

# 分布式仿真系统的自适应通信中间件设计

邬国安<sup>1,2</sup>, 赖兰剑<sup>1,2</sup>, 张大海<sup>1,2</sup>, 陈鼎才<sup>1,2</sup>

(1. 通信信息控制和安全技术重点实验室, 浙江 嘉兴 314033;

2. 中国电子科技集团公司第三十六研究所, 浙江 嘉兴 314033)

**摘要:**为了解决分布式仿真系统中如何实现实时和按需分发信息的问题,在分析了分布式仿真系统的通信需求的基础上,文中提出了一种用于分布式仿真系统中的通信中间件的设计方法。该设计借鉴了高层体系结构(HLA)的发布订阅的思想,使用了华盛顿大学发布的开源的自适应通信环境(Adaptive Communication Environment)组件来完成此设计,并使用“主动对象”的方法解决了并发执行的效率问题。与以往使用的通信方法相比,它具有实时高效、可动态通信和跨操作系统平台的特点。实现了分布式仿真系统中信息的实时和按需分发。

**关键词:**自适应通信环境;订阅发布;动态通信;分布式仿真系统

**中图分类号:**TP31

**文献标识码:**A

**文章编号:**1673-629X(2012)12-0075-05

## Design of Adaptive Communication Middleware in Distributed Simulation System

WU Guo-an<sup>1,2</sup>, LAI Lan-jian<sup>1,2</sup>, ZHANG Da-hai<sup>1,2</sup>, CHEN Ding-cai<sup>1,2</sup>

(1. Key Laboratory of Communication Information Control and Security Technology, Jiaxing 314033, China;

2. No. 36 Research Institute of CETC, Jiaxing 314033, China)

**Abstract:** In order to resolve how to achieve real-time and on-demand distribution of information in a distributed simulation system, on the basis of analyzing the communication requirements of distributed simulation systems, present a design method of communication middleware for distributed simulation systems. The design is based on publish subscribe ideas of HLA, uses the open source ACE (Adaptive Communication Environment) components released by university of Washington to complete the design and applies "active object" to solve the problem of concurrent execution efficiency. Compared with the previous method of communication, it has characteristics such as the real-time and efficiency, dynamic communication and cross-operating the platform. Achieved the target of the real-time and on-demand in a distributed simulation system.

**Key words:** ACE; Pub-Sub; dynamic communication; distributed simulation system

## 0 引言

随着传感器网络在战场上的应用,以及民用的物联网等行业的兴起,如何将一个网络节点的信息实时地传送到对此信息感兴趣的节点,以及新的网络节点如何能快速地加入到网络中进行有效工作成了研究热点。目前,对于在分布式仿真系统中如何实现信息的实时和按需分发,是众多研究人员关注的热点问题之一<sup>[1]</sup>。

文中在分析了分布式仿真系统的通信需求的基础上,借鉴了高层体系结构(HLA)的发布订阅的思想,通过研究华盛顿大学发布的开源的自适应通信环境

(Adaptive Communication Environment, ACE)组件,并使用此组件实现了一种适用于分布式仿真系统,具有实时高效、跨计算机操作系统平台,并可适应通信动态变化的通信中间件(文中简称为通信中间件)。

## 1 问题的提出

通信问题是分布式仿真系统中非常核心的问题,直接影响到整个仿真系统的效率和生命力(可扩展性)<sup>[2]</sup>。

根据分布式仿真系统的特点,文中的通信中间件要解决以下几个问题:

1) 通信速度问题,即信息要实时高速地发到对此信息感兴趣的节点上;

2) 动态通信问题,即仿真系统要能自动适应新的仿真实体的动态加入,即在新的仿真实体加入仿真系

收稿日期:2012-04-08;修回日期:2012-07-13

基金项目:总装预研基金(9140C1303010903)

作者简介:邬国安(1976-),男,工程师,主要从事特种通信仿真技术研究。

统时,在不对已有的仿真系统做改动的前提下实现将新的仿真实体的信息发送给对此信息感兴趣的现有的仿真实体,同时将已有的仿真实体的信息发送给对其感兴趣的新的仿真实体。

3) 跨操作系统平台问题,即仿真系统的各仿真实体可运行在不同的操作系统平台上,根据实际的应用需求,需要能够支持目前已有的主流操作系统如 WINDOWS、LINUX 和 VXWORKS 操作系统。

目前分布式仿真使用的通信方式主要有两种:

1) 基于客户端/服务器(C/S)的通信模型,通信双方通过远程调用的方式进行数据通信。这种传统的 C/S 模式的解决方案虽然已经有了自己相应的协议标准,在很多系统和应用中广泛使用<sup>[3]</sup>,但这些方案无法适用于通信可动态变化的通信模型。

2) 高层体系结构(HLA)方案,它的优点是能够进行智能分发,可以满足通信的动态变化,但它的性能不好,满足不了大容量的高速实时通信的需求<sup>[4]</sup>。同时它的跨操作系统平台性也不好。

显然,这两种方式都解决不了上述的 3 个问题。

所以设计了通信中间件,借鉴 HLA 中的发布/订阅的思想(如图 1),使通信中间件能解决动态通信的问题;在实体间的数据通信时使用 P2P 通信,才使通信的实时性最好,同时考虑到要面对多个仿真实体大规模的并发数据通信问题,使用了并发处理机制,从而保障整个仿真系统的高效通信,使通信中间件能解决通信速度问题;使用了跨平台的 ACE,解决了跨操作系统平台问题。下面就一一详细论述以上解决方案。

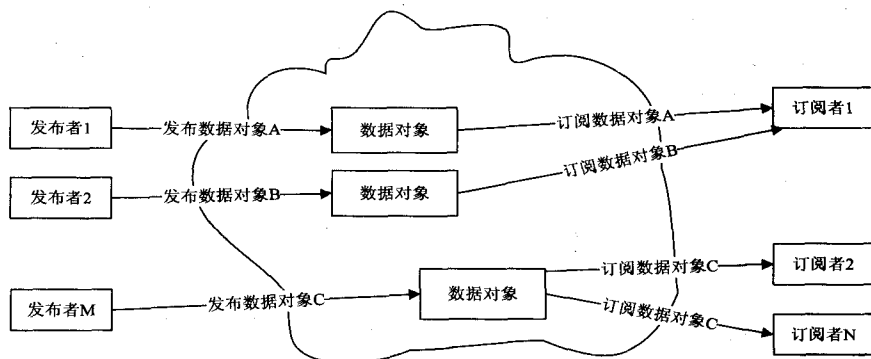


图 1 发布/订阅机制

## 2 ACE 介绍

ACE 是华盛顿大学发布的开放源码的软件开发

框架,主要用于高性能和实时通信服务和应用的设计,它通过一系列的接口屏蔽了各操作的细节,使得它具有跨操作系统平台的特点。另外,它的封装大多数都是使用内联函数,使得它在不同的操作系统平台上运行均具有很高的效率<sup>[5]</sup>。

ACE 的目标用户是高性能和实时通信服务和应用的开发者。它简化了使用进程间通信、事件多路分离、显式动态链接和并发的 OO 网络应用和服务的开发<sup>[6]</sup>。此外,通过服务在运行时与应用的动态链接,ACE 使系统的配置和重配置得以自动化。

ACE 的特点:

- \* 具有很好的可移植性(跨操作系统平台)<sup>[7]</sup>;
- \* 更好的软件质量;
- \* 更高的效率和可预测性。

ACE 体系结构如图 2 所示,ACE 具有分层的体系结构。在 ACE 框架中有三个基本层次<sup>[8]</sup>:

- \* 操作系统适配层;
- \* C++ 包装层;
- \* 框架和模式层。

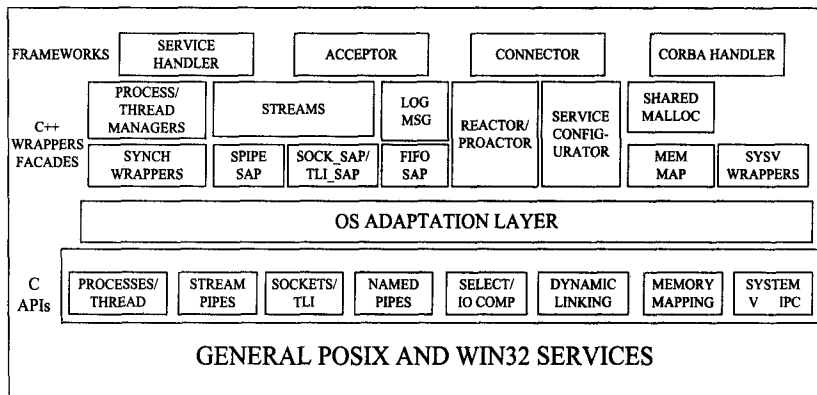


图 2 ACE 的架构图

### 2.1 操作系统适配层

操作系统适配层是位于本地操作系统 API 和 ACE 之间的代码层,它实现 ACE 与平台依赖性屏蔽开来,从而使得通过 ACE 编写的代码保持了相对的平台无关性。使得 ACE 具有很好的可移植性(跨操作系统平台)。目前 ACE 可跨的操作系统平台包括:实时 OS(VxWorks、LynxOS、pSoS 和 Chorus)、大多数版本的 UNIX(HP-UX 9.x, 10.x 和

11.x; SunOS 4.x 和 5.x; SGI IRIX 5.x 和 6.x; DEC UNIX 3.x 和 4.x; AIX 3.x 和 4.x; DG/UX; Linux; SCO; UnixWare; NetBSD 和 FreeBSD)、Windows 系列(WIN2000、XP 和 WIN SERVER 2003、2008 等)。

2.2 C++包装层

C++包装层包括了并发和同步,IPC,内存管理,定时器,信号(事件)处理<sup>[9]</sup>,文件系统,线程管理的 C++ 包装,这是 ACE 代码量最大的一部分,大约占了 ACE 总源码的一半。C++包装层可用于构建类型安全和高效率的 C++应用。

2.3 ACE 框架组件

ACE 框架组件是一个比 C++包装层高级的网络编程框架,它对 C++包装层进行了封装,并对其进行了增强。它实现了对分布式服务的动态配置。

ACE 框架组件由以下组件组成:

事件多路分离组件:ACE 的反应器(Reactor)和前摄器(Proactor)是可扩展的面向对象的多路分离器,通过对不同类型的事件分派给其专有的处理器,实现了对基于定时器、信号同步、I/O 等多种类型事件的响应。

服务初始化组件:ACE 接受器(Acceptor)和连接器(Connector)组件通过 SVC\_HANDLER 和 \_ACE\_PEER\_ACCEPTOR(或 \_ACE\_PEER\_CONNECTOR)分别实现了初始化完成后的通信服务的业务处理和通信的初始化,实现了二者的去耦合。

服务配置组件:ACE 的服务配置器(Service Configurator)实现了对应用的配置,使得在安装时和/或运行时动态装配应用的服务<sup>[10]</sup>。

分层的流组件:ACE 的流组件(Stream)对由分层服务组成的通信软件应用的开发进行了简化,像用户级协议栈的开发使用流组件将会更快捷。

3 自适应通信中间件的设计

3.1 系统的工作流程

3.1.1 发布和订阅

借鉴 HLA 的思路,引入了“发布”和“订阅”概念,发布是指数据的发送方先向中间件声明它要发送的数据种类。订阅是指数据的接收方向中间件声明它要接收的数据种类<sup>[11]</sup>。发布和订阅都是通过“主题”关键字进行声明。然后中间件服务器根据发布和订阅关系,通知主题配对成功的双方(即同一主题的发布方和订阅方)直接建立网络通信(即 P2P 链路),发布方发送数据时直接通过 P2P 链路发送(发布与订阅是 N - N 的映射关系,所以发送时可能会对多个 P2P 链路发送)。发布的时序图见图 3。订阅的时序图类似。

3.1.2 数据通信

数据发送前一定要先进行发布,发送时使用已建立的 P2P 链路发送给订阅方。数据接收时,通信中间件收到数据后,通过回调函数通知订阅方。为防止订阅方处理数据时影响通信中间件的性能,使用了“主动对象”的方法进行并发处理。

3.1.3 关闭通信

仿真节点在不需要通信时,要关闭通信。对发布方来说,就是注销发布,对订阅方来说,就是注销订阅。

3.2 并发执行的解决方法

由于通信中间件同时要完成多个仿真实体间的通信任务,如果使用串行执行,那么整个系统的通信将会成为整个系统的瓶颈,所以必须要使用并发执行。在实现时并发执行的最小单位是一个通信链路。在并发

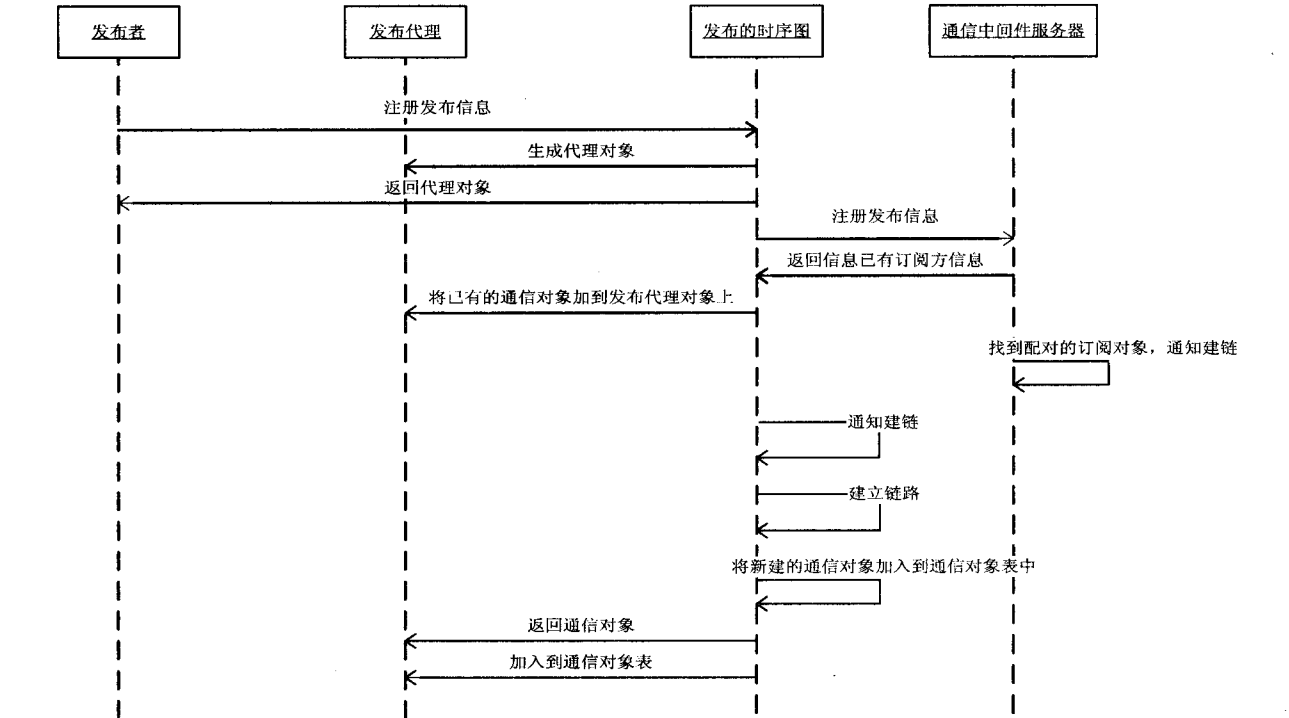


图 3 发布的时序图

执行的实现时引入了“主动对象”的概念。即通过分离方法(或函数)的调用请求和实现,使客户对该方法(或函数)的调用看起来与平常的方法(或函数)调用是一样的,但其在具体的实现时是不同的,该方法(或函数)被自动转换为请求对象,并存入队列中。在另一个线程中读取队列,将其取出并执行方法的实现。主动对象由以下组件组成:“代理”表示对象的接口,“仆人”提供对象的实现。通过使代理运行在客户的线程中,仆人运行在另一个独立的线程中,从而实现了方法(或函数)的并行运行。代理将方法调用转换为方法请求,并提交给调度者按预定的存储策略存储在“启用队列”中,“调度者”(它与仆人运行在同一个线程)持续不断地查询启用队列,当发现启用队列中存在可运行的方法请求时,就让它们出队列,并启用个人中其对应的实现,实现完成后将方法执行的结果返回给“期货”,客户可通过查询期货来获得返回结果。主动对象的时序图见图 4。

主动对象模型中有以下参与者:

代理:代理是客户访问方法的接口,即方法请求,它是运行在客户的线程中<sup>[12]</sup>。当客户调用方法请求时,代理将方法请求转换为请求对象,并将其存储在调度者的启用队列上。

方法请求:方法请求用于将方法调用转换为请求对象,主要是将方法调用所需的信息(上下文信息),比如代码和参数等,封装到请求对象中。它的基类是方法请求类,此基类定义了方法请求类的接口(即虚函数)。此基类中还有守卫方法,用于判断该方法请

求的各种同步约束是否满足。对于每个主动对象方法,必须要创建其抽象方法类,这些类由代理在其方法被调用时创建,并且要包含方法调用和返回结果时需要的各种信息。

启用队列:启用队列是一个方法请求对象的队列,代理在客户的线程中将方法调用转换为方法请求对象后加入启用队列中,仆人在其独立的线程中从启用队列中取出方法请求对象,并执行方法请求对象的实现<sup>[13]</sup>。所以它是在两个线程中并发运行的。

调度者:调度者负责管理启用队列中的出队策略,它运行在与客户线程独立的线程中。它的调度策略基于各种标准,比如先到先出、后到先出、同步约束以及优先级排序等。调度使用各个方法请求对象类的守卫方法来判断该方法请求的各种同步约束是否满足。

仆人:仆人是方法请求的实现部分,即它实现方法请求的执行。同时它还定义主动对象的状态和行为。调度者在执行相应的方法请求时调用仆人方法,从而实现了方法请求的执行,所以仆人和调度是在同一个线程中。另外,方法请求还可使用仆人的其它方法来实现守卫方法。

期货:期货是用于管理方法调用的返回结果的值和状态的对象。在客户调用方法时,代理立刻将期货返回给客户。当方法请求执行完毕,其返回结果的值会存储到期货中。客户可以通过轮询或阻塞来访问期货的值,轮询模式时,客户可通过期货的状态来获悉方法请求是否已经返回结果;阻塞模式时,方法请求返回结果后才会通知客户结果已返回,在此之前一直处于

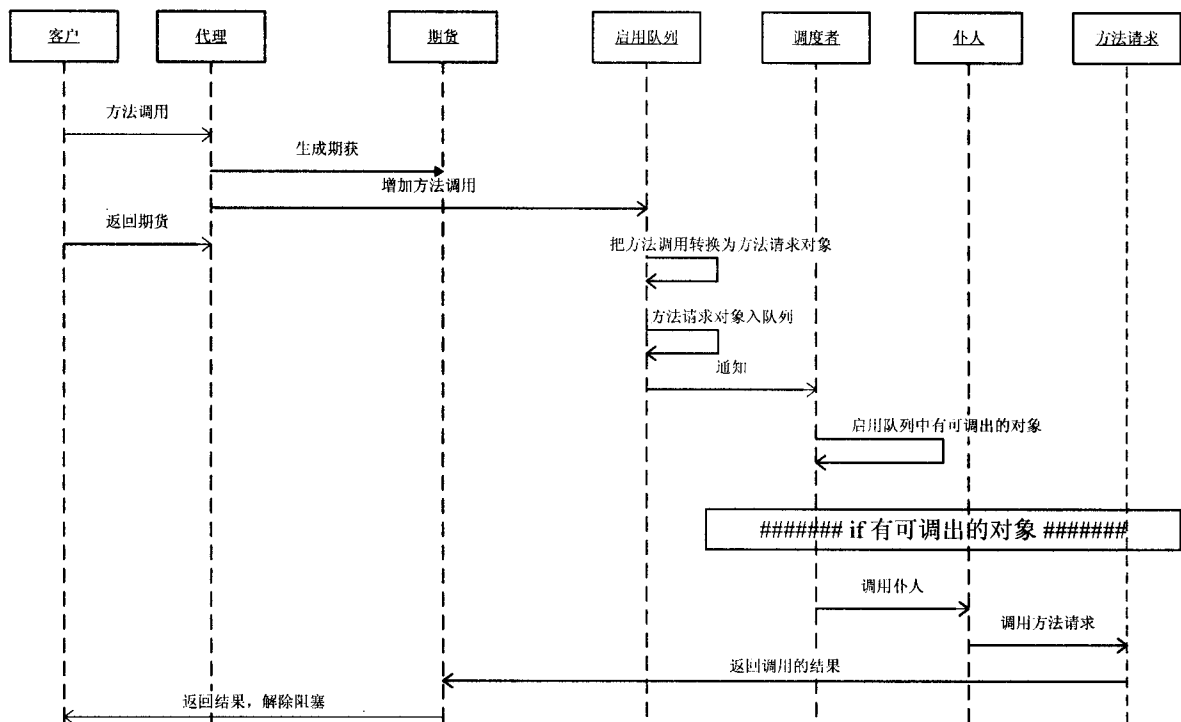


图 4 主动对象的时序图

阻塞状态。

### 3.3 通信中间件的调用接口

1) 设置通信中间件服务器的参数。

```
SetServerParam( char * pIpAddress, unsigned int *
pvPort, unsigned int uiPortCount );
```

用于创建与通信中间件服务器的通信,包括 IP 地址,端口号(可多个)。

2) 注册发布信息。

```
RegPubTraffic( TrafficBaseInfo varTrafficBaseInfo );
```

用于发布者向通信中间件服务器注册发布信息。参数是一个结构体,包括信息的主题、通信优先级。

3) 取消注册发布信息。

```
UnregPubTraffic ( TrafficBaseInfo varTrafficBaseInfo );
```

用于发布者向通信中间件服务器取消注册发布信息。参数是一个结构体,包括信息的主题、通信的优先级。

4) 注册订阅信息。

```
RegSubTraffic( TrafficBaseInfo varTrafficBaseInfo );
```

用于订阅者向通信中间件服务器注册订阅信息。参数是一个结构体,包括信息的主题、通信的优先级。

5) 取消订阅信息。

```
UnregSubTraffic ( TrafficBaseInfo varTrafficBaseInfo );
```

用于订阅者向通信中间件服务器取消注册订阅信息。参数是一个结构体,包括信息的主题、通信的优先级。

6) 发布信息数据。

```
SendInfoData ( char * pSendBuff, unsigned int
uiBuffLen );
```

用于发布者对外发送已发布的信息数据。参数是信息数据的指针及信息数据的字节数。

7) 装载回调函数。

```
InstallReadCallbackFun ( TrafficBaseInfo varTrafficBaseInfo, CommReceProcBase * pFun );
```

用于订阅者向通信中间件代理挂载收到订阅的数据时的回调函数。参数包括信息的主题、通信的优先级和回调函数指针。

8) 卸载回调函数。

```
UninstallReadCallbackFun ( TrafficBaseInfo varTrafficBaseInfo, CommReceProcBase * pFun );
```

用于订阅者向通信中间件代理卸载收到订阅的数据时的回调函数。一旦卸载后,在此通信中间件代理中收到此主题的数据后将不做处理。一般与取消注册订阅信息配合使用。参数包括信息的主题、通信的优

先级和回调函数指针。

## 4 结束语

本方案通过引入发布订阅的概念,解决了通信可动态变化的问题;使用 ACE 中间件,实现了应用的跨平台,经测试支持的平台包括: WINDOWS SERVER 2003、WINDOWS7 专业版、LINUX(FEDORA 13)和 VX-WORKS V6.7;通过引入主动对象的设计方法,实现了各仿真节点之间并行、高效的网络通信。下一步工作中,将把各种高速总线(如 PCI-E, RapidIO 等)的通信加入到此中间件中,以实现在实物/半实物仿真中使用此中间件解决通信的问题,最终实现将此产品嵌入实装产品中,为实际的装备解决自适应通信的问题。

### 参考文献:

- [1] 张大海, 赖兰剑, 陈鼎才. DDS 在分布式系统仿真中的应用[J]. 计算机技术与发展, 2011, 21(3): 250-253.
- [2] 陈鼎才, 王敏. 一种基于仿真的装备模型可视化建模方法研究[J]. 计算机技术与发展, 2011, 21(1): 238-241.
- [3] Schmidt D C, Levine D L, Mungee S. The Design and Performance of Real-time Object Request Brokers[J]. Computer Communications, 1998, 21(4): 294-324.
- [4] 杨传顺, 王学万. 实时数据分发系统的服务质量控制的研究[J]. 计算机技术与发展, 2011, 21(5): 231-234.
- [5] Schmidt D C, Huston S D. C++ Network Programming, Volume 1: Mastering Complexity with ACE and Patterns[M]. [s. l.]: Addison Wesley, 2001.
- [6] Johnson R, Foote B. Designing Reusable Classes[J]. Journal of Object-oriented Programming, 1988, 1(6): 22-35.
- [7] Huston S D, Johnson J C E, Sygid U. ACE Programmer's Guide: The Practical Design Patterns for Network and Systems Programming[M]. [s. l.]: Addison-Wesley, 2003.
- [8] Schmidt D C, Huston S D. C++ Network Programming, Volume 2: Systematic Reuse with ACE and Frameworks[M]. [s. l.]: Addison-Wesley, 2002.
- [9] Stevens W R. UNIX Network Programming[M]. 2nd ed. [s. l.]: Prentice Hall, 1997.
- [10] Schmidt D. Patterns for Concurrent and Networked Objects, Volume 2: Pattern-oriented Software Architecture[M]. [s. l.]: Wiley, 2000.
- [11] 马建刚, 黄涛, 汪锦岭, 等. 面向大规模分布式计算发布订阅系统核心技术[J]. 软件学报, 2006(1): 134-147.
- [12] Gamma E, Helm R, Johnson R, et al. Design Patterns: Elements of Reusable Object-oriented Software[M]. [s. l.]: Addison-Wesley, 1995.
- [13] Zinky J A, Bakken D E, Schantz R. Architectural Support for Quality of Service for CORBA Objects[J]. Theory and Practice of Object Systems, 1997, 3(1): 32-39.