

文章编号: 0258-2724(2010)04-0603-06 DOI: 10.3969/j.issn.0258-2724.2010.04.020

基于 SOA 的协同设计网络性能优化

蔡海尼^{1,2}, 文俊浩^{1,2}, 姬楷¹

(1. 重庆大学软件学院, 重庆 400044; 2. 重庆大学计算机学院, 重庆 400044)

摘要: 为了优化协同设计网络的服务性能,引入服务容器分层注册机制和基于服务质量(quality of service, QoS)的动态消息路由访问策略,提出了面向服务体系结构中的企业服务总线改进模型.该模型通过构建服务容器间的树形结构关系,用深度优先搜索法实现服务查找,将查找范围限定在以服务请求容器和服务注册容器为叶节点的最小子树,增强了局部注册访问性能,减少了中心总线的请求负荷和网络开销,并在 QoS 加权算法中综合当前测试数据和历史数据进行服务评估,以获取高质量的动态服务.应用实例表明:当并发访问量较大时,改进模型的平均响应时间比现有全局注册模式减少 30% 以上,并且在网络服务质量变差时能自动调整路由恢复到良好状态,提升了协同设计系统的效率.

关键词: 协同设计;企业服务总线;分层注册模式;动态消息路由

中图分类号: TP311.5 **文献标识码:** A

Performance Optimization of Collaborative Design Network Based on Service Oriented Architecture

CAI Haini^{1,2}, WEN Junhao^{1,2}, JI Kai¹

(1. College of Software Engineering, Chongqing University, Chongqing 400030, China; 2. College of Computer Science, Chongqing University, Chongqing 400030, China)

Abstract: In order to optimize the service performance of collaborative design network, an improved enterprise service bus (ESB) model with a layered service registry and QoS-based dynamic message routing was proposed in service-oriented architecture. By setting up a tree-like hierarchy among the service containers, this model uses a depth-first search to implement lookup service and narrow the search area to a minimal subtree with the service request container and service registry container as its leaves. This enhances the efficiency of local registry access to reduce the request load and network overhead of the central bus. Besides, a weighted quality of service (QoS) algorithm is adopted in the model for analysis of the current and historical measurement data, so that high-quality dynamic services are achieved. The results of application to the collaborative design system in a survey and design enterprise show that the improved ESB model reduces the average response time of the current global service registry by more than 30% under heavy access concurrency, and can automatically adjust the message route to regain a good working condition when the quality of network services is getting worse. Therefore, the collaboration of the collaborative design system is improved efficiently.

Key words: collaborative design; enterprise service bus; layered service registry; dynamic message routing

收稿日期: 2009-07-24

基金项目: 国家自然科学基金资助项目(70672011); 国家科技支撑计划项目(2007BAF23B0302)

作者简介: 蔡海尼(1972-), 女, 讲师, 博士研究生, 研究方向为服务计算与面向服务的软件工程, E-mail: hainic@tom.com

面对企业日益增长的信息共享和业务整合需求,实现基于面向服务架构(service-oriented architecture, SOA)的企业应用集成是软件开发的发展趋势. SOA 将系统按照业务需求定义为松散耦合的粗粒度应用组件^[1],作为共享资源,提供以服务为中心的设计方法.

作为支撑 SOA 的消息框架,企业服务总线(enterprise service bus, ESB)的研究受到广泛关注. 文献[2]从消息机制、消息转换、消息路由、服务容器 4 个方面分析了 ESB 核心功能组件及目前存在的主要问题. 文献[3-4]分别提出了基于公共基础组件的软总线模式和基于 ServiceMix 的 SOA 架构的简易实现方案. 多服务发布必然使访问量加大,产生通信拥塞和服务质量下降,而多层结构层次间的协议转换也会带来额外的系统开销,因此,现有企业服务总线注册访问模式存在性能瓶颈.

针对上述问题,文献[5]通过动态组件模型改进了发布/订阅模式,以减少冗余,保证目录的简单性,但未给出具体实现方法. 文献[6-7]分别对动态可配置的 ESB 服务路由框架和 ESB 中的可靠动态服务路由进行了研究,但其中的 QoS (quality of service) 算法基于单次测试数据评估服务,容易出现偏差. 本文提出了 ESB 改进模型,设计并实现了服务容器分层注册模式和基于 QoS 的动态消息路由,更好地解决了基于 SOA 的勘察设计企业应用软件架构系统的性能优化问题.

1 模式分析

1.1 现有注册模式

目前的 ESB 产品主要提供服务总线全局注册与服务容器本地注册两种服务注册方式. 服务总线全局注册是指服务容器本身不管理注册信息,每次服务请求都要通过查询总线全局注册表获取服务信息.

ServiceMix 是一个实现 JBI (Java business integration) 规范的开源 ESB 和 SOA 工具包,其开源特性有利于深入研究 ESB 的内部机制和系统架构. 同时,ServiceMix 支持 ESB 核心功能,可扩展性好,文档说明完备,适应勘察设计企业在发展阶段、业务需求、IT 背景等方面的特点,因此,选用 ServiceMix 作为研发平台.

JBI 规范定义了一个中间消息交换模式的架构,组件接口和交互语义采用 WSDL 标准进行描述,组件之间的消息通过标准化消息路由

(normalized message router, NMR) 进行中转,彼此不直接通信.

JBI 环境中组件间的交互过程为:首先由服务使用者产生一个服务请求消息,并将其转换为标准化消息,发送给 NMR, NMR 根据消息中的服务名或接口名进行目的组件查找,并最终将消息转发至目的组件. ServiceMix 中使用的就是全局注册方式,基于服务总线的应用由多个组件构成,组件之间通过消息进行交互,其结构如图 1 所示^[8].

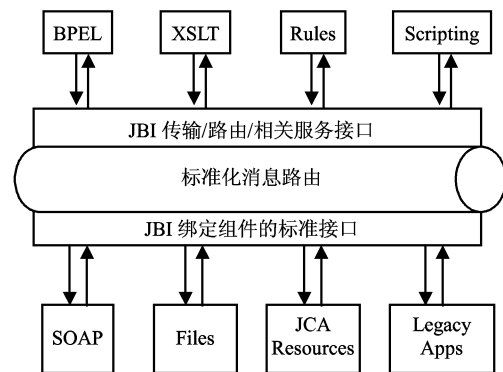


图 1 ServiceMix 结构

Fig. 1 Architecture of ServiceMix

全局注册方式实现了组件的位置透明和协议透明,借助在 ServiceMix 上部署不同的绑定组件,可支持具有不同接口和使用不同通信协议(如 JMS, SOAP 等)系统与 ESB 进行通信,但存在如下局限性:

(1) 所有 JBI 组件只能在一个 Java 虚拟机环境中运行,应用集成的负载集中在计算节点,形成整个系统的性能瓶颈.

(2) 负责应用集成的服务可能造成单点故障,当该节点出现故障,整个系统将处于瘫痪状态. 总线全局模式中,可能出现单点故障问题,即如果中心总线故障,则整个系统将无法再执行服务注册和访问,陷入瘫痪状态.

(3) JBI 规范不支持消息传输的持久性,一旦 JBI 节点出现故障,所有内存中的消息将被丢弃,即使 JBI 环境重新启动,也无法获得丢失的消息^[9].

服务容器本地注册是指服务容器维护本地注册表,同时服务总线维护中心服务目录. 服务容器接收到请求时,将从本地注册表中检索服务提供者. 当中心服务目录或者本地服务发生变化时,需要实现与中心目录的通讯,以保证两个目录的一

致性^[2].

本地注册模式中,每个服务容器都与中心总线维护的服务列表一致,相对安全可靠,但是每次注册变更都需要更新中心总线目录.再由中心总线将该变更信息广播到所有其它服务容器,并更新容器目录.因此,当中心服务目录或本地服务发生变化时,中心目录与本地目录频繁地同步更新,可能导致过多的系统网络资源耗费.

1.2 服务容器分层注册

针对上述两种方式的局限性,分层注册模式在中心总线与各本地服务容器之间建立层次关系,每个容器采用树形结构存储本地服务的注册表.当需要查找服务时,首先看本地服务容器的服务注册列表中是否含有相应服务,若能满足,则本地服务容器自行处理该请求;否则,将请求提交到父容器,查找父目录^[10].因此,能降低冗余,保证目录的简单性.在层次模式下,所有服务使用者和提供者都封装在服务端点中,请求消息发送方向始终向上,若本地容器注册了新服务,则逐级向上更新,直至中心总线.

分层注册的方法具有以下优势:

(1) 减少中心服务器请求负荷.只有当本地服务器所在路径中的所有服务容器注册列表都无法找到对应请求服务,并将请求转发到上层容器时,中心服务容器才收到该请求.

(2) 相对可靠.总线全局模式中,可能出现单点故障问题,而分层模式不依靠中心总线,各容器都维护本地注册列表,中心总线出现故障时,各本地容器仍可以提供部分服务.

(3) 占用网络资源相对较少.在分层模式下,当变更服务注册时,只需更新本地服务容器到中心总线路径上的服务容器节点的服务信息,中心总线接收该注册服务信息,但并不将该信息广播到其他服务容器.

1.3 基于QoS的动态消息路由

在一个ESB环境中可能存在多个服务符合某个请求的接口规范,使用者需根据各种逻辑或外界因素从中选择该请求的路由目的地.在ESB中进行服务提供者选择的过程称为消息路由,是ESB的核心功能之一.动态消息路由指服务提供者的选择是在服务调用时才确定,而不是预先设定^[7].

由于各勘察设计系统在功能上存在重复性,对应同类服务请求通常会有多个可供选择的目標服务在同时运行,使用者需根据QoS在这些服务中

选择一个最优者,作为该请求的路由目的地,实现如图2所示的基于QoS的动态消息路由,以获得快速响应和高质量的服务^[11].

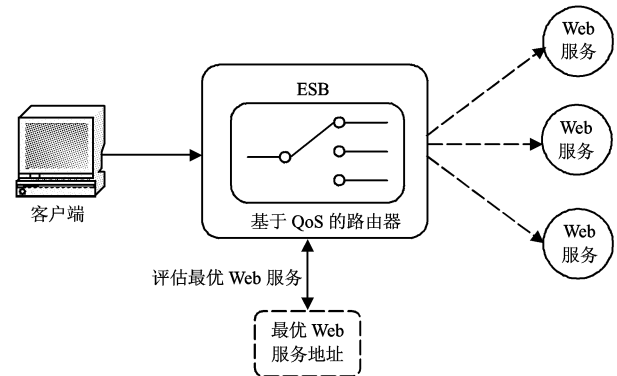


图2 基于QoS的动态消息路由

Fig.2 Dynamic message routing based on QoS

2 改进模型

2.1 分层注册模式设计

用分层注册模式建立容器之间的层次关系,形成一个树形拓扑结构.树形结构中的根节点是中心总线,叶节点是本地容器,中间节点是叶节点的上层容器.服务查找采用深度优先搜索,将查找限定在尽可能小的范围内.当本地容器接收到服务请求时,先在本地容器的服务注册列表内搜索,若找到相应服务则查找成功;否则,将此服务请求发送至上层容器,即将查找范围扩展到以父容器为根节点的子树,如此依次向上转发请求,逐步扩大查找范围,直到查找成功.

分层注册模式的工作机制如图3所示.服务使用者发送一个服务请求到本地服务容器,JB1绑定组件接收该请求,并将请求传递给本地标准化消息路由,如果消息路由在所维护服务列表中找到服务,则提供给使用者;否则,调用转发服务,本地消息路由利用服务引擎发送请求到上层容器,直至找到适合的服务为止.

2.2 基于QoS的动态消息路由设计

实现基于QoS的动态消息路由,即基于QoS的服务动态绑定,关键在于程序运行时根据各个服务提供者的QoS动态更新路由表中的路由信息,选择消息发送的目的地(服务提供者)^[12].

目前的QoS算法大多是基于权重的加权算法^[13],但仅依据QoS参数某一次的测试数据评估服务容易出现偏差,因此,通过综合各历史测试数据的方法对传统评估算法加以改进,求得更为可靠

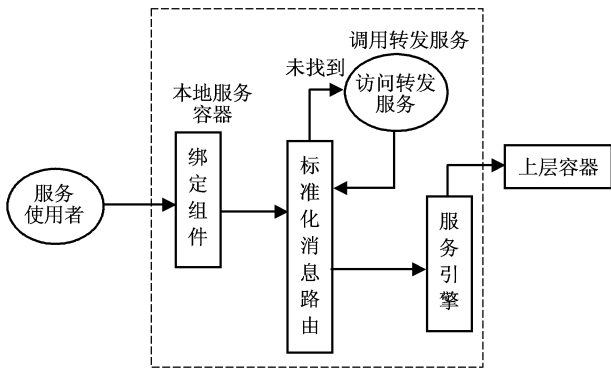


图3 分层注册机制

Fig. 3 Working mechanism of layered service registry

和全面的最优服务. 例如, 服务响应时间可能不断变化, 可分别为其当前值和历史数据设定权重, 求出其历史均值, 并根据各参数的权重值, 采用加权平均法, 最终得到该服务的 QoS 得分. 同时, 考虑到算法中的归一化过程可能对个别 QoS 参数过于夸大或缩小, 导致结果出现偏差, 可以针对参数设定期望值, 用判定是否达到期望值的方式替代其中的归一化过程.

设计如图 4 所示的框架实现动态路由, 在 ESB 和其相应服务组件上添加服务评估、服务测试和服务发现模块, 并设置动态消息路由库和存放历史测试数据的容器, 其中动态消息路由库为存放可用路由信息的 XML 文件^[14]. 在系统运行过程中不断发现新的服务并存入路由库, 从路由库中取出服务地址测试各服务, 并将测试结果存为历史测试数据. 用历史数据评估服务, 选取其中最优服务提供者更新当前路由表中的路由, 达到动态优化路由的目的.

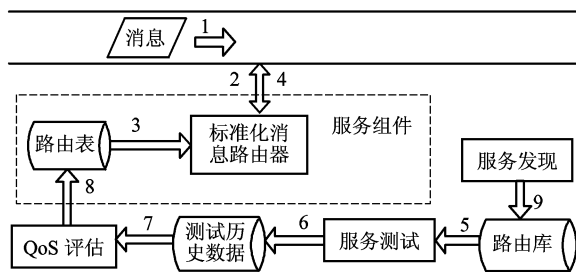


图4 动态路由实现框架

Fig. 4 Framework of dynamic routing

3 应用实例

目前勘察设计企业各部门和工作环节之间存在许多信息孤岛, 各种信息资源无法有效共享^[15]. 以多方参与设计项目的协同管理为主线, 基于

SOA 技术研发的勘察设计企业协同设计与管理系
统, 旨在促进企业信息资源的有效利用, 提升勘察
设计企业的信息化管理水平.

3.1 协同设计系统体系结构

勘察设计企业协同设计系统的体系结构模型
如图 5 所示.

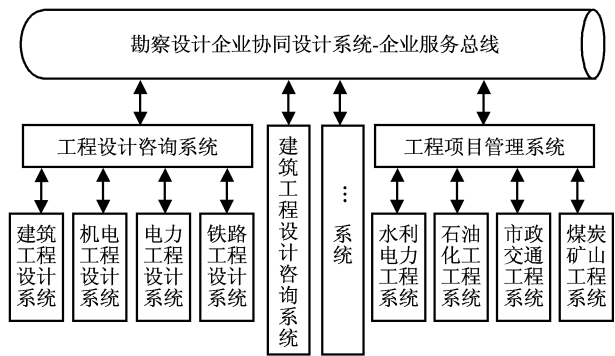


图5 协同设计系统体系结构

Fig. 5 Architecture of the collaborative design system

勘察设计企业协同设计系统的中心服务总线
由 ServiceMix 承担, 各子系统的实现机制基本相
同. 例如: 工程设计咨询端点对工程设计咨询系
统中的应用和组件进行封装、以服务容器的形式与
中心总线相连. 该服务端点承担两个角色:

(1) 作为服务提供者为其他端点提供服务. 工
程设计咨询系统注册了咨询合同查询服务, 若此时
建筑工程设计系统需要查询这些合同信息, 就可以
向工程咨询端点发出请求消息, 经确认后即可使用
该服务.

(2) 作为服务使用者, 工程设计咨询端点也
可以发送请求到自身或者更高的端点. 例如: 工程
设计咨询系统需要得到水利电力工程系统提供的
公开数据, 工程设计咨询端点首先发送请求到本地
服务器, 如本地服务注册列表中没有该项服务, 该
端点将请求发送给中心总线, 中心总线发现水利
电力工程端点可以提供服务, 就转发给水利电力
工程端点, 让其为工程设计咨询系统提供服务.

3.2 系统实现

分层注册模式实现的要点是转发服务, 其实现
过程如下: 搭建 ServiceMix 运行环境, 用 Netbeans
创建“转发服务”. 转发服务接收到标准化消息路
由发送的调用消息, 以自身请求服务的方式向消息
路由发送传递至上层容器的请求, 消息路由接收该
消息, 并通过服务引擎将消息发送给上层容器.

在 ServiceMix 中, 可以通过简单 Java 对象
(plain ordinary Java objects, POJO) 很方便地操作消

息并将其发往目标服务,所以,系统用 POJO 形式的路由处理器完成消息的路由处理^[16].路由处理器的具体工作过程如下:当路由处理器收到一个消息时,若测试时间未到且原路由服务质量没有下降,则向原路由终点发送消息;若测试时间已到,或者原路由服务质量已经下降,则开始新一轮的服务测试和评估过程,测试候选服务的服务质量,取历史测试数据,评估候选服务,取最优服务为路由终点,向路由终点发送消息.

3.3 系统应用测评

系统测评的目的是为了测试基于 SOA 的勘察设计企业协同设计系统的运行能力,以便完善后用于实际业务系统^[17].测试的网络环境为局域网和教育网,表 1 为测试的软件和硬件配置.

表 1 软件和硬件配置表
Tab. 1 Software/hardware configuration for testing

类型	软件	CPU	内存
Web 服务器	Tomcat 5.5	赛扬 4 2.0 G	1.25G
Web 服务框架	CXF		
数据库	SQL Server 2000	Intel T5500	1.66G
操作系统	Windows XP		2G

并发访问压力测试的目的是:评估协同设计系统在大量用户并发访问时的整体性能,并对比服务总线全局注册和服务容器分层注册两种模式的系统性能,以确保其集成到应用系统的效率.根据勘察设计企业的业务规律,假设本地容器自行处理的服务请求与需要转发到上层容器的服务请求各占 50%,其中需转发到中心总线的占 20%.测试用例创建多个线程,同时调用相应的 Web 服务,考察两种注册模式下并发服务请求数与平均响应时间的关系.测试结果如图 6 所示.

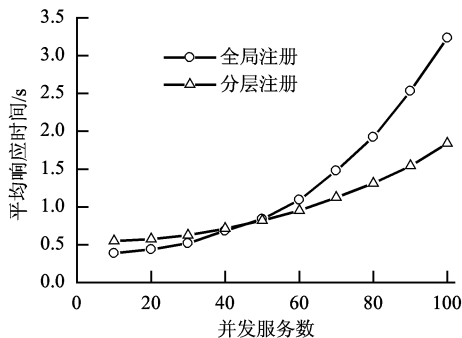


图 6 注册模式性能比较
Fig. 6 Perfance comparison of the registry modes

由图 6 可见,随着并发访问量的增加,两种注

册模式的平均响应时间都呈上升趋势.当并发服务数较少(50 以下)时,两者的响应时间均小于 1 s,其中分层注册模式因向上层容器及总线转发部分服务产生额外时间开销,性能略逊于全局注册模式;但当并发量增大到一定程度(50 或以上)时,分层注册模式响应曲线的上升趋势明显比全局注册模式缓和,表明在大并发访问量情形下,即使存在转发服务的时间开销,分层注册模式通过均衡请求负载,显著地改进了系统性能.

动态消息路由的测试过程如下:客户端在第 1~80 s 之间每隔 5 s 就分别以动态路由和固定路由的方式调用应用服务器上的 Web 服务,同时,在第 35 s 时人为地在应用服务器之间进行大型文件拷贝.两种路由方式在不同时刻的时间耗费如图 7 所示.

开始时因为都是请求同一应用服务器的 Web 服务,动态路由和固定路由所耗费的时间差别不大.到了第 35 s 时,由于应用服务器网络质量变差,两种路由所耗费的时间都急剧增加,之后两种方式所耗费的时间开始出现差别,固定路由耗费的时间变化不大;动态路由在第 40 s 时耗费了特别多的时间,但之后耗费时间恢复到最初的良好水平.上述结果表明,基于 QoS 的动态路由提高了服务质量.

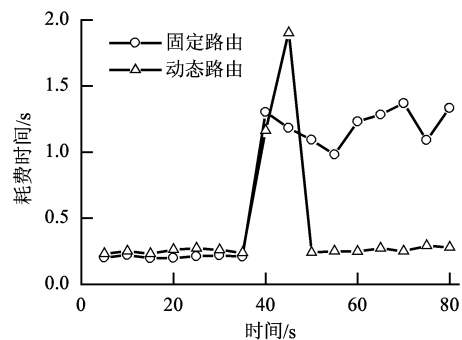


图 7 消息路由时间耗费对比
Fig. 7 Time cost comparison of the message routings

4 结 论

本文提出了 ESB 改进模型,引入了服务容器分层注册机制和基于 QoS 的动态消息路由访问策略,减少了中心总线的请求负荷和网络开销,增强了局部服务器的注册访问性能,并能动态地获取高质量的服务.系统应用测评表明:该模型可有效改善勘察设计企业 ESB 的协同效果.随着企业应用集成的深入,上述 ESB 框架还应从以下两方面进

一步完善:

(1) 当企业环境中产生新的服务时,实现服务自动注册;

(2) 建立事件驱动的更新机制,即一旦有服务更新,消息路由就相互通信,实现自动更新.

参考文献:

- [1] CURL A, FERTALJ K. A review of enterprise IT integration methods [C] // Proc. of the ITI 2009 31st International Conference on Information Technology Interfaces. Zagreb: University of Zagreb, 2009: 107-112.
- [2] 谢继晖,白晓颖,陈斌,等. 企业服务总线研究综述 [J]. 计算机科学, 2007(11): 13-18.
XIE Jihui, BAI Xiaoying, CHEN Bin, et al. A survey on enterprise service bus [J]. Computer Science, 2007 (11): 13-18.
- [3] 王淑营. 面向制造业产业链的协同商务平台集成框架 [J]. 西南交通大学学报, 2008, 43(5): 643-647.
WANG Shuying. Integrated framework of collaborative commercial platform for manufacturing industrial chain [J]. Journal of Southwest Jiaotong University, 2008, 43(5): 643-647.
- [4] 彭政,聂瑞华,李飞,等. 基于 ServiceMix 的 SOA 架构的研究与实现 [J]. 计算机工程与科学, 2009, 31(4): 153-158.
PENG Zheng, NIE Ruihua, LI Fei, et al. Research and implementation of SOA based on ServiceMix [J]. Computer Engineering & Science, 2009, 31(4): 153-158.
- [5] LOUREIRO E, BUBLITZ F, BARBOSA N, et al. A flexible middleware for service provision over heterogeneous pervasive networks [C] // Proc. of the 2006 International Symposium on a World of Wireless, Mobile and Multimedia Networks. Piscataway: IEEE Press, 2006: 609-614.
- [6] BAI Xiaoying. DRESR: dynamic routing in enterprise service bus [J]. e-Business Engineering, IEEE, 2007 (10): 528-531.
- [7] WU Bin, LIU Shijun, WU Lei. Dynamic reliable service routing in enterprise service bus [C] // Proc. of 3rd IEEE Asia-Pacific Services Computing Conference. Piscataway: IEEE Press, 2008: 349-354.
- [8] JIANG Jichen, GAO Ming. Enterprise service bus and an open source implementation [C] // Proc. of 2006 International Conference on Management Science and Engineering. Piscataway: IEEE Press, 2006: 926-930.
- [9] FU Ning, ZHOU Xingshe, WANG Kaibo, et al. Distributed enterprise service bus based on JBI [C] // Proc. of 3rd International Conference on Grid and Pervasive Computing Symposia/Workshops. Piscataway: IEEE Press, 2008: 292-297.
- [10] WEN Rui, MA Yaping, CHEN Xiaoqing. ESB infrastructure's autonomous mechanism of SOA [C] // 2009 International Symposium on Intelligent Ubiquitous Computing and Education. Piscataway: IEEE Press, 2009: 13-17.
- [11] ZENG Liangzhao, BENATALLAH B. QoS-aware middleware for web services composition [J]. IEEE Transactions on Software Engineering, 2004, 30(5): 311-327.
- [12] RAN Shuping. A model for web services discovery with QoS [J]. ACM SIGecom Exchanges, 2003, 4(1): 1-10.
- [13] 张志民,陈昊鹏. 一种综合评估 QoS 的算法研究 [J]. 计算机应用与软件, 2008, 25(12): 256-259.
ZHANG Zhimin, CHEN Haopeng. Research on an algorithm of QoS combination [J]. Computer Application and software, 2008, 25(12): 256-259.
- [14] NQUYEN X T, KOWALCZYK R, PHAN C A. Dynamic asynchronous aggregate search for solving QoS compositions of Web services [C] // Proc. of 2006 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology. Piscataway: IEEE Press, 2006: 142-145.
- [15] 王静,陈岱林,梁晓,等. 勘察设计企业信息化实施方案探讨 [J]. 施工技术, 2005(2): 16-18.
WANG Jing, CHEN Dailin, LIANG Xiao, et al. Discussion on application plans of the survey and design enterprise information building [J]. Construction Technology, 2005(2): 16-18.
- [16] VINOSKI S. Java business integration [J]. IEEE Internet Computing, 2005(9): 89-91.
- [17] LEWIS G A. Common misconceptions about service-oriented architecture [C] // Proc. of ICCBSS 2007: Sixth International IEEE Conference on Commercial-off-the-Shelf (COTS)-Based Software Systems. Piscataway: IEEE Press, 2007: 123-130.