• 计算机软件与算法 •

基于DDS模型的数据分发中间件的设计与实现

姚 兵， 蔡 婷， 李峻林， 赵 恒， 孙为民
(武汉数字工程研究所，湖北 武汉 430074)

摘 要：对OMG颁布的以数据为中心的数据分发服务规范模型进行了深入的研究，根据该规范提出的数据分发服务模型，设计并实现了一个高效，实时的数据分发中间件系统，以发布/订阅模式降低数据分发各节点之间的时空耦合，提供灵活的数据传输Qos控制策略，并提出数据质量和优先级的概念，提高分发系统的实时能力。

关键词：中间件；服务质量；发布/订阅；分布式；数据分发服务
中图法分类号：TP393.09 **文献标识码**：A **文章编号**：1000-7024 (2009) 03-0619-05

Design and implementation of data distribution service based on DDS

YAO Bing, CAI Ting, LI Jun-lin, ZHAO Heng, SUN Wei-ming
(Wuhan Digital Engineering Institute, Wuhan 430074, China)

Abstract: Based on an in-depth study of OMG data distribution service specification, which is a data centered distribution service standard, an efficient, real-time data distribution middleware system is designed and implemented. Using pub/sub model, the system reduces the coupling between the data distribution nodes, providing flexible Qos controlling strategy. It also defines the conception of quality and priority of data, enhancing the real-time captivity of the system.

Key words: middleware; quality of service; publish/subscribe; distribute; data distribution service

0 引 言

随着分布式系统的广泛应用，各分布节点之间信息交换的需求越来越大，信息交换的质量(可靠性，时间延迟)要求也越来越高，如何实现数据高效实时地分发是网络通信领域的一个亟需解决的问题。传统的基于C/S构架的分布式软件系统基本上都是以业务流程为中心的，数据参杂在业务流程中，随着业务流程而流动，比如CORBA[1]，DCOM，RMI等都属于这种构架。这种构架下系统大多采用RPC (remote procedure call)来完成节点之间的信息交换，数据通过函数参数或返回值的形式传送。这样传送的数据量小，效率低，更存在中心服务器瓶颈、单点失效等严重问题，不能够满足像空管系统、舰载指控系统(C2S)[2]等对系统实时性，可靠性，健壮性要求很高的应用领域。

本文根据OMG(object management group)发布的数据分发服务规范[3]基于ACE(adaptive communication environment)[4]设计并实现了一个以数据为中心的分布式实时数据分发中间件(real-time data distribution middleware)[5]。降低了通信节点之间的空间、时间耦合性；实现了节点的动态加入、退出并对节点状态进行实时监控；提供了灵活的服务质量(QOS)[6]控制策略：解决了单点失效，服务瓶颈等问题。

另外本文还提出对数据的质量进行评估，用以满足数据需求者的不同要求。给数据主题分配优先级，优先处理优先级高的数据，以满足系统对延迟敏感，实时性要求高的数据传送需求。

1 DDS理论模型

DDS框架分为两层，分别是数据本地重构层DLRL (data local reconstruction layer)[7]和以数据为中心的发布－订阅DCPS (data centered publish subscribe)[7]。DCPS层是DDS的核心和基础，负责数据的传输以及相关服务质量的控制保证等。DLRL层建立在DCPS之上，将DCPS层提供的服务进行抽象，并与底层服务建立映射关系。

1.1 数据本地重构层DLRL

在DDS的两层架构中，本地重构层DLRL处于上层，是可选层，它建立在下层DCPS基础之上，把DCPS提供的服务进行封装，映射，从而简化上层的编程实现工作。本文的主要工作在于对DCPS的研究与实现，因此不详细介绍DLRL。

1.2 以数据为中心的发布－订阅层 (DCPS)

DCPS层是DDS规范的核心，是DDS的基础设施。其模

收稿日期：2008-02-25 **E-mail**：ybing198528@gmail.com
基金项目：国家“十一五”预先研究基金项目 (513150103)。
作者简介：姚兵 (1985－)，男，湖北恩施人，硕士研究生，研究方向为分布式处理、软件构件；蔡婷 (1984－)，女，湖北孝感人，硕士研究生，研究方向为嵌入式应用；李峻林 (1966－)，男，湖北武汉人，研究员，硕士生导师，研究方向为软件工程、分布式计算、计算机应用；赵恒 (1966－)，女，湖北武汉人，博士，高级工程师，研究方向为软件工程、构件技术；孙为民 (1966－)，男，湖北武汉人，硕士，高级工程师，研究方向为软件工程、分布式计算。

型如图1所示，该层提出了“全局数据空间”这一概念。全局数据空间中的数据模型由“主题”和“类型”标识。“主题”就是在全局数据空间中惟一标志某种数据的标志符。“类型”则提供了中间件如何操纵这些数据所需的结构信息。任何一个感兴趣的应用程序都可以访问这个“全局数据空间”。数据生产者可以注册声明为这个空间的“发布者”，把生产出的数据推送到这个空间中。

类似的，想要从这个空间中获取数据的应用则声明成为一个“订阅者”。系统通过数据主题来关联发布者和订阅者，发布方向系统注册时声明其生产的数据类型，主题号，并描述提供的服务质量(QOS)。订阅方注册时则描述其需要的数据类型，主题号，并对服务质量提出要求。中间件根据双方的数据类型、主题号以及服务质量来进行关联，若匹配，则建立数据链路，这样每次发布者将新的数据对象推送到“全局数据空间”，DDS中间件将正确地把数据分发给所有的订阅者。

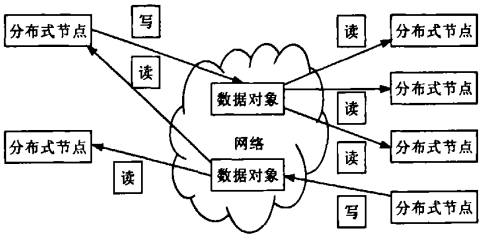

图1 DCPS模型

2 数据分发服务的设计

本系统最终需要在分布式环境下实现各个节点之间数据的正确高效分发。单个节点保持高度的自治性，各个节点之间的耦合降到最低，任何一个节点出现异常不影响整个系统的运行。

一次数据的传输过程需要完成如下任务：

(1)用户数据资源的发布与订阅；

(2)数据资源的发现与识别；

(3)配对节点选择与数据链路建立；

(4)数据高效传输；

(5)传输过程中节点实时状态监控；

(6)用户取消发布、订阅，数据链路释放与重建。

根据DDS规范，本文按照网络应用开发的经典分层模式[8]将系统实现分为4层。如图2所示。

(1)传输层：该层实现各种协议链路的数据传输，提供给上层使用，数据通过该层真正的传输到对端节点，该层是整个系统的基础设施。

(2)管理层：该层对本节点的所有信息进行管理，包括本节点发布的主题信息管理；订阅的主题信息管理；本节点关心的对端节点信息管理；数据链路的建立、释放管理；数据组织管理；数据保存管理；本节点与对端节点运行状态获取等。该层往下调用传输层提供的服务将本节点的数据以及控制命令发送到相应对端节点，同时从对端节点收取本节点需要的数据

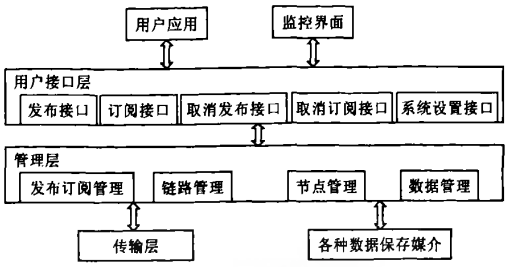

图2 数据分发服务系统构架

以及控制命令。向上则以服务的形式提供给接口层调用，响应来自接口层的用户操作指令。

(3)用户接口层：提供给用户操作接口，包括发布/订阅接口，取消发布/订阅接口，系统设置接口等，用户调用这些接口完成数据资源的发布与订阅，取消发布与退订，根据本节点的实际运行情况设置调整本节点的运行参数，包括运行线程数，数据保存策略等。

(4)应用层：即应用界面层。完成节点的状态监控显示。

3 实现关键技术

从系统功能来说，本系统需要完成链路动态管理、数据高效管理以及系统实时性保证3大任务。

对于链路管理，其难点主要体现在链路的动态建立及正确地释放。

对于数据的管理，其难点主要在于找到适当的数据组织方法，以解决数据的便利存取、数据优先级的实现、数据的本地零拷贝、多线程安全以及数据保存的问题。

对于系统实时性的实现，其难点则主要在于攻克高效的本地数据组织方法、节点之间数据投递方式以及节点对数据的处理模式。

3.1 链路管理

本系统是一个完全分布式的系统，各个节点可以运行在不同的计算机上，也可以运行在同一台计算机上，各个节点之间的数据链路是根据需要动态建立和删除的。

3.1.1 链路的建立

数据链路建立的过程就是发布订阅的配对过程。系统初始并不知道哪些节点会形成发布/订阅关系，而且发布订阅关系也是随节点运行情况动态变化的。这要求发布者/订阅者之间的数据链路必须根据节点运行状态动态建立，以适应节点的运行变化。本系统采用询问/应答/确认(三次握手)[9]、定时器技术以及超时重传机制[9]解决链路建立问题。其流程如图3所示。

定义1

- Puber_peer_set (topicID)：订阅方主题topicID的对端发布者集合。
- Suber_peer_set (topicID)：发布方主题topicID的对端订阅者集合。
- PubMap：本节点的发布主题集合
- SubMap：本节点的订阅主题集合
- TOPICQUERYMSG (topicID)：订阅方为了订阅topicID

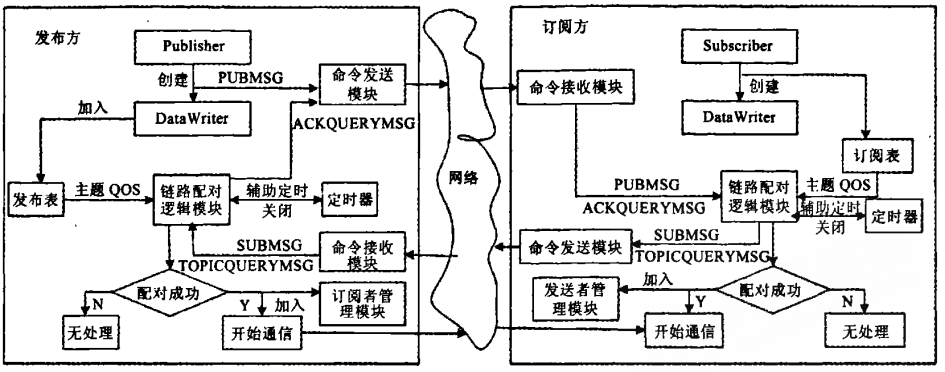

图 3　发布订阅配对流程

数据需要发送的订阅查询消息。

• ACKQUERYMSG(topicID)：发布方应答订阅方订阅查询 topicID 的消息。

• SUBMSG (topicID)：订阅方发送给发布方的确认订阅 topicID 的消息。

• PUBMSG(topicID)：发布方发布新主题 topicID 的发布消息。

发布方按照如下算法进行配对：

(1) if 有新主题 topicID 发布，发送 PUBMSG；

(2) if 收到 TOPICQUERYMSG，检查自己的 PubMap，若有满足条件的发布记录，转(3)；

(3) 初始化应答次数为 10，启动定时器；

(4) 若超时，发送 ACKQUERYMSG；

(5) if 收到 SUBMSG，转(6)，否则转(7)；

(6) 建立链路，把对端节点加入 Suber_peer_set(topicID)，配对成功；

(7) 应答次数减 1，if 次数大于 0，转 4，否则配对失败。

订阅方按照如下算法进行订阅配对：

(1) if 新的主题 topicID 订阅，转(3)；

(2) if 收到新 PUBMSG，转(8)；

(3) 订阅询问次数初始化为 10，启动定时器；

(4) 若超时，发送 TOPICQUERYMSG；

(5) if 收到 ACKQUERYMSG，则转(6)，否则，转(7)；

(6) 发送 SUBMSG，建立链路，订阅成功；

(7) 询问次数减 1，if 次数大于 0，则转 4，否则，订阅失败；

(8) 检查自己的 SubMap，若满足订阅要求，则转(6)；

3.1.2　链路的删除

数据链路的删除过程就是发布订阅解除配对的过程。系统中任何已经建立链路的双方随时均有可能退出、取消发布、取消订阅或者受其它因素影响导致链路不通。此时系统会释放链路并重新寻找节点建立新的发布订阅配对关系。本系统采用心跳技术来侦测对方的存活状态以决定是否解除连接，同时采用双方协商以及定时器技术保证双方都释放链路，使系统一致。其流程如图 4 所示。

定义 2

• UNPUBMSG (topicID)：发布方取消发布 topicID 的消息。

• ACKUNPUBMSG(topicID)：订阅方应答发布方的确认取消发布 topicID 消息。

• UNSUBMSG (topicID)：订阅方取消订阅 topicID 的消息。

• ACKUNSUBMSG(topicID)：发布方应答订阅方的确认取消订阅 topicID 消息。

取消发布和取消订阅处理流程类似，这里以取消发布为例说明：

发布方按如下算法解除配对：

(1) if 取消发布 topicID 则从 PubMap 中移除相关项，启动定时器；

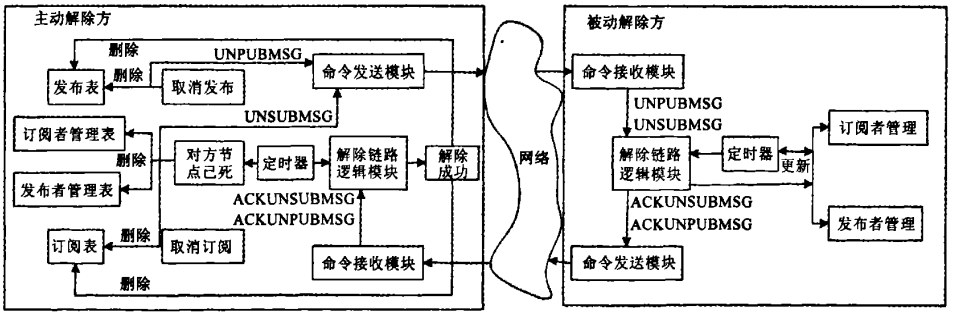

图 4　解除发布订阅配对流程

(2) 若定时器超时向Suber_peer_set(topicID)中的每个节点发送 UNPUBMSG;

(3) If 收到节点 P 的 ACKUNPUBMSG,则在 Suber_peer_set (topicID)删除节点P,若需要删除和P之间的链路,则删除链路;

(4) if Suber_peer_set(topicID)所有节点删完,则取消发布成功,否则,转(2);

(5) if 对方节点 P 已死 则在所有的 Puber_peer_set 中删除对方节点,在所有的 Suber_peer_set 删除对方节点。删除和 P 之间的链路。

订阅方按如下算法解除配对:

(1) If 收到来自节点 P 的 UNPUBMSG 则在 Puber_peer_set (topicID)删除节点 P;

(2) 向节点 P 发送 ACKUNPUBMSG;

(3) if 需要删除和 P 之间的链路,则删除链路。

3.2 数据管理

本系统以数据为中心进行信息交换,数据在系统中占据了最重要的地位,高效管理的数据对系统性能至关重要。

3.2.1 内存数据组织

本系统采用队列技术[10]、定时器技术以及锁机制实现了一个高效的消息队列来对数据进行管理,满足了系统对数据管理的要求,其实现基于如下两个类:

(1)Message_Node:代表队列中的一个节点,内部分配缓冲区用以保存需要管理的消息数据。一个 Message_Node 对象实例管理着一条消息数据。该类提供接口设置信息的优先级,用以实现在队列中根据优先级排队。

(2)Message_Queue:消息队列。由很多 Message_Node 连接而成。内部采用了定时器技术来实现进队、出队的定时,进队、出队操作会一直阻塞到操作成功或者超时失败,从而避免了一直阻塞导致系统无法工作。采用互斥量技术来对多线程操作队列进行同步,保证进队、出队操作的多线程安全。同时提供了put_prior接口实现按优先级进队,操作会根据Message_Node 实例的优先级将其插入到队列的合适位置,而不是始终插入到队尾,从而实现数据的按优先级排序。更重要的是队列不直接管理 Message_Node 实例,而是管理 Message_Node 实例的指针,从而避免了进队出队的拷贝操作,实现了数据本地零拷贝,减小了系统开销。Message_Queue 对 Message_Block 的管理结构图如图 5 所示。

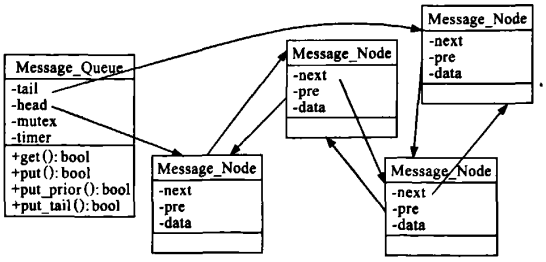

图 5 Message_Queue 结构

3.2.2 数据持久性

数据持久性策略主要解决节点之间时间耦合问题。对于每一个主题数据,均可以配置其持久性用以决定主题数据是否可重现。系统提供 4 种策略:

(1)实时数据。系统不保存数据,发布方生产出数据则直接交付系统投递,投递时在线的订阅者可获得该数据,不在线的节点则不能获取到数据,数据没有重现性。

(2)N 条有效。系统在内存中保存最新的 N 条数据。后加入的订阅者可以获取到这 N 条"历史数据",这里的 N 用户可以设置。

(3)N 秒有效。指的是数据从生产出来的时刻算起,有 N 秒的存活时间,一旦时间超过 N 秒,则数据失效。系统保存所有尚未失效的数据。后加入的订阅者可以获取到还存活的"历史数据",这里的 N 用户可以设置。

(4)永久保存。即保存所有该主题数据,后加入的订阅者可以获取到所有的"历史数据"。

系统默认将这些数据保存在内存,用户可根据设置策略、数据量及 N 设置值的大小指定数据保存媒介,比如数据库,XML 文件等。

3.3 系统实时性

DDS 设计的初衷就是为了在分布式节点之间进行数据的实时分发,所以高实时性是本系统的最终目标。

对数据分发系统来说,衡量实时性的一个最直观的变量就是从发布者生产出数据到订阅者获取到数据之间的时间间隔。假设 T 是这个时间间隔,t1 是发布者生产出数据 D 的时刻,t2 是订阅者获取到数据 D 的时刻,则

$$T = t2 - t1 \ (t1,t2 \text{ 为系统统一时钟值}) \quad (1)$$

提高系统的实时性就是要尽量减少时间间隔 T。为了便于分析,式(1)可具体表示为

$$T = T_{enq} + T_{wnq} + T_s + T_f + T_r + T_{enq} + T_{wnq} \quad (2)$$

其中:

T_{enq}:发布者生产出数据到进入到队列的时间开销。主要包括对数据进行封装,加上应用层协议头以及进队阻塞的时间开销。

T_{wnq}:数据消息在队列中等待到出队发送的时间开销,这主要和队列中处于该数据前面的待发数据消息多少有关。

T_s:消息出队到消息数据发送到本地物理网络传输媒介上的时间开销。主要指数据从用户内存空间拷贝到内核空间的时间以及数据通过网卡到达物理传输媒介上的时间。主要依赖于 OS 和硬件性能、消息数据的长度以及数据本地拷贝的次数。

T_f:数据在两个节点之间的物理媒介上的飞行时间。主要和物理距离以及网络拥塞情况相关。

T_r:接收时间。主要指数据从本地物理媒介被接收到 OS 内核空间以及从内核空间拷贝到用户空间的时间。

T_{enq}:接收端进队时间。主要指对数据进行应用层的解包,剥除应用层协议头所花费的时间以及进队阻塞时间。

T_{wnq}:在接收端队列中等待时间。主要依赖于接收队列中处于该消息前面待处理的消息条数以及消息处理流程的复杂度。

由公式可看出,只要减少其中任何一项就可以减少 T,提高系统的实时性。本文主要是依靠以下技术来减小式中各项以获取系统实时性的。

(1)采用简单应用层协议。简化应用层协议,仅仅在数据

前面加了一个表示消息类型以及消息长度的消息头。简单的协议减小了消息的封装与解封开销，减小 T_{enq} 和 T_{enq}。

(2)采用多线程技术。多个线程并发从发送端队列中读取消息进行发送，使得多条消息可以并发的向外投递从而减小消息在队列中的等待时间 T_{wnq}。同时提供注册回调接口供用户采用多线程并发处理接收到的数据，减小数据在接收端队列中的等待时间 T_{wnq}。

(3)采用消息队列管理数据指针。避免数据在本地内存的拷贝，数据从生产出来直接被拷贝到系统内核空间进行发送，减小 T_s 和 T_r。

(4)数据点对点传送。节点间的数据直接从发布者流向生产者，不经过任何中间节点转发，同时避开使用广播，缓解网络拥塞从而减小数据在网络间的飞行时间 T_f。

(5)采用优先级技术。在发布方和订阅方的数据队列均提供按优先级入队的接口，用户通过此接口可以将对延迟敏感的数据插入到队列的前部，从而获取到优先发送和处理的权利，减小 T_{wnq} 和 T_{wnq}。

(6)采用数据质量衡量技术。本文对同一个主题的数据进行质量分级，用户订阅主题时可以指定所需要数据的质量，这样系统可以自动过滤掉一些质量比较低而不符合用户需求的同主题数据，避免用户处理一些不符合要求的数据，减少系统开销，提高系统实时性。

4 相关工作比较

4.1 几种基于发布订阅模型系统的比较

由于OMG只是制定颁布了DDS规范，而没有对DDS进行商业支持，目前基于DDS发布订阅模型的系统实现很少，文献可查的有RTI公司以DDS1.0规范为基础研发的NDDS (net data distribution service)。但由于保密安全等原因，NDDS的详细性能参数没有办法获取得到。

当然，存在与DDS相近的其它发布订阅模型，例如JMS[11]，TIBCO Rendezvous[12]等，它们各都有自己的特点。表1是本系统与其它几种常用的发布订阅模型系统的比较。

表1 DDS中间件与其它发布/订阅模型系统的比较

比较项		系统		
		本系统	IBM MQ Series	TIBCO Rendezvous
模型		DDS	JMS	Rendezvous
体系结构		分布式	集中式	分布式
传输媒介		无	中心数据库	无
服务质量	单点失效	无	有	无
	持久性	√	√	—
	优先级	√	√	√
	可靠性	√	√	√
	事务处理	√	√	单一会话
跨语言		√	√	√
跨平台		√	√	√
异步数据传输		√	√	√
通信协议支持		TCP/IP, UDP/IP,SNA 等	TCP/IP, SNA 等	TCP/IP

4.2 系统特点

本系统采用全分布式体系结构，发布者和订阅者通过主题和服务质量进行关联，实现通信。较之其它的数据交换系统，本系统具有如下特点：

实时性：本地线程之间数据零拷贝、多线程同时发送数据、数据点对点传送、多线程同时处理数据以及优先级机制保证了通信的实时性。

空间松耦合性：发布方只管发布主题并将数据推出，而不需要了解数据的具体流向；订阅方只需要订阅主题并定义好数据处理流程，而不需要知道数据来源。

时间松耦合性：系统提供对数据持久性的支持，解决了发布方，订阅方必须同时在线的耦合。发布方只管推出数据，订阅方也不需要了解数据何时到达。

可扩展性：节点可以随时加入或退出系统，也可以随时增加、减少主题，而不会影响到系统正常运行。

健壮性：全分布式结构，不存在服务器瓶颈以及节点失效等问题。

灵活性：多种QOS策略以及多种底层传输模式支持。

5 结束语

本文根据DDS规范设计了一个基于ACE的分布式实时数据分发系统，屏蔽了操作系统之间的差异，降低了各节点之间的时空耦合，并提供了数据优先级、数据质量、数据持久性以及传输服务方式等Qos控制策略，满足了分布式应用系统对信息交换实时性、灵活性的需求。下一步工作将加强对DLRL层的研究实现，提供对数据类型的支持，提供更灵活的Qos策略，研究高效的数据加密/解密算法，提供数据传输过程中数据安全保密控制，进而针对主题管理分配负担问题实现基于数据语义[13]的发布/订阅机制。

参考文献：

[1] Object Management Group.Real-time CORBA specification[R]. Version1.2, 2005.

[2] Joe Schlesselman, Brett Murphy. Publish-subscribe approach aids military Comms [Z]. COTS Journal, www.timesys.com, 2003.

[3] OMG. Data distribution service for real-time systems specification[S].www.omg.org,2004.

[4] Douglas C Schmidt, Stephen D Huston.C++ Network programming [M]. Beijing: Publishing House of Electronics Industry, 2004:22-261.

[5] Mike Rogosin.Design strategies for real-time data in distributed systems[M].2005:158-171.

[6] Network data distribution service (NDDS) real-time publish-subscribe network middleware[Z]. http://www.rti.com/.

[7] Gerardo Pardo-Castellote. DDS spec outfits publish-subscribe technology for the GIG[Z].COTS Journal, 2003.

[8] Stephen D Huston, James CE Johnson,Umar Syyid. The ACE programme's guide, practical design patterns for network and systems programming[M].Beijing: Publishing House of Infopower,2004:25-155.

(下转第682页)

图5展示了本文的树组件在TM平台中的应用界面，从图中可以看到本文的组件具有较好的UI展示力，其中左侧图是树组件的主操作界面，它提供了丰富的操作菜单以供用户完成对树的各种操作；右上图是带有Checkbox的树，用户能够通过选中Checkbox以选中树中的某些节点，这些被选中的节点ID可以被发送到服务器端用来建立节点间的关联关系；右下图是树状表格，它提供了以树状结构显示表格的表现形式。

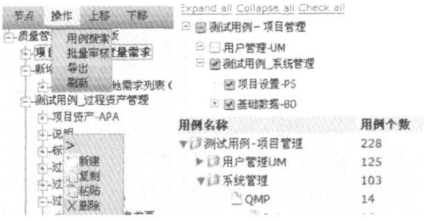

图5 树组件、Checkbox树和树状表格

5 结束语

本文的组件较好地实现了客户端和服务器程序的分离。用户可以在不改变服务器端程序的基础上，方便的更换客户端表现层技术，比如使用Flash。组件服务器端的业务逻辑提供了较好的对外接口，不仅方便用户对业务逻辑进行扩展，而且，可以将现有接口进行包装以发布成Web Service，进而使得用户可以通过更多的方式调用组件服务器端业务逻辑。

此外，本文的树组件具有丰富的UI表现力，能够满足大部分用户对树型结构对象操作的要求。组件采用的AJAX异步通讯机制，大大降低了服务器和客户端通讯的数据量。Server Push技术的使用，较好的解决了AJAX中数据一致性的问题，保证了客户端数据最新。这些技术的使用都大大提高了Web应用中用户的体验。

参考文献:

[1] Bert Vermeulen.Choosing a web tree component[EB/OL].http://www.surftree.com/WebTree.pdf.

[2] 陆海晶,刘万军.基于Ajax的Web应用技术的研究与实现[J].科学技术与工程,2007,7(3):415-418.

[3] Rao S S,Vin H M,Tarafdar A.Comparative evaluation of server-push and client-pull architectures for multimedia servers [EB/OL].Austin: Technical Report, Department of Computer Sciences,University of Texas,1996.http://citeseer.ist.psu.edu/rao96comparative.html.

[4] 李然,高会生,徐扬,等.Push技术在基于Web的网络管理中的应用[J].电力系统通信,2004(12):12-15.

[5] Rod Johnson.Expert one-on-one J2EE design and development (programmer to programmer)[M].New ed.Wrox,2002:113-178,251-284,441-514.

[6] Deepak Alur,John Crupi,Dan Malks.Core J2EE patterns: Best practices and design strategies[M]. 2nd ed. Prentice Hall PTR, 2003: 22-29,390-407,408-420.

[7] Jagadish H V,Laks V S Lakshmanan,Divesh Srivastava.Hierarchical or relational? A case for a modern hierarchical data model [C]. Proceedings of the Workshop on Knowledge and Data Engineering Exchange. Washington,DC,USA:IEEE Computer Society, 1999.

[8] 张勇, 马玉祥. JSP中的定制标记库介绍 [J]. 现代电子技术, 2004,27(24):100-101.

[9] Christian Heilmann.Beginning JavaScript with DOM scripting and Ajax: From novice to professional: Beginning: from novice to professional[M]. Apress, 2006:61-84,299-342.

[10] 赵永屹,宿红毅,胡韶辉.基于AJAX与J2EE的新型Web应用的设计与实现[J].计算机工程与设计,2007,28(1):189-192.

[11] Periodic Refresh [EB/OL]. http://ajaxpatterns.org/Periodic_Refresh.

[12] Gerschefske M,Gray J,Yoo J.An analysis of AJAX with SOAP and comet [EB/OL].cs.uccs.edu.

[13] 周婷.Comet:基于HTTP长连接的"服务器推"技术[EB/OL]. IBM developerWorks 中国, 2007-08-31. http://www.ibm.com/developerworks/cn/web/wa-lo-comet/index.html.

[14] Meteor Website. Interaction modes [EB/OL]. http://meteorserver.org/interaction-modes/.

[15] Greg Wilkins. Ajax, Comet and Jetty [EB/OL].http://www.webtide.com/downloads/whitePaperAjaxJetty.html,2006-06-11.

[16] Erich Gamma,Richard Helm,Ralph Johnson, et al.Design patterns: Elements of reusable object-oriented software[M].李英军,马晓星,蔡敏,等译. Addison Wesley/Pearson.北京:机械工业出版社,2005:194-200.

(上接第623页)

[9] Richard Stevens W. Advanced programming in the UNIX environment volume 1[M]. Beijing: Publishing House of Tsinghua, 2001:188-432.

[10] Douglas C Schmidt,Stephen D Huston.C++ network programming, volume 2[M].Beijing: Publishing House of Electronics Industry, 2004:32-314.

[11] 王传胜,李乔儒.基于JMS的消息服务的研究与开发[J].计算机工程与设计,2005,26(12):3423-3426.

[12] TIBCO Corp. TIB/Rendezvous(White Paper)[Z].http://www.rv.tibco.com/whitepaper.html, 2000.

[13] Banavar G,Chandra T,Mukherjee B,et al.An efficient multicast protocol for content-based publish-subscribe systems[C]. Proceedings of the 19th International Conference on Distributed Computing Systems,1999.