面向方面的实时系统中间件

李 行,张立臣,陈 成
(广东工业大学 计算机学院,广东 广州 510006)

摘 要:基于面向方面的中间件构建技术,能够把实时、安全性、可靠性、易管理性、容错性等横切关注从中间件中分离出来,形成独立于中间件的核心功能的方面,遗留中间件实现系统零侵入的实时扩展或重构。运用面向方面的编程技术可实现中间件横切关注和核心功能关注的并行设计与开发。中间件构建实现良好模块化,可配置性,部署时或运行时的可剪裁性,从而平衡实时系统中间件的通用性和专用性。

关键词:面向方面;中间件;实时系统;AspectC++;CORBA
中图分类号:TP311.52 **文献标识码**:A **文章编号**:1673-629X(2008)07-0008-03

Aspect Oriented Middleware for Real-Time System

LI Xing, ZHANG Li-chen, CHEN Cheng
(Department of Guangdong University of Technology, Guangzhou 510006, China)

Abstract: Aspect oriented middleware can separate crosscutting concerns, such as real-time, security, reliability, manageability, fault-tolerance, etc., from the core functions of middleware, and implement them independent of core functions. Then can refractor legacy middleware or extend the real-time feature of middleware without modifying any existing codes. Using aspect oriented programming the crosscutting concerns and core function concerns can be designed and developed concurrently and independently. To achieve the tradeoff of generality and specialization of middleware, aspect oriented software development provides middleware more modularity, configurability, and can tailor the architecture of middleware to fit specific needs at deployment time or at runtime.

Key words: aspect-oriented; middleware; real-time system; AspectC++; CORBA

0 引 言

中间件,比如CORBA, J2EE, .NET, Web Services已被广泛地运用以解决企业应用中的事务处理、消息传递、数据通信、安全性、性能监控等复杂的分布式计算问题[1]。而中间件在实时系统领域,实时系统对时间、可靠性、安全性、容错性等方面更苛刻,这些与非系统业务功能正交的核心系统功能在现行的中间件标准缺乏相应的描述。中间件要在通用性和专用性上平衡,传统的软件构造方法导致了系统功能代码分散到各个核心业务功能模块,与具体实现紧密耦合的实现,阻碍了中间件模块的通用性。传统中间件构建的应用程序要适应各种域的应用会增加系统运行时的开销,中间件的内部消耗则阻碍了中间件的专用性,导致在实时系统开发中,程序员常常越过中间件自行开发实时确保的非功能性约束代码。面向方面的分离关注点的思想可以自然地模块化系统的功能性需求和非功能性核心需求,在通过方面编织器将方面和类整合,整合之后的代码能根据具体应用环境自适应剪裁后编译成可执行代码。

1 背 景

中间件的开发,最近的方法,比如OpenCOM和DymamicTAO,通过引入,比如基于构件与使中间件适应不同平台或具体实例的反射技术的体系结构,这些软件工程方法来映射横切关注易造成维护性差、难扩展和客户定制难等问题,尤其当前实时中间件的明显发展趋势——硬件变得越来越小,速度越来越快,花费越来越低;相反,软件却变得速度越来慢,规模越来越大,花费越来越大等带给实时中间件系统非功能性要求更苛刻的挑战[1]:

(1)许多关键任务分布式应用,比如超大规模交换系统、作战系统(战区导弹防御系统)、工业控制过程、在线交易、电信、航空等需要实时确保。

收稿日期:2007-10-21
基金项目:国家自然科学基金资助项目(60474072,60174050)
作者简介:李 行(1981-),男,湖南郴州人,硕士研究生,研究方向为软件设计与实时系统;张立臣,博士,教授,硕士生导师,研究方向为分布式处理和实时系统等。

(2)创建实时应用程序繁琐,易出错,而且花费昂贵,经常很难确保实时中间件的开发在预算内。

(3)传统的中间件不能满足实时方面的需求。

2　面向方面的开发方法

传统的软件开发方法(面向过程,面向对象等)对现实世界－需求的建模采取垂直考虑,虽然面向对象技术较成熟,特别编程模式的运用在一定程度上缓解了逻辑混杂问题,改善了模块的模块性,降低了模块间的耦合性,增强了内聚性,但即使经验丰富,或实现了模式的标准和计量评价,模式没能实现复用——模式可以复用,而实现不可复用。传统的分离关注对于功能性关注进行了很好的抽象和建模,面向对象则在面向过程的基础上理顺了抽象与细节的关系[2]。但对非功能系统关注,或者说正交关注的抽象却缺乏描述,这实际超越了一纬抽象的考虑,要求多纬度来分解关注点。面向方面正是对这种横切关注建模,它突破类的封闭性,以一种开放编程观点来对需求进行分析。

3　中间件的构建方法

实时中间件,分布性、实时性和可靠性(见图1)是中间件体系结构的非功能上的属性[3]。从一个编程人员的观点来看,希望先能把开发的精力集中到实现功能的需求上,在这以后再适时地增加分布性、实时性和可靠性。对于模块化构件的中间件系统来说尤为如此,因为当构件被构造的时候,它应该被设计成在不同的应用环境中都是同样有用的,不管是在在具有软实时特性的系统还是在具有硬实时特性的系统中。很自然地,这需要把处理分布性、实时性和可靠性等非功能的要求和功能性的构件代码分离开来,这正是面向方面编程所要处理的问题。

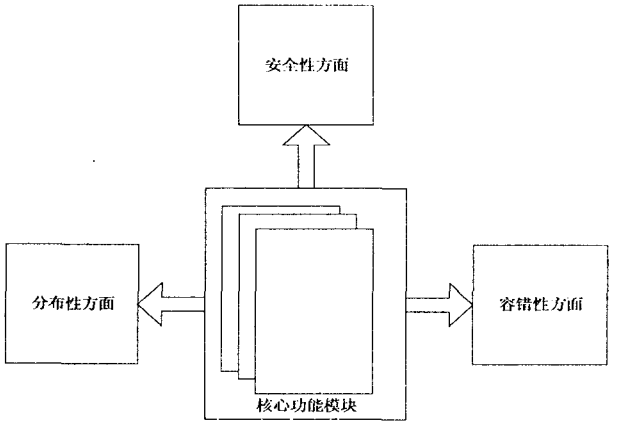

图1　中间件的有关属性

3.1　关注点分离的编织方法

用"关注点分离"的概念把中间件中所谓的构件代码和方面代码划分开来,同方面的有关的代码可以由几个不同的方面程序组成,这些不同的方面程序分别用一种面向问题的语言实现了一个确定的方面。然后一个方面编织器对构件代码和方面代码进行预处理,解释它们,找到连接点,然后把它们编织在一起以形成一个单独的实体,见图2。

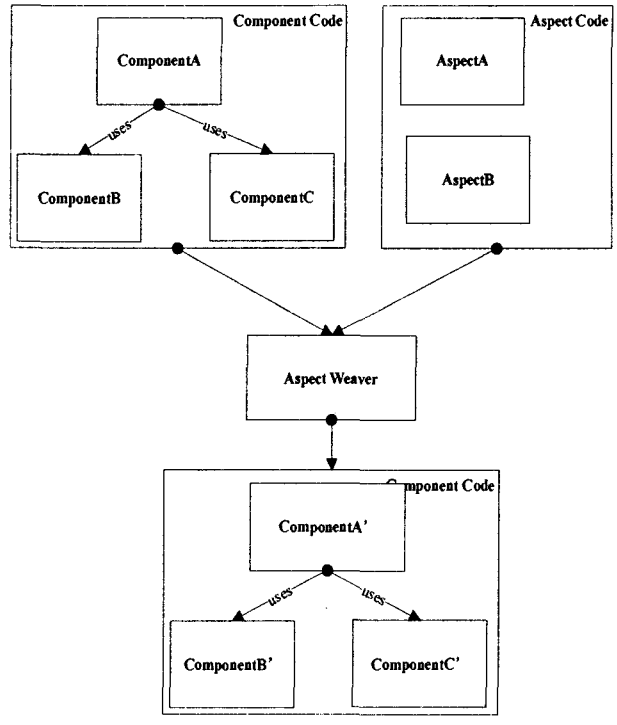

图2　编织模型

与编织的过程相符合的不同方法已经提出,它们有一个共同之处是在程序运行时所有的功能性和非功能性的程序代码部分已经被编织在一起了。即由方面代码和由功能构件代码产生的指令,根据方面程序和方面编织器定义的顺序被合成与执行。为了实现这些要求,应用运行时和编译时这两种机制都是合适的。

3.2　分布性方面

写分布式应用程序时的第一步常常是选择合适的分布式中间件[4]。和确定无误的语法相比,接口的语义常常大相径庭。比如,在一些场合下,COBRA强制使用特定的CORBA的数据类型和动态内存分配。而这在其它的中间件平台上没有必要符合这些要求。按照面向方面编程的观念,允许把构件代码和功能代码分离开来。

3.3　实时性方面

在实时的中间件系统中,各个部件不但要正确地完成任务,而且也要符合确定的时间限制要求。一般用途的构件,比如图形用户接口框架,是常常没有满足实时场合的时间约束设计。因此实时程序员常常被迫检查应用程序的每个部分,增加时间约束代码来满足

要求。建造适合实时应用程序的构件是一个相当困难的任务,因为除了功能性的要求外还要考虑非功能性的时间限制要求。这些复杂性使得设计实时构件比一般用途的构件更加费力,而且容易出错。而使用面向方面编程方法来处理功能性的构件部分与非功能性的时限行为的分离,从而确保原系统的功能描述完整性和良好的实时约束[5]。

一个监视和控制构件代码,使它在未知的和不可预知的时间界限内运行的简单方法是应用所谓的运行时间限制。线程监视器用来确保对不可靠的构件的一个请求在给定的时间界限内完成(见图 3)。这个监视器必须在构件代码执行时开始计时,以及可以在当构件成功在规定时限内完成请求时自动停止。如果在构件代码停止处理这个请求时已经超过了预定的时间,那就会调用错误处理过程(见图 4)[3]。

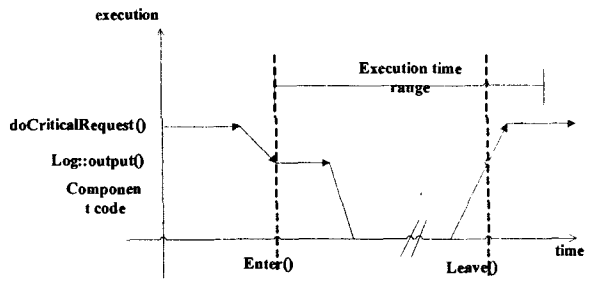

图 3 被监控的日志请求在时限内被处理

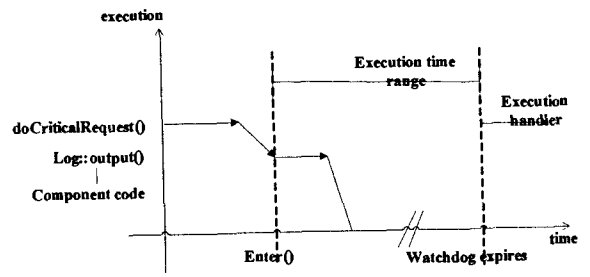

图 4 日志请求没有按时完成的异常中断处理

如果不用面向方面的编程方法,必要的监视器代码需要被人工地插入到构件代码中去。这不但会引起构件源代码的混乱,而且还会在构件不用于实时领域时增加不必要的开销,导致过剩的代码和维护难度增加、可理解性降低等问题。

如果构件只是受执行时间限制影响的话,上述的问题会在当它在应用程序中被具体的地方调用时变得更加糟糕。应用程序的具体逻辑将不得不被增加到构件代码中去,以便使得构件代码确定这个请求来自哪里和应用了什么样的时间限制。与具体应用紧密耦合的实现将极大地降低了构件的可重用性,由它们组合形成的中间件则成为不可复用的一般应用系统。

3.4 容错性方面

使用面向方面编程方法来实现一个中间件的容错性方面,使它作为分布性方面的扩展。考虑这样一个情景,当一个移动机器人被要求通过某个地区而且需要保持与监视节点之间的联系(见图 5)。

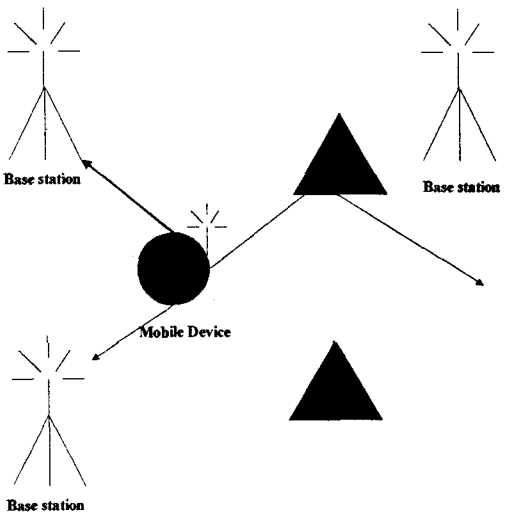

图 5 一个移动机器人和散布的基站

三个基站分布在这个区域,但环境因素可能会阻碍当移动到区域中的某些地方时与监视节点的通信。在这个情景中,控制机器人的应用程序正确工作的前提是确保日志信息到达其中至少一个监视节点。图 6 展示了对客户端用 AspectC + + 源代码编写的相应的方面程序。这个方面来自分布性方面(见图中第 4 行),作为使用在这个情景中的容错算法。它可以看作

```
1: pointcut log_method(const char * arg) =
2: executions("void Log::output(arg)");
3:
4:aspect FTLogClient dominates GuardedExecution :
5: public LogClient {
6: CORBA::Object_ptr serverLog[3];
7:
8: advice log_method(msg) : void around(const char * msg)
{
9: CORBA::Request_ptr req[3];
10: for (int i = 0; i < 3; i++)
11: req[i] = create_request(serverLog[i], msg);
12: CORBA::ORB::RequestSeq rseq(3, 3, req, CORBA_
FALSE);
13: orb->send_multiple_requests_deferred(rseq);
14: CORBA::Request_ptr rreq;
15: orbp->get_next_response(rreq); // wait for first re-
sponse
16: CORBA::release(rreq);
17: }
18: };
```

图 6 容错日志的方面代码

(下转第 14 页)

入进行解析,并取出数据,对数据进行分析并将最后结果输出到Scope Window,同时会将一些状态信息输出到Browser Window上。

2.3 Everest体系架构设计的特点

该体系结构是在研究其他浏览器技术及和欣平台的基础上进行设计的,考虑了嵌入式应用的需求,也结合了和欣作为网络操作系统的特点。满足了2.2.1中提出的设计原则中的需求。其主要特点有:

(1)良好的模块化特性。

作为嵌入式浏览器,选择了基本的功能模块作为体系结构的基础,而且在实现中,这些基本的功能模块满足了应用的要求。同时对这些功能模块进行了很好的模块化。

(2)良好的可扩展化特性。

它具有良好的扩展性,一方面通过抽象出文档对象树,使得浏览器可以方便地扩展为DOM模型,进一步支持JavaScript;另一方面,模块之间采用消息机制来进行通讯,有效降低了模块之间的耦合,并且这种松耦合状态可以很方便地添加对新的数据类型的支持;同时,通过扩展机制,有效地把CAR构件融合到浏览器中,使得CAR构件可以方便地在浏览器中支持。

3 结束语

提出了一个适合于Elastos的嵌入式浏览器框架,该框架具有简单、模块化好、模块耦合性小、易扩展等特点。该框架在包含基本功能模块的基础上,参考了DOM的机制,对页面数据进行对象化,抽象出文档对象树模块。以对象化的方式对页面管理,使得对页面的管理更加方便,逻辑性更好,同时有利于提供对DOM的支持。该框架提供了扩展机制,可以很好地支持CAR构件等第三方控件。

参考文献:

[1] Koretide. CAR's Manual[M]. 陈 榕译. 上海:上海科泰世纪科技有限公司,2004.

[2] 徐会建. 嵌入式浏览器DeltaBrowser表示层的设计与实现[D]. 成都:电子科技大学,2004.

[3] 王光磊,张 跃,杜伟力. 基于虚拟客户/服务器的清华嵌入式浏览器THEWEB的设计与实现[J]. 计算机应用与研究,2002(4):96-99.

[4] 陈 榕. 和欣资料库[EB/OL]. 2002. HTTP://www.koretide.com.cn.

[5] 朱剑民,陈 榕,倪光南. "和欣"操作系统的浏览器设计模型[J]. 计算机工程与应用,2003,39(13):13-15.

[6] 周正勇,阳富民,胡贯荣. 一种嵌入式浏览器的核心技术及特色[J]. 计算机工程与设计,2003,24(3):21-23.

[7] 陈 榕,刘艺平. 基于构件、中间件的因特网操作系统及跨操作系统的构件、中间件运行平台[R]. 上海:上海科泰世纪科技有限公司,2003.

(上接第10页)

是前面讨论过的分布性方面的扩展。第7到14行的方面代码使用CORBA并行地传递多个请求。第13行的代码是在等待至少有一个服务器产生响应。

为了使得上述的容错性方面代码和执行时间监视器的方面代码一起执行,AspectC++需要被通知首先执行有关时间监视器的方面代码。可以用dominates语句来完成该功能,当在构件代码中同时使用两个方面程序时,产生的编织代码将尝试联系所有的基站和传送信息给基站。如果没有基站响应,客户端的执行时间仍然是被约束的,当预定的时间用完了的时候,程序就会引发一个错误中断。

4 结 语

实时系统中间件存在着分布性、实时性、容错性、安全性、可靠性、易管理性这些非功能的要求,是产生问题的重要原因。面向方面是一种强大的分离和简化设计关注点的技术,它允许从中间件的核心功能中分离出非功能方面;中间件不需要一开始就包含这些方面的特性,因为它们可以在之后通过方面编织器来增加这些缺少的特性,而不需要采取易出错的人工硬编码方法来编辑构件代码。这增加了中间件的设计灵活性、可理解性、模块化、可扩展性、可配置性、重用性,减少了中间件开发风险和维护负担,符合重构思想,有利于中间件的进一步演化。

参考文献:

[1] Zhang C, Hans-Arno J. Refactoring Middleware with AspectJ[J]. IEEE transactions on parallel and distributed systems, 2003,14(11):1-16.

[2] 刘瑞成,张立臣. 实时系统的面向方面模型[J]. 计算机工程与设计,2006,27(6):937-940.

[3] 黄宇刚. 面向方面的实时系统中间件[D]. 广州:广东工业大学计算机学院,2006.

[4] Zhang C, Hans-Arno J. Quantifying Aspects in Middleware Platforms[C]//AOSD 2003. Boston, MA: USA: [s. n.], 2003.

[5] 刘瑞成,张立臣. 基于面向方面的实时系统建模方法[J]. 计算机科学,2006,33(7):262-265.