第17卷 第7期
2007年7月

计算机技术与发展
COMPUTER TECHNOLOGY AND DEVELOPMENT

Vol.17 No.7
Jul. 2007

TUXEDO在银行中间业务系统中的设计与实现

童旺宇,熊盛武,吴进波
(武汉理工大学 计算机科学与技术学院,湖北 武汉 430070)

摘 要:以交易中间件为框架基础的三层客户机/服务器模式,已被广泛应用在金融分布式系统中。介绍了交易中间件TUXEDO的组成及设计方法,并结合实际例子来解释如何利用TUXEDO中间件技术来开发银行中间业务系统,完成国库集中支付系统的前后台程序的开发。从而说明利用TUXEDO中间件来开发银行中间业务系统,可以大大缩减系统开发的工作量,并能充分保证系统的稳定运行。

关键词:中间件;TUXEDO;银行中间业务系统
中图分类号:TP311.5;F830.49 **文献标识码**:A **文章编号**:1673-629X(2007)07-0240-04

A Bank Intermediary System Application Using TUXEDO Middleware Technology

TONG Wang-yu, XIONG Sheng-wu, WU Jin-bo
(College of Computer, Wuhan University of Technology, Wuhan 430070, China)

Abstract: In recent years, take the transaction middleware as the frame foundation, the three-tier client / server model has been widely used in the financial distributed system. The component and design of TUXEDO middleware were introduced in this paper. At the same time, a real example was given to explain how to use the TUXEDO middleware technology in intermediary transactions of bank , and to complete the development of treasury single account system (TSA). It has been proved that using TUXEDO middleware to develop bank intermediary system can greatly reduce the workload of system development and fully ensure the system's steady running.

Key words: middleware; TUXEDO; bank intermediary system

0 引 言

长期以来,人们一直使用着"客户端/服务器"的两层结构,这种两层结构曾让无数人为之兴奋和惊叹,同时也意识到这种结构的很多弊端。为解决传统两层模式与应用需求的日益突出的矛盾,近年来,以交易中间件为框架基础的三层客户机/服务器模式应运而生,已被广泛应用在金融分布式系统中。在目前我国金融行业数据大集中的趋势下,应用中间件技术,使得开发和集成基于三层模式下的业务应用软件变得更加高效和灵活。文中将结合某银行中间业务系统的具体设计和实现,详细介绍目前应用广泛的TUXEDO交易中间件的原理及开发流程。

1 交易中间件TUXEDO

中间件(Middleware)现在是与操作系统、数据库并列的三大基础软件之一,顾名思义,它是处于操作系统软件与用户应用软件的中间,管理计算机资源和网络通信,它是个独立的系统软件或服务程序,分布式应用软件借助它在不同的技术之间实现共享资源。一般认为中间件是平台(开发平台和运行平台)和通信功能的组合。最早具有中间件技术思想及功能的软件是IBM的CICS,但是由于CICS不是分布式的产物,因此,人们一般把TUXEDO作为严格意义上的中间件产品。

TUXEDO[1]是一种分布式联机事务处理的交易中间件,通过它为应用程序提供运行环境及各种服务,如程序加载、程序启动、内存管理、负载平衡、出错恢复及一些应用管理功能,因此利用它可以快速建立三层结构的联机事务处理应用系统[2]。

TUXEDO由服务器端的事务管理器(T-TransactionCore)、客户端的工作站(WS-Work Station)、可靠队列服务(Q-Queue)、应用域(Domain)和与分布式计算环境(DCE)等几个核心部分组成。

运行于服务器端的事务管理器是TUXEDO体系

收稿日期:2006-09-25

作者简介:童旺宇(1981-),男,湖北随州人,硕士研究生,主要研究方向为软件工程;熊盛武,博士,教授,主要研究方向为人工智能。

结构的中心,它是每个 TUXEDO 服务器的核心,提供重要的分布式应用服务,如名字服务、数据路由、负载平衡、配置管理、事务管理和安全性管理等[3]。

同时,TUXEDO 提供了一个简单、可靠的队列机制,保证应用系统提交的请求和数据可在网络故障或目的服务器瘫痪等情况下也能提交到目的服务器。应用程序能将服务请求入队和出队,并可设置系统使队列中的请求自动地转发给 TUXEDO 的服务进程,并取回处理结果。这种可靠性队列作为一种资源管理器,可以和其他资源管理器(如数据库)协作,完成全局事务处理。

TUXEDO 的域模式特性把客户/服务器模型扩展到多个独立自治的应用系统。一个域既可以是一组 TUXEDO 的应用程序,也可以是一组运行在另一个非 TUXEDO 环境中的应用程序。TUXEDO 和其他中间件的互操作是利用域网关来实现的。

2　TUXEDO 在某银行中间业务系统项目中设计及实现

某银行中间业务系统项目,以现有系统为基础,实现银行客户与第三方之间的代收代付业务[4]。本系统将直接连接第三方系统,完成各种通讯协议与报文格式的转换,但不作为银行系统内各渠道的交换中心,只连接省前置或地市前置,其它银行内部交易渠道的交易统一由前置转发。从而实现中间业务系统的集中运行和管理,在提高运行效率的同时进行业务上的适应性调整和优化。

2.1　中间业务系统的结构

中间业务系统采用客户机、中间件、服务器三层结构,具体结构如图 1 所示。

客户端的操作系统为 SCO UNIX,同时需要安装 DB2 数据库、TUXEDO 的客户端和客户端应用程序。客户机上的数据库中存放了机构网点表、操作员表、交易描述表、FML 缓冲池描述表、上下传包结构描述表等并负责收集交易信息、发起交易请求、处理交易响应结果。

服务器端的操作系统是 IBM 的 AIX,同时安装 DB2 数据库、TUXEDO 的服务器端。服务器端负责对交易请求的过程进行处理并对 DB2 数据库进行操作。数据库服务器负责存储、管理各种业务数据。

2.2　TUXEDO 在中间业务系统中的设计

虽然 TUXEDO 提供许多种客户端和服务端通讯方式,但在该应用中只用两种,分别是:同步 Request/Response 模式和事务模式。对一些只是对少量表进行操作且数据量不大及表间数据无关联业务,使用同步 Request/Response 模式[5]。在实际的中间业务系统中,需要利用 TUXEDO 中间件提供事务管理方式、请求优先级和交易处理完整性保证等机制。以下为具体的设计考虑。

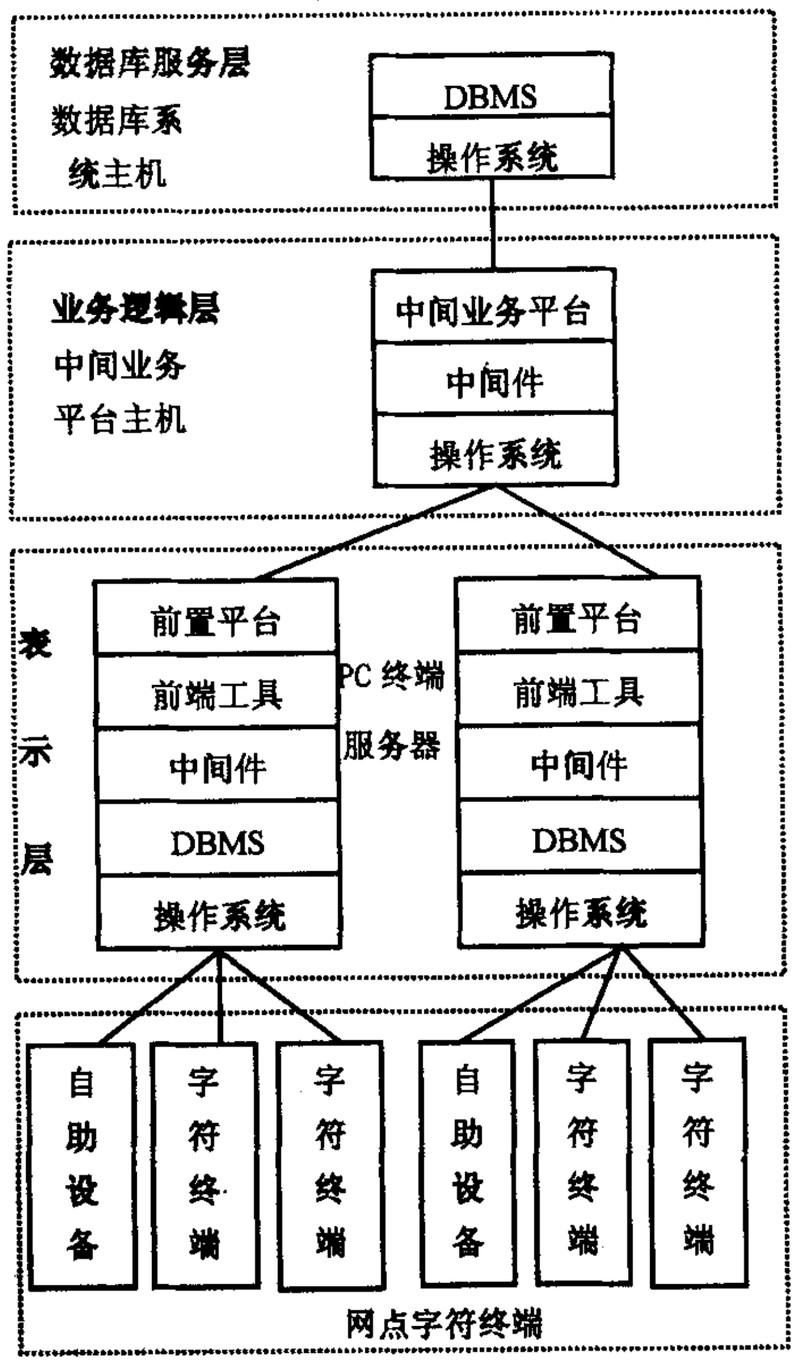

图 1　中间业务系统结构图

2.2.1　事务管理方式

由于银行业务跨接前后台两个应用系统,故采用由数据库系统、TUXEDO 中间件配合建立的全局事务机制来保证事务处理的一致性。设置了两种事务模式:

(1)前台发起的全局事务:交易由前台发起,事务在前台启动,在前台确认或者回滚,如记账类交易;

(2)后台发起的全局事务:交易由前台发起,而事务在后台启动,在后台确认或者回滚,如管理类交易。

另外还有非事务方式:交易由前台发起,后台不启动事务,如查询类交易。

2.2.2　请求优先级交易通讯管理的方式

在交易通讯管理方面,没有采用任何方式将 Service 设置优先级,根据优先级的不同赋权值,系统根据优先级权值将客户请求(Service)按优先级排队管理。而是采用最为常见的先来先服务的方式。

另外,将 TUXEDO 交易优先级设定方法有两种:一种是 TUXEDO 的系统管理员人为的设定;一种是将交易的优先级在 TUXEDO 主服务器进程中进行全系统范围的统计后设定。这两种方法都有不足之处:第一种人为设定优先级,主要凭的是个人经验;第二种方

法涉及到全系统,势必对系统的整体性能造成影响。

第三方发起的交易均采用 C/S 模式:

(1) Client 向 System/T 发出查询请求,以找到 Server 消息队列的地址;

(2) Client 根据找到的入口地址将请求发送到 Server 的消息队列;

(3)Server 处理请求,并将结果返回给 Client 的消息队列。

而由前台柜面发起的交易则统一使用域模式,一个域可以跨越多台(个)机器或处理器。所有的客户端通过 BB(Bulletin Board)了解所有的机器上提供的交易。BB 间的一致性通过 DBBL(Distinguished Bulletin Board Liaison)来保证。这两种方式的配置文件是不一样的。

2.2.3 交易完整性保证

在分布式应用系统中,由于网络的不可靠性,会出现传输过程中交易数据的丢失,造成客户端与服务端的交易不完整或数据不一致,从而导致应用系统的资金风险和信誉风险[6]。

为了防范由于通讯原因产生的风险,保证数据的一致性,把交易数据加锁与交易自动确认(系统自己确认交易结果)/冲正(若到账务主机成功,而到第三方失败。系统自己到账务主机把该笔交易销掉)结合起来考虑,利用交易数据加锁方式控制前台和后台数据一致性问题,利用交易确认/冲正方式释放交易数据加锁所占用的资源。

2.3 后台服务的设计及实现

2.3.1 后台应用服务器的划分

应用系统的后台处理部分主要提供完整的商业逻辑,负责提供具体银行业务的应用服务,并将这些应用服务按照一定的设计思路归类。每一类相对独立,形成一个业务,每一个业务在后台系统中以服务进程的方式体现,即通常所说的 APSERVER;而每一个业务含有的多种不同服务,称之为 APSERVICES;中间件 TUXEDO 负责统一管理 APSERVER,并建立前台应用请求和后台应用服务 APSERVICES 的连接关系。系统中按基本业务类、中间业务类、管理业务类三大部分来划分应用服务器[7]。

在系统中使用 6 位交易码,每个 APSERVER 总共可以提供 100 个 APSERVICES,为防止一个 APSERVER 包含的应用服务数 APSERVICES 太大(通常情况下<60),目前系统用了 15 个。某些类型的应用服务器可以用多个 APSERVER。

2.3.2 共享内存 数据池的管理

系统无论前台还是后台,均划分两类数据池(BUFFER 池):

(1)输入池(TIA)和处理池:输入池存放交易上传包要素,处理池存放交易处理过程中产生的中间要素,通常情况下,将输入池和处理池合并,这样容易做到多个原子交易共享交易输入要素和处理中间要素;

(2)输出池(TOA):输出池用于组织存放交易下传包的数据池通过中间件进行管理,采用 FML(字段控制语言)的管理模式。FML 的数据池可以容纳各种数据类型以及同种数据类型的多个实例,且可以自动扩展大小。每个应用服务器使用独立的一套数据池。

2.4 前端的设计及实现

前端界面是用 Unix 下的 C 语言开发的字符界面,操作员通过操作具体菜单或直接输入交易码的方式调用具体交易。每个交易是可以单独编写的可执行程序,通过命令行参数传入交易码信息和柜员信息。由于前台的菜单主控界面可以由 Windows 客户端工具统一编制,因此只需要编写可执行交易打包数据项,并向菜单主控中增加菜单项即可。

另外为了编写前端交易的方便,在前端提供基本的 API。因此开发前台程序时不需要了解交易与中间件的具体通信,只要调用这些接口,将根据交易码打交易请求数据包并送出即可到中间业务平台。

3 应用实例

现以一个国库集中支付系统来说明交易中间件 TUXEDO 的实际应用。

3.1 国库集中支付系统的系统结构

该系统是由中国人民银行前置机连接各商业银行前置机,再连接各商业银行主机来实现信息传递,从而铺设起中国人民银行和各商业银行之间资金传输的电子通道(如图 2 所示)。

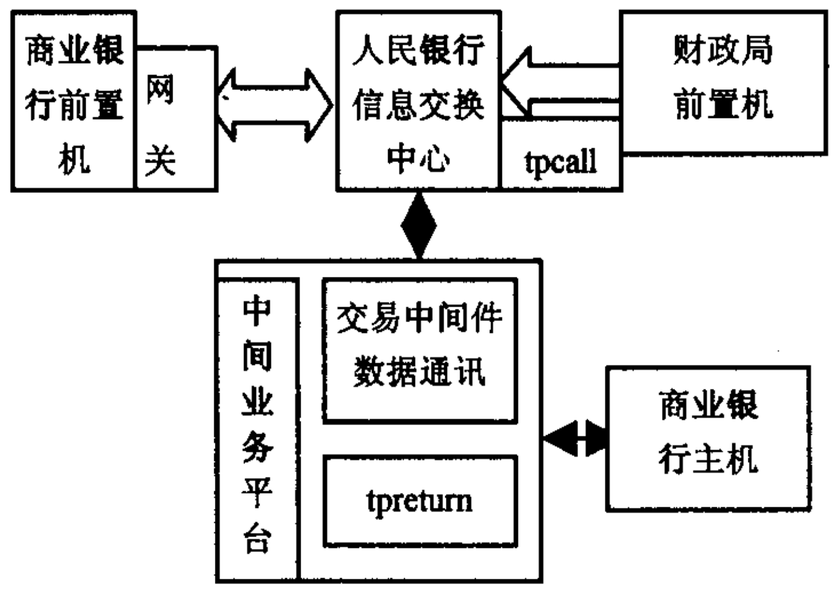

图 2 国库集中支付系统结构

银行前置机和核算中心的管理平台都是 TUXEDO 的客户端,即前端;其服务端则安装在省行、二级分行的应用服务器上,即后端。整个系统以 DB2 作数据库运行在 SCOUNIX 环境下,从而实现逻辑上三层结构而物理上多层结构的应用系统。

3.2　国库集中支付系统的实现

国库集中支付系统的网点部分由录入基本信息、查询和打印凭证三个模块十几个交易组成,而清算行除了拥有上述三个模块外,还可以审核、发送报文、日终处理及打印报表,归集行拥有清算行交易模块外,还有生成凭证等。对于一个全局事务(也称作交易),要做的就是编写客户端运行的客户进程和服务器端运行的服务进程,数据的通讯是由 TUXEDO 的 API 函数来完成的,其通讯保证也是由 TUXEDO 实现的。

这里仅以清算行的审核交易为例来说明一般全局事务例程的编写与实现。审核主要是检查支付金额要小于等于当前额度,满足要求后,要同时更新数据库中响应字段,如金额、审核状态、审核不通过时的原因等。

下面首先以客户端为例说明客户端交易处理的过程:

```
Main(argc,argv){
    tpinit(); /* 连接初始化 */
    p=tpalloc(,);/* 交易中变量分配缓冲池 */
    tpbegin();/* 交易开始 */
    /* 从前台采集输入数据,如行号、金额等,及系统取得的必要基本信息组织上传池,发起交易 */
    pubcall();/* 通讯函数 */
    /* 从下传池获取交易在后台的处理结果,并回显 */
    if()/* 取交易例表里交易号成功 */
    tpcommit()/* 发送信息 */
    else
    tpabort();/* 交易失败则放弃交易 */
    tpfree();/* 释放缓冲池 */
    tpterm();/* 中断连接 */
    return(0);
}
```

客户端程序中在 tpbegin 和 tpcommit 间以及 tpbegin 和 tpabort 间的交易请求由 TUXEDO 负责保证数据的完整性。利用 TUXEDO 命令生成客户进程。

接下来,继续编写其后台相应的服务程序,从而完成一个具体的交易处理过程。

```
/* 系统在后端已经被看成一个 SERVER,而 ClrAuditPayRtfVoucher 只是这个 SERVER 众多 Services 中一个 */
if(chkcol==false)/* 检查是否为清算行 */
    {}/* 返回前台无此交易权限 */
If(chklocvou==false)/* 是否为本清算行的凭证 */
    {}/* 返回前台不是本清算行的凭证 */
switch(chkvou_t)/* 检查凭证类型 */
    case 1: DrtPayVoucher();
    break;/* 直接支付 */
    case 2: AuthPayVoucher();
    break;/* 授权支付 */
    case 3: DrtRfdVoucher();
    break;/* 直接退款 */
    case 4: AuthRfdVoucher();
    break;/* 授权退款 */
if(condition) /* 执行数据库操作是否正确 */
    rollback; /* 出错,交易回滚 */
/* 将处理过的结果放入下传池,完成交易 */
tpreturn(TPSUCCESS,返回数据);
}
```

上面的后台程序利用 TUXEDO 命令生成服务进程,并由该系统的主控程序负责调度并完成交易所要求的功能。

3.3　系统评价

该系统经过 7 个月的开发已完成,投入运行两个多月来,一切运行正常。该系统使用中间件 TUXEDO 实现,在不同平台的分布式处理,不同客户端请求都结合在一起统一处理。它是一个功能复杂、安全性和稳定性高、开发维护容易的高可用性的系统。

4　结束语

介绍了利用 TUXEDO 中间件技术开发银行中间业务系统,它可以大大缩减系统开发及维护的工作量,并能充分保证系统的稳定运行,当前国内银行的业务系统基本上都使用了中间件技术。随着中间件技术的成熟,其应用范围越来越广泛,目前不只是应用的银行系统上,也有用于电信、医院联网、石油勘探、汽车制造等行业。

参考文献:

[1] BEA System, Inc. Education Services Document Edition 3.0 [J/OL]. BEA TUXEDO 6.5 1999. http://edocs.bea.com/tuxedo/tux65/index.htm.

[2] 黄　昕. 基于 Tuxedo 中间件的多层体系结构研究[J]. 计算机工程与应用,2003(1): 94-96.

[3] 何红波,王文军. Tuxedo 的技术特点及典型应用[J]. 信息技术,2000(5):37-38.

[4] 陈　杰. TUXEDO 中间件系统在银行代理业务上的应用[J]. 中国金融电脑,2001(1):55-56.

[5] 吴爱华. 基于 Tuxedo 的中间业务系统设计与实现[D]. 武汉:武汉科技大学,2005.

[6] 高建华. 联机事务处理应用软件的可靠性设计方法[J]. 微型电脑应用,2001(3):22-25.

[7] 徐春金. Tuxedo 中间件开发与配置[M]. 北京:中国电力出版社,2003.