时，就不存在这种阻塞了。但是BTP协议欠缺与事务活动终止特性相关的事务时限语义考虑，并且其中一些协调在时序上是不合理的：BTP要求事务协调者先提出最小时限要求，而事务的各个参与者则在收到此要求后，立即根据是否能满足该要求作出是否参与的响应。但事务提交的时限是由各参与者反馈回的承诺维持“准备”状态的时限之最小值决定的。

1 时限非阻塞性扩展两阶段协议 Tx2PCt 模型介绍

1.1 预备

本文约定 Web Services 事务服务由资源管理器(RM)、事务管理器(TM)和事务调度器(SM)组合实现。

定义 1 活动：Web Service服务实例所施行的具有原子性的操作系列；Web Services事务：由参与者的活动集构成的特定过程，而各个活动本身就是一个原子性事务[2]。Web Services事务表示为$Trans = <S, \prec>$,表示事务*Trans*依序关系$\prec$完成所有在活动集$S=\{a_1,a_2,\ldots,a_n\}$中的活动。

预设：一般分布式事务处理模式可分为两类[3]：一是集中协调式；另一种是完全分布式。本文仅讨论第 1 类的处理模式，即假设存在一个集中的协调者即TM+SM，并且不考虑协调者失败的情况[3]。

协议前提：假设事务参与者最终都能在 ready 状态下确定事务的最终状态。

1.2 Tx2PCt 消息顺序模型

Tx2PCt模型同时也是按活动终止特性[4]调度的事务处理模型，这里依照图 1 所示的消息活动模型来说明其消息格式及相应语义：

(1)Prepare(timeoutForResponse, minReadyduration, synTime, timeoutForComfirmed)消息。其中 timeoutForResponse 表示 TM 要求事务参与者的响应必须返回的超时限制； minReadyduration 表示 TM 希望事务参与者承诺保持 Ready 状态的最低时限；timeoutForComfirmed 表示 TM 最迟在此时间内发出关于事务最终状态的消息通知，它隐含事务结束的期限。本消息设此参数主要用于实时应用，其真正有意义的值有待在 Commit 消息中产生(重设)。synTime 是 TM 规定的当次事务的同步基准时间。

作者简介：杨海涛(1962—)，男，博士生、高工，主研方向：网格事务处理；陈楚潮、林锦堂，助工

收稿日期：2005-11-24 **E-mail：**yhtyxc@hotmail.com

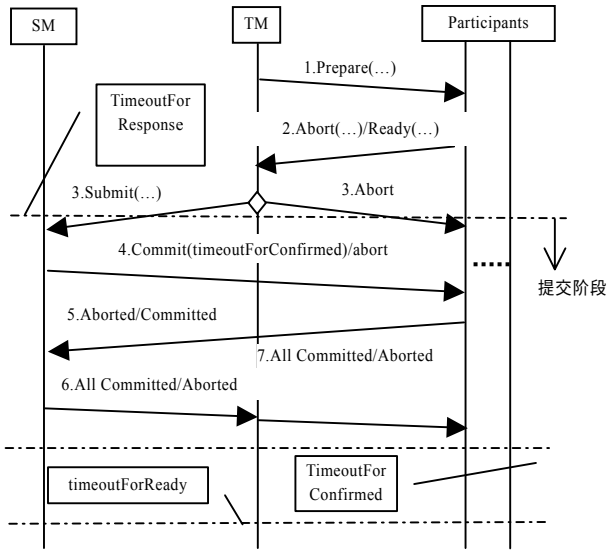


图 1 Tx2PCt 协议消息活动模型

(2)Ready(activityURI, TypeOfActivity, timeoutForReady, intendedDecision)消息。本消息仅当参与者的 timeoutForReady － timeoutForResponse>>ε 时才可发送，否则响应 Abort(timeoutForReady)。参与者向 TM 返回本消息作为对 Prepare 请求的正面响应，表示仅在 timeoutForReady 之前保持 Ready 状态，之后即独立施行 intendedDecision 说明的活动。这里 Ready 的语义有 2 种：一是只做好执行本次活动的准备，待收到 TM 的 commit 通知后才执行；二是指已执行本次活动，并准备好在 收到或自主作出 Abort 决定时就可采取补偿行动来撤消本次活动执行结果，而当直到 timeoutForReady 时 commit 决定仍有效的情况下，则撤消补偿措施。

(3)Submit(ActSet, timeoutForConfirmed)消息。这是TM在timeoutForResponse时限内得悉各参与者全参与后，若能按 min (timeoutForReady$_i$) >> timeoutForConfirmed +ε的要求重设timeoutForConfirmed的值，则立即将此消息发往SM，若不能，则Abort。此消息为SM提供当次事务要协调调度的活动集。其中, ActSet = <S, L >，S =< $a_1, a_2, ..., a_n$ > 为有序活动集，L=List of <timeoutForReadyi,T$_i$>，$T_i \in TypeOfActivity$，T_i 对应于 a_i。

(4)Commit(timeoutForConfirmed)通知事务各参与者，提交各自的活动。本模型中由 SM 发出 Commit 消息。参数 timeoutForConfirmed 告诉各参与者，SM 确认事务最终结果状态的时限。

(5)Abort, Aborted, Committed 消息。这里 Abort 消息包括了由 TM 或 SM 发出的 Abort 以及参与者响应的 Abort(timeoutForReady)。SM 接收各参与者返回的 Aborted 或 Committed 消息，并且在 timeoutForConfirmed 的时间内断定整个事务最终是 Aborted 或 Committed：发出 committed/aborted (all)消息。在协议前提条件下，TM 能保证各参与者在 timeoutForReady 时限内知道最终结果。

1.3 协议 Tx2PCt 前提的例外及处理

协议前提的例外出现在 Tx2PCt 协议实现的提交阶段：当 TM 作出 Commit 决定的时刻临近某个参与者 timeoutForReady 时，TM 发出的 Commit 消息到达该参与者时可能过了 timeoutForReady，而某些参与者则在其 timoutForReady 时限内收到 Commit 消息。

针对上述例外，在 Tx2PCt 协议中，提供了 Commit 消息核实机制：消息序号 5.Aborted/Committed 就是参与者向 TM 报告是否已收到 Commit 消息的情况；消息序号 6/7.All aborted/ Committed 则是由 TM 发往已在其 timeoutForReady 时限内收到 Commit 消息的各参与者的通知。

2 Tx2PCt 协议的实例及测试结果分析

2.1 例子及实现方案

某球赛联盟为海外球迷提供订飞机票、酒店房间、球赛门票一体化服务的网站以满足球迷的需要。要到现场观赛，海外球迷必须做到 3 个件事：a_1 ——订好到赛地的飞机票；a_2 ——订好在赛地下榻的酒店房间，a_3 ——订好球赛的门票。$a_1 \wedge a_2 \wedge a_3$ 就是一个由原子 Web 服务(活动)$\{a_1, a_2, a_3\}$ 构成的粗粒度整体事务。

2.2 吞吐量测试分析

由测试结果数据图表(图 2)可以看出 Tx2PCt 协议的事务吞吐量跟 Web 服务“承诺”的服务时间和服务器负载压力因素有很大的关系。测试结果显示，服务时限越小它的吞吐量就越大：注意到，①、②的服务时限为 40s，③、④的服务时限为 60s，⑤、⑥的服务时限为 120s；①、②的成功数大于③、④，而③、④又比⑤、⑥的成功数还大。

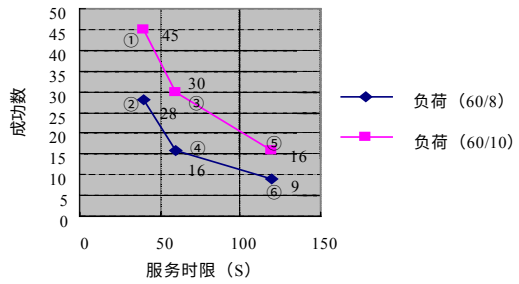
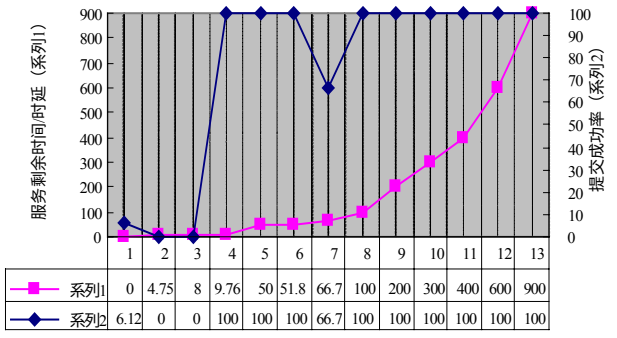


图 2 Tx2PCt 协议事务吞吐量测试结果

同时图 2 也显示了服务器的负载越重，则事务的吞吐量就越小。而传统的 2PC 模型的实现会因服务方不提供显式时限机制供整合事务实施一致性提交使用，从而在上述测试的服务时限下(最长是 120s)几乎都以某一项服务超时而告终，并且出现部分成功、其余失败的非原子性结果。

2.3 例外测试分析

图 3 的横坐标表示数据采集点。而左纵坐标表示(系列 1)：服务剩余时间与网络时延的比值；右纵坐标表示(系列 2)：相应数据点的事务提交成功率。服务剩余时间是指：timeoutForReady－timeoutForResponse。我们关注的是，这种比值的不同所导致的整体事务提交成功率的变化。



	1	2	3	4	5	6	7	8	9	10	11	12	13
系列1	0	4.75	8	9.76	50	51.8	66.7	100	200	300	400	600	900
系列2	6.12	0	0	100	100	100	66.7	100	100	100	100	100	100

图 3 Tx2PCt 协议例外测试结果

(下转第 119 页)