面向服务环境中的NServiceBus服务总线应用研究

唐蓉君[1] 叶 波[2] 文俊浩[2]
(重庆大学信息与网络管理中心 重庆 400030)[1] (重庆大学软件学院 重庆 400044)[2]

摘 要 传统的面向服务环境中,服务请求者和服务提供者需要通过UDDI进行服务绑定,但其无法提供一种异步、可靠、可管理的过程。文中引入NServiceBus开源服务总线来提供一种异步通信模型。对注册服务进行集中管理和服务质量监控可有效提升系统服务调用的灵活性与可扩展性。除此之外,通过分析NServiceBus开源服务总线的消息转发机制,提出一种服务响应时间模型来预测系统在高并发访问下的负载情况,并对其进行过载保护,设计并实现了一种自适应优先级队列来对服务请求进行分级调度,以在一定程度上保证服务响应时间,提高系统的响应速率。最后经过实际项目中的应用测评实验对比分析,验证了本方法对提升NServiceBus服务性能的有效性和实用性。

关键词 SOA,Web服务,NServiceBus,过载保护,消息队列
中图法分类号 TP31 **文献标识码** A

Research and Application of NServiceBus Service Bus in SOA Environment

TANG Rong-jun[1] YE Bo[2] WEN Jun-hao[2]
(Center of Information and Network, Chongqing University, Chongqing 400030, China)[1]
(College of Software Engineering, Chongqing University, Chongqing 400044, China)[2]

Abstract In the traditional service-oriented environment, service customers and service providers need to complete service binding through the UDDI server which cannot provide a asynchronous, reliable and manageable process. This paper introduced NServiceBus which is a open source service bus to provide a kind of asynchronous communication model. The centralized management and service quality control for registration services can effectively improve the system service calls' flexibility and extendability. In addition, this paper analyzed message forwarding mechanism of NServiceBus open source service bus, and proposed a kind of service response time model to forecast overload condition of current system in high concurrent access and provide overload protection. This paper also designed and implemented a kind of self-adaptive priority queuing to provide hierarchical scheduling of service request, to a certain degree to ensure service response time and improve the system response rate. Finally the evaluation experiment and contrast analysis in the application of actual project prove validity and practicability of this method to improve the performance of NServiceBus service.

Keywords SOA, Web services, NServiceBus, Overload protection, Message queue

1 引言

当前,信息化技术的作用日益突出,企业的市场竞争力完全取决于对数字信息的获取、处理和整合能力。如何结合自身的特点和需要,利用软件体系结构的设计方法整合企业现有系统资源,全面改进企业的业务经营管理平台,提升信息系统的工作效率,成为信息市场战中面临的迫切问题。

值得注意的是,目前企业级应用系统通常只关注某一个单一功能或者业务流程,各子系统彼此独立运行,要在这些应用程序之间交互通信、共享数据是非常困难的。信息孤岛、平台异构、集成难度大以及功能扩展性差成为了解决企业级应用整合的关键问题。SOA的软件架构思想和设计方法取代了传统设计思路,所建设的系统具有跨平台、跨数据库、跨应用服务器的特性,能够为企业级应用资源的整合提供解决方案,实现企业信息资源的再次利用。与此同时,SOA中的服务注册管理、消息路由和服务质量保证等技术也成为了研究热点。

针对以上问题,本文通过在原有SOA架构体系的服务请求者和服务使用者之间引入NServiceBus开源服务总线来提供一种异步通信模型,以对注册服务进行集中管理和服务质量监控,有效提升系统服务调用的灵活性与可扩展性。除此之外,本文通过分析NServiceBus开源服务总线的消息转发机制,提出一种服务响应时间模型来预测系统在高并发访问情况下的负载情况,并对其进行过载保护,引入自适应优先级队列来对服务请求进行分级调度,使其能够在一定程度上保证服务响应时间,提高系统的响应速率。

2 相关技术

SOA是一种重要的构建分布式系统的组件模型,是设计和构建松耦合软件解决方案的方法,它以软件服务的形式公

到稿日期:2012-09-22 返修日期:2012-12-23 本文受重庆市自然科学基金(CSTC,2008BB2191)资助。
唐蓉君(1973-),男,博士生,高级工程师,主要研究方向为信息系统建设,E-mail:rjtang@cqu.edu.cn;叶 波(1988-),男,硕士生,主要研究方向为服务计算与面向服务的软件工程;文俊浩(1969-),男,博士,教授,主要研究方向为服务计算与面向服务的软件工程。

开业务功能，可以针对粗粒度服务进行分布式部署、组合和使用，使得其它应用程序可以通过已发布接口来使用。下面首先探讨 SOA 架构中的相关技术和原理。

2.1 Web 服务体系的理论模型

Web Services 体系结构是面向对象分析与设计的发展结果，其基于 SOA 实现一种分布式计算机制，应用程序都被封装为服务并可以通过网络调用。

Web Services 体系结构的基本原理是基于 SOA 和 Internet 协议的。它代表基于标准和采用这些标准技术的可集成应用程序解决方案，确保 Web 服务应用程序的实现方案与标准规范相兼容，从而实现与兼容应用程序之间的交互操作。Web Services 体系结构的关键设计要求如下：

(1)提供一致的解决方案模型，将应用程序定义为模块化组件，使其成为可发布的服务。使用基于标准的结构模型和协议来定义框架，以便在 Internet 上支持基于服务的应用程序。

(2)可使分布式应用程序成为集中式和分散式应用程序环境，可支持企业内部和企业之间应用程序的无边界通信。

(3)可将服务发布到一个或者多个公共目录，从而使用户能够使用标准机制找到已经发布的服务。

2.2 基于消息路由的服务通信模型

基于消息路由的服务通信模型规定了一种松散、事件驱动的通信方式。图 1 中服务使用者在调用该模型下的服务时，可以通过同步或者异步的方式进行服务调用。对于消息的同步传输机制，服务调用者线程需要同步等待服务结果的响应；通过消息的异步传输机制，服务调用者则不再需要等待服务提供者的及时响应。服务使用者同时也可以接受消息路由机制下转发的 Web 服务消息文档。

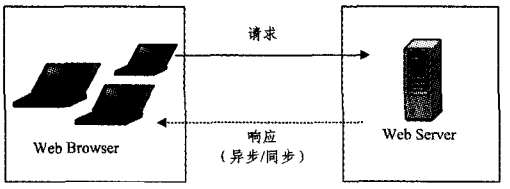

图 1 基于消息路由的通信模型

2.3 分布式服务总线

SOA 体系结构中的服务总线是一种软件体系结构设计模式，可以使不同的系统在企业内或企业间组装为联合总线，它是企业构建基于面向服务体系结构解决方案时所使用的基础架构的关键部分。服务总线是指由中间件基础设施产品技术实现的，通过事件驱动和基于 XML 消息引擎为更复杂的面向服务的架构提供软件架构的构造物。

从本质上分析，SOA 中的服务总线提供可靠消息传输、服务接入、协议转换、数据格式转换和基于内容的路由等功能，屏蔽了服务的物理位置、协议和数据格式，并利用 Web 服务标准结合消息中间件来作为 SOA 中互连的协议接口。SOA 中的服务总线有如下特性：支持消息传递(同步、异步、点对点、发布/订阅)、基于内容的路由服务、支持队列、连接异构的系统以及在系统传输层上传送 SOAP 数据流。

从功能上分析，SOA 中的服务总线改变了传统的软件架构，并在以往的架构基础上支持事件驱动和消息路由机制。复杂数据也能够在总线上进行序列化后传输，通过服务总线来实现服务调用的集中管理、监控，其提供的接口可以用来桥接服务请求者和服务提供者。服务总线在 SOA 的实践中是一种实现服务端服务集成的解决方案，提供了服务调用过程中的消息传输格式统一定义、协议集中转换和消息路由机制等功能。

3 NServiceBus 开源服务总线的研究

NServiceBus 是一个. NET 架构下的轻量级开源服务总线，它在消息发布/订阅支持、工作流集成和高度可扩展性等方面表现优异，为 SOA 体系结构提供了一个异步可靠的消息通信框架。

3.1 NServiceBus 的消息转发机制

3.1.1 异步通信模型

NServiceBus 框架能够配合 MSMQ 组件进行消息通信，在消息队列通信模型中最主要的参与者就是发送方、消息队列和接收方，而这些发送方和接收方可以处于同一台机器上，也可以分布在网络中的不同端系统中。NServiceBus 中消息转发机制的通信原理如图 2 所示，分为以下 3 个步骤。

(1)服务使用者将服务请求消息发送到一个消息待处理容器，等待接收方接收处理。

(2)待处理的服务请求消息被转发到系统共用消息队列(Message Queue)中。

(3)服务提供者从消息队列中取出服务请求消息并进行相应的处理，将返回结果发送到响应结果队列中，等待服务使用者接收。

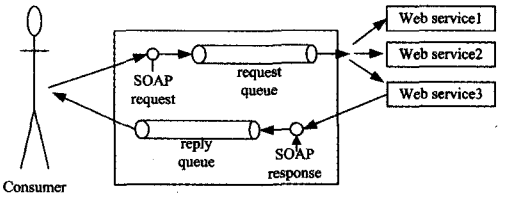

图 2 NServiceBus 的异步通信模型

3.1.2 消息发布订阅机制

NServiceBus 支持消息发布/订阅机制，图 3 展示了整个流程将经历的 4 个步骤。

(1)服务消息发布者通过调用 NServiceBus 服务总线中的 Bus. Publish(Msg)方法进行消息发布，然后向订阅数据库服务器发送获取所有订阅了该消息类型的订阅者信息集合的查询请求。

(2)订阅数据库查询到对应消息请求的所有订阅者，将信息集合回发给发布者。

(3)发布者根据订阅端点地址列表将消息请求分发到各个消息分发器。

(4)消息分发器将发布的消息请求传递给订阅的终端机器，完成一次发布/订阅消息转发流程。

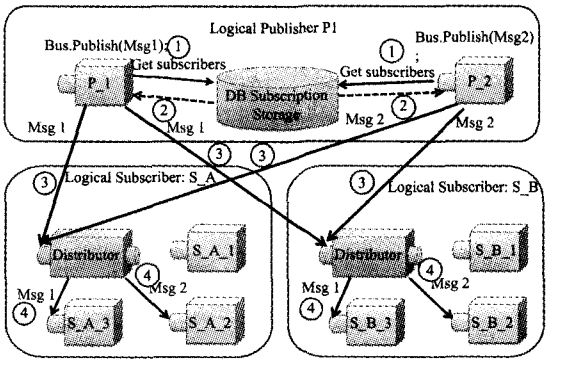

图 3 NServiceBus 的发布/订阅流程图

3.2 服务响应时间模型

服务响应时间模型通过一个客户端模拟器，定时向NServiceBus总线上发送服务请求，NServiceBus则定时向Web服务端发送服务请求，调用Web服务，记录服务响应所需要的时间，以此来判定当前负载情况下对下一个服务请求是否能够在预期的响应时间内得到响应。如图4中所示，客户端模拟器主要包含计时器、模拟消息发送器和响应时间哈希表。客户端模拟器同NServiceBus中心服务器部署在同一个局域网内，以确保消息模拟器与服务器之间的网络延迟时间基本处于稳定状态。

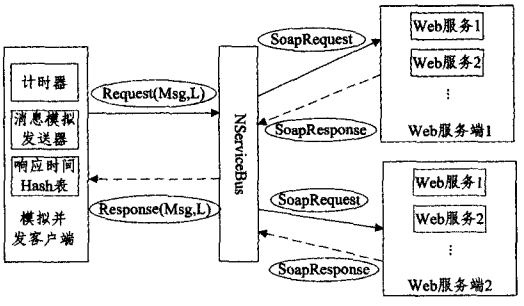

图4 服务响应时间模型

服务响应时间模型通过定时执行以下步骤来对NServiceBus服务总线以及Web服务端的当前负载情况进行预测分析。

(1)客户端模拟器通过多线程模拟并发客户端，定时发送包含消息优先级(L)和模拟标志位(T)的消息请求到NServiceBus服务总线，同时启动一个计时器记录当前的发送时间。

(2)NServiceBus服务总线接收到服务请求，根据标志位(T)得知其来自模拟客户端，那么跳过负载监测器，直接调用相关Web服务端的服务进行处理，处理结束后将结果返回给客户端模拟器。

(3)客户端模拟器接收到返回消息，停止计时器，记录此次消息处理的响应时间，并将此次服务调用过程的消息优先级、Web服务的服务标识和响应时间存入响应时间哈希表。

客户端模拟器通过定时发送的分组数据获取响应时间结果，从而得到不同优先级的消息请求的最近预测响应时间。不同优先级的消息请求的响应时间由多次模拟服务请求获取的响应时间统计之后得出。式(1)为最近预测响应时间的计算方法。

$$T_{recent(L)}=\sum_{i=1}^{n(L)}(t_{Li}-t_{delay})/n(L) \tag{1}$$

式中，L表示消息请求的优先级；$n(L)$表示优先级L下的消息请求数量；t_{Li}表示消息请求优先级为L的第i次消息请求的响应时间；t_{delay}表示网络时延，可通过实验得知。

该模型的优点在于没有复杂的模型设计，客户端模拟器与真实客户端所用的消息请求格式、网络环境都十分相似，能够较为准确地反映NServiceBus服务总线以及Web服务端的当前负载情况。

3.3 服务请求过载保护策略

通过上一小节的统计计算，记录不同优先级下的消息请求的最近预测响应时间值。由于不同优先级的服务请求对响应时间的要求不一样，因此需要对服务请求的响应时间进行质量保证，将当前负载下能够保证响应时间内返回的消息请求送到消息处理队列中。

通过对当前消息请求的$T_{recent}(L)$(最近预测响应时间)和$T_{Max}(L)$(最大响应时间)进行比较来判定NServiceBus服务总线以及Web服务端是否过载，同时拒绝过载情况下不能在预期时间内响应的服务请求。其中$T_{Max}(L)$是根据Web服务端与客户端在SLA(Service-Level Agreement，SLA)中预先制定的服务优先级为L的最大响应时间。

$T_{recent}(L)\leqslant T_{Max}(L)$表示当前消息请求最近预测响应时间比最大响应时间短，说明Web服务端属于正常负荷，则接受该消息请求并转发到待处理消息队列等待处理。

$T_{recent}(L)>T_{Max}(L)$表示当前消息请求最近预测响应时间超出最大响应时间，说明当前Web服务端负担相对较重，则该消息请求会被负载监测器拒绝并回发给客户端，并提示“服务器繁忙，请稍后重试”。

由此，能够保证在NServiceBus服务总线上注册的Web服务的负载始终维持在一个相对稳定的水平。

3.4 自适应优先级消息队列

消息请求一旦被NServiceBus服务总线接收，就会被放入对应的优先级消息队列里面等待处理。MSMQ组件支持分优先级的消息传输，优先级一共有7种，MessagePriority枚举里全部进行了封装。NServiceBus服务总线需要保证每个消息队列里面的消息请求都能够在其优先级最大响应时间之内进行处理并返回响应结果。

由于Web服务端在某个时间段总是接收到最高优先级的消息请求，因此处于低优先级队列中的消息请求很有可能不能在响应时间范围之内得到响应而被饿死在消息等待队列，从而无法对Web服务请求的响应时间提供质量保证。

为了避免上述情况的发生，图5中设计了一种自适应优先级策略来进行改进。

(1)在消息队列中，从低优先级L所在消息队列进行自适应检查。

(2)当$T_{wait}(T)>T_{waitmax}(L)$，即优先级为$L$的消息请求在消息队列中等待的时间超过优先级$L$的最大等待时间时，调用优先级更新函数将消息的优先级上升为$L+1$级。

(3)低优先级L对应的消息队列检查完毕，将L设置为$L+1$，重复步骤(1)。

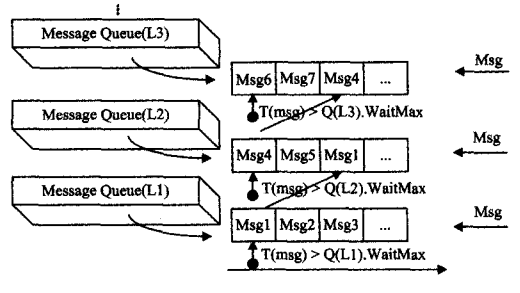

图5 自适应优先级消息队列

由以上的策略可知，如果优先级为L的消息上升为$L+1$级之后的消息请求的等待时间超过$L+1$级所对应的最大等待时间，则将当前消息请求的优先级继续上升为$L+2$级。如果该优先级下的消息仍然没有得到处理，根据自适应算法，它的优先级会不断上升，最终到最高优先级队列。在单个优先级分组队列里面，则根据FCFS(First Come First Service，

FCFS)的消息调度策略对消息请求进行调用处理，因此保证了该请求能够在$\sum_{i=L}^{L_{max}} T_{waitmax}(i)$时间之内进行响应处理。

4 基于 NServiceBus 的房地产管理系统设计与实现

本研究课题来源于香港理工大学和重庆大学的合作科研项目，旨在对房地产企业信息模型提出一种面向服务的软件体系架构，将系统内部的业务功能设计划分为粒度适中的服务，在 NServiceBus 服务总线上对分层服务进行分级调度和过载保护，提升服务调用过程的响应效率，保证服务调用质量。

4.1 项目整体框架

在面向服务的企业级应用中，域分解可以理解为将具有相似功能模块的部分独立出来，针对单个功能块内部的业务流程进行更详细的用例分析、流程分析等设计。分析本文系统，域主要可分为异构数据源访问、访问权限控制、图片文档库、地图搜索、报表服务 5 大功能域，各个域中的业务功能相对独立，因此可以针对各自单独的子域进行相对独立的系统设计，各子域提供基于 Web 服务的程序接口，最终在 NServiceBus 服务总线上实现域之间的相互通信。项目整体框架图如图 6 所示。

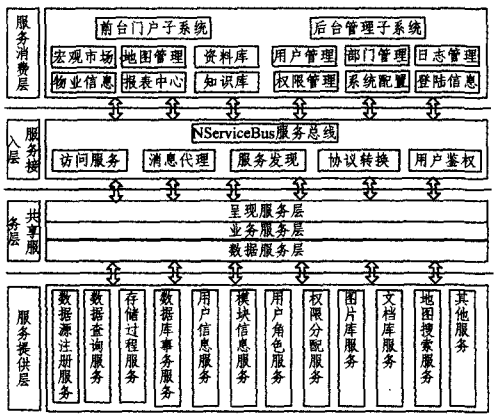

图 6 项目整体框架图

4.2 自适应优先级算法实现

NServiceBus 服务总线为了保证不同优先级的消息队列里面的服务请求都能够在其优先级最大响应时间之内进行处理，需要动态地从低优先级队列到高优先级队列进行循环监测，对超过最大等待时间值的消息进行优先级动态更新。升级某个优先级下的消息队列的代码如下所示。

```
//q:当前 Web 服务端的优先级队列
void UpdatePriority(Queue[]q){
    for(int i =0;i< q. Length && q[p]!=null;i++){
    //循环当前优先级对应的队列
   foreach(RequestMsg msg in q[i]){
    //检测是否有消息的等待时间超出最大阈值
  if((i+1) < q. Length && (System. CurrentTime - msg. InQueueTime) > q[i]. WaitMax){
        //原优先级队列删除 msg
        q[i]. Remove(msg);
        //将 msg 加入到高一级的优先级队列中
        q[i+1]. Add(msg);
        //更新 msg 进入高一级队列的入列时间
        msg. InQueueTime=System. CurrentTime;
    }
   }
  }
}
```

4.3 系统测评

测试的目的在于验证在 NServiceBus 中采用响应时间模型可使系统服务的响应时间始终控制在一个相对稳定的水平，不同优先级的服务请求、不同响应时间要求的服务请求都能够在预期的响应时间之内得到系统的处理响应，保证系统的服务质量。

系统测评的软硬件环境配置清单如表 1 所列。

表 1 系统测评软硬件配置清单

软硬件	规格
服务器操作系统	Windows Server 2003 with SP1 Pack
Web 服务器	Microsoft IIS Server 6.0
数据库服务器	Microsoft SQL Server 2005
客户端浏览器	Internet Explorer 6.0 及以上
CPU	>2.0GHz
内存	≥512M
硬盘	空闲 80G 及以上

实验测评(见图 7)中，通过客户端模拟器模拟不同并发客户数目定时向 NServiceBus 上发送测试服务请求。在没有引入响应时间模型之前的实验结果显示，随着客户端并发数的增加，服务器对服务响应时间也相应增加。而在采用本文的服务响应时间模型进行系统过载保护和优先级动态调整之后，系统在随着客户并发数增加到一定程度后，能够将响应时间稳定在 550 至 650 毫秒之间，从而对服务质量提供了一定程度的时间保障。

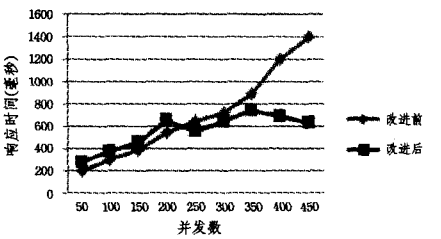

图 7 应用测评分析

结束语 本文通过引入 NServiceBus 服务总线来实现 SOA 架构中服务使用者和服务提供者之间的通信，首先深入分析了 NServiceBus 服务总线的消息转发机制以及消息发布/订阅机制，然后提出一种服务响应时间模型来对系统当前的负载进行有效预测，从而参考预测值来对系统进行过载保护。最后为了避免待处理消息队列中的低优先级消息饿死，采用一种自适应优先级消息队列来定时动态更新低优先级消息的优先级，为服务质量提供了一定的保障。通过实际应用测评和效果分析，证实了基于 NServiceBus 服务总线的 SOA 框架具有一定的合理性和实际应用价值。

参 考 文 献

[1] 徐罡，黄涛，刘绍华，等. 分布应用集成核心技术研究综述[J]. 计算机学报，2005，28(5)：433-444

[2] 刘家红，吴泉源. 一个基于事件驱动的面向服务计算平台[J]. 计算机学报，2008，31(4)：590-599

[3] 房洪臣，冷文浩，吴建波. 基于 SOA 的企业 IT 架构[J]. 计算机

工程,2008,18(23):143-185
[4] 魏楚元. Web Services 体系结构与实现机制探讨[J]. 航空计算技术,2003,33(1):101-105
[5] Jeng J J. System Dynamics Modeling for SOA Project Management[C]// Service-Oriented Computing and Applications, SOCA'07. IEEE International Conference,2007(7):286-294
[6] 徐明伟,胡春明,刘旭东,等. 一种基于 WebService 的分级 Qos 的研究与实现[J]. 计算机研究与发展,2005,42(4):669-675
[7] Liu Yan. Performance Prediction of Service-Oriented Applications based on an Enterprise Service Bus[J]. Computer Software and Applications Conference,IEEE,2007(7):327-334
[8] Jiang Ji-chen. Enterprise Service Bus and an Open Source Implementation[J]. Management Science and Engineering, IEEE, 2006(10):926-930
[9] Ziyaeva. Content-Based Intelligent Routing and Message Processing in Enterprise Service Bus[J]. Convergence and Hybrid Information Technology,IEEE,2008(8):245-249
[10] 李国勇,陈蜀宇,高峥. Web 服务中心的跨应用单点登录[J]. 重庆理工大学学报:自然科学版,2011,25(2):68-73

(上接第 130 页)

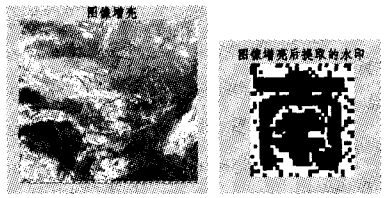

图 4 图像增亮攻击后的图像及提取的水印

表 1 为正常状态、受攻击状态下基于 DCT 的自适应盲数字水印峰值信噪比与归一化相关系数分布情况。由表可见,3 种状态下,PSNR 均大于 36.6254,NC 均大于 0.8017,具有良好的隐蔽性和鲁棒性。

表 1 基于 DCT 的自适应盲数字水印效果

攻击名称	PSNR	NC
椒盐噪声攻击	36.6254	0.8242
图像增亮攻击	36.8527	0.8017
未受攻击	37.9452	0.8512

为验证本算法的先进性,选择传统的 DCT 水印算法,在同样的平台上进行仿真实验。效果如表 2 所列(图略)。

表 2 传统 DCT 数字水印效果

攻击名称	PSNR	NC
椒盐噪声攻击	34.5456	0.7845
图像增亮攻击	35.1086	0.7427
未受攻击	36.8431	0.7890

比较表 1、表 2 可见:基于 DCT 的自适应盲数字水印较传统 DCT 数字水印,峰值信噪比提高率分别为 6.02%、4.96%和 2.99%,归一化相关系数提高率分别 8.12%、7.94%和 7.88%,这表明改进的算法在抗攻击性和透明性方面都比传统的算法要好。

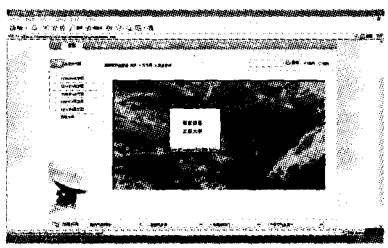

图 5 天气图和嵌入水印"南京信息工程大学"

基于数字水印的气象信息发布系统已成功用于业务中。如在天气图的上传过程中需要对天气图进行水印的嵌入,以防止天气图在发布后被非法修改。用户通过左侧的菜单项可以选择浏览相应的高空图,点击提取水印菜单项即可提取并显示嵌入在天气图中的水印信息(见图 5)。

图 6 为南京 2011 年降水量图和提取出的嵌入水印"南京"的合成界面。

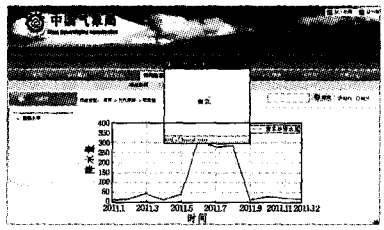

图 6 降水量预测图和嵌入水印"南京"

结束语 本文在充分考虑了卫星云图的保密性和纹理特征[8]的基础上,根据云图的纹理和暗影特征,结合人类视觉系统(HVS),采用自适应的方法将水印嵌入到卫星云图中。在 DC 系数中,采用系数奇偶量化方法嵌入水印;在 AC 中频系数中,采用确定一个固定系数去修改图像嵌入点的系数。其在保证了水印不可见的同时,对一般的噪声攻击和椒盐攻击具有很好的鲁棒性。

参 考 文 献

[1] 杨文泉,陆阳. 基于离散余弦变换图象水印算法的研究[J]. 计算机工程与应用,2003,13
[2] Cox I J, Killian J, Eighton F T, et al. Secure spread spectrum watermarking for multimedia[J]. IEEE Trans on Image Processing,1997,6(12):1680-1685
[3] Nikolaidis A, Pitas I. Region-based Image Watermarking [J]. IEEE Transactions on Image Processing, 2001, 10(11): 1726-1740
[4] 赵红,郑琳琳,陈彩琼. 一种基于 DCT 和 DWT 的自适应图像水印算法[J]. 漳州师范学院学报,2008(2):46-49
[5] 邵晓根,孙天凯,王兴元. 基于 HVS 和分形自相似的数字水印算法[J]. 计算机工程与设计,2010,31(14)
[6] 黄武辉. 基于 DCT 域的自适应盲数字水印算法研究[D]. 南昌:南昌大学,2010:36-40
[7] 路玲,孙新德. 基于图像子块 DCT 系数对的盲检测数字水印[J]. 郑州大学学报:工学版,2010,31(2)
[8] 庄陵,王光宇,邵凯. 基于离散余弦变换的 SC-FDMA 系统[J]. 重庆邮电大学学报:自然科学版,2011,23(6):691-694