第 34 卷 第 22 期 计 算 机 工 程 2008 年 11 月
Vol.34 No.22 Computer Engineering November 2008
·软件技术与数据库· 文章编号：1000—3428(2008)22—0089—02 文献标识码：A 中图分类号：TP393.09

基于 WS-Notifications 的网格服务开发模型

朱碧岑[1]，夏清国[1]，朱郑州[2]
(1. 西北工业大学软件学院，西安 710065；2. 重庆大学计算机学院，重庆 400044)

摘 要：针对轮询模型效率低下的特点，研究 Web 服务通知模型(WSN)的 3 个标准接口，根据 WSN 的特点，设计一个基于 WSN 的网格应用模型。结合实例，以网格服务中通知模型的方法在 Globus Toolkit4.0 中具体实现了该模型。实验证明，使用该模型建立网格服务具有灵活高效的优点。

关键词：Web 服务通知模型；Web 服务资源框架；网格服务

Grid Service Programming Model Based on WS-Notifications

ZHU Bi-cen[1], XIA Qing-guo[1], ZHU Zheng-zhou[2]
(1. College of Software, Northwestern Polytechnical University, Xi'an 710065;
2. College of Computer, Chongqing University, Chongqing 400044)

【Abstract】To resolve inefficiency of the polling approach, this paper focuses on three standard interfaces of WS-Notifications(WSN). It designs a grid service model based on WSN. Through the concrete steps, it implements the model using the notification based on the Globus Toolkit 4.0 platform. Experimental results show that the model is flexible and efficient.

【Key words】WS-Notifications(WSN); WS-Resource Framework(WSRF); grid service

1 概述

在面向对象分析和建模过程中，事件驱动或者说基于事件的对象间的通信是一种常用的模式。在Web服务的领域中，事件驱动同样是一种常用的模式。为了使各个厂家提供的服务之间具有更好的互操作性，以及在面向服务的体系结构(SOA)中更容易地整合各种不同的服务，需要定义一种规范来约束事件驱动模式的各种具体实现，这个规范就是Web服务通知(WS-Notification, WSN)[1]，与其相关的规范是Web服务资源框架(WS-Resource Framework, WSRF)。Globus Toolkit 4.0(GT4)实现了WSRF和WSN标准。GT4 提供API来构建有状态的Web服务，其目标是建立分布式异构计算环境。所有知名的GT3 协议都使用WSRF重新设计。

2 WSN

WSN制定了在 Web 服务环境中创建事件驱动系统的标准流程。WSN通过基于Web服务主题(WS-Topics)的发布和订阅模式定义了事件的订阅和通知机制。WSN在网格服务中使用观察者设计模式提供了一系列的规范机制和接口，从而允许客户订阅感兴趣的主题。而WSRF为表示和构成通知提供了有用的构造材料。发布Web服务主题的称为通知生产者(NotificationProducer)，订阅Web服务主题的称为通知消费者(NotificationConsumer)[1-2]。

WSN的规范主要由 3 个部分组成：Web服务主题，Web服务基本通知(WS-BaseNotification)和Web服务代理通知(WS-BrokeredNotification)[1]。

2.1 Web 服务主题

Web服务主题就是客户可以订阅的事件。一个Web服务可以发布一些主题让客户订阅，当主题变化时客户会接收到通知。一个主题可以包括一些子主题。当客户订制一个主题时，它的所有子主题也会被订制[3]。

2.2 Web 服务基本通知模型

Web服务基本通知模型定义了通知生产者和通知消费者之间通信的标准Web服务接口。通知消费者通过调用通知生产者的subscribe操作订阅Web服务主题。通知生产者通过调用notify操作告知通知消费者[4]。

如图 1 所示，首先，订阅者为通知消费者订阅了一个SystemLoadHigh 主题。接着，在通知生产者的状态发生改变时引发一个通知。比如在系统的负载超过 50%时会触发一个通知。最后，通知生产者通过调用通知消费者的 notify 操作触发这个通知。当然，通知的引发是和主题相关的。订阅者和通知消费者也可以是同一个实体，即订阅者可以为自身订阅主题。

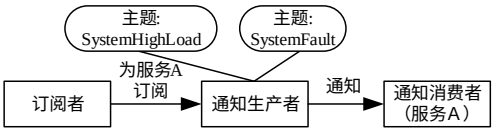


图 1 订阅者和消费者为非同一实体的 Web 服务通知模型

2.3 Web 服务代理通知模型

在这种模式下，通知从通知生产者发往通知消费者前首先发向一个代理。通知生产者必须向代理注册和发布主题[5]。订阅者也必须通过代理订阅主题，而不是直接通过通知生产者。所以当通知发生时，会通过代理发往通知消费者。

基金项目：国家发改委科学研究计划基金资助项目“CNGI远程教学公用通信平台系统”(CNGI-04-15-3A)；国家“十一五”重大科技攻关基金资助项目“数字教育公共服务平台中的若干关键技术研究”(2006BAH02A24-6)

作者简介：朱碧岑(1982－)，男，硕士研究生，主研方向：软件工程，网络技术，网格计算；夏清国，副教授；朱郑州，博士研究生

收稿日期：2007-12-25 **E-mail：**zhubicen@gmail.com

3 基于 WSN 的网格服务开发模型

网格服务的开发主要涉及 2 个部分：服务与资源。服务是建立在资源之上的商业逻辑，而资源代表服务的状态信息。服务本身仅仅是一个无状态的POJO(Plain Old Java Object)，但它可以用来发现并操作与请求相关的资源。然而服务和资源却可以在同一类中实现[4]。具体来说，服务通过实现ResourceHome接口的对象来管理和发现资源。服务通过WSDD进行配置，而WSDL则描述了服务的接口[6]，见图 2。

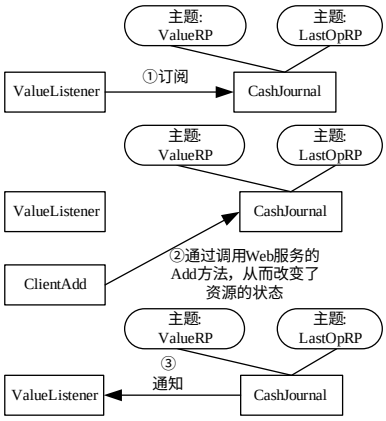


图 2 典型的 Web 服务通知模型

基于 Web 服务基本通知模型的应用主要分为 3 大部分：通知生产者，订阅程序和通知消费者。通知生产者管理 Web 服务资源的状态并发布可以订阅的 Web 服务主题，订阅程序为通知消费者订阅感兴趣的主题。当所订阅的主题发生变化时，通知消费者会接收到通知，从而进行相应的处理。

下面以网格服务Cash Journal[6]为例，描述在GT4 平台下开发Web服务通知的基本模式。在本例中通知生产者即是CashJounal服务，它处理个人账户金额的增加与减少，通知消费者ValueListener为自身订阅一个Web服务主题ValueRP并负责监听余额的变化。ClientAdd程序调用CashJounal服务改变用户的余额。程序运行的流程与 2.2 节中的描述一致，只是在本例中通知订阅者和通知消费者为同一个实体，都是ValueListener。如图 3 所示，CashJournal既是Web服务又是Web服务资源。

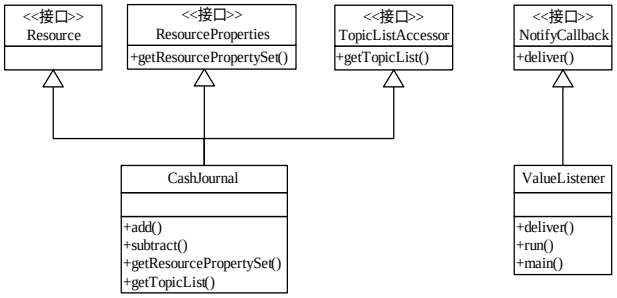


图 3 CashJounal 系统类图

Web 服务通知的设定和接收包括以下步骤：

(1)发布主题。主题是一个实现 Topic 接口的对象，生成主题后，还需要生成一个主题树，然后把主题存放进主题树。要发布主题 Web 服务资源需要实现 TopicListAccessor 接口，该接口要求实现返回一个 TopicList 的 getTopicList 方法，还要增加一个 ToplicList 属性来记录要发布的主题。最后把要发布的主题放入该属性中。在 CashJounal 服务中发布了 2 个主题 ValueRP 和 LastOpRP。ValueRP 记录了账户的余额，LastOpRP 记录最后一次操作的类型。以下发布是 Web 服务主题 ValueRP 的代码：

```
this.topicList = new SimpleTopicList(this);
valueRP = new ResourcePropertyTopic(valueRP);
((ResourcePropertyTopic) valueRP).setSendOldValue(true);
this.topicList.addTopic((Topic) valueRP);
this.propSet.add(valueRP);
```

(2)实现回调接口。通知消费者必须实现NotifyCallback接口[2]。当该通知到来时，该接口的deliver方法将被调用，在该方法中可以获取资源的新值和旧值。因为多个通知可能同时到达，所以deliver方法必须是线程安全的。由于网络的原因，因此通知可能会无序地到达，有些通知也可能被丢失。

```
class ValueListener implements NotifyCallback{
public void deliver(List topicPath, EndpointReferenceType producer, Object message)
{
…}}
```

(3)启动 NotificationConsumerManager

```
NotificationConsumerManager consumer;
consumer = NotificationConsumerManager.getInstance();
consumer.startListening();
```

(4)注册回调接口。当NotificationConsumerManager启动之后，使用createNotificationConsumer方法注册回调接口，该方法返回一个指向通知消费者的引用[2]。

```
ValueListener vl = new ValueListener();
EndPointReferenceType epr=
Consumer.createNotificationConsumer(cb)
```

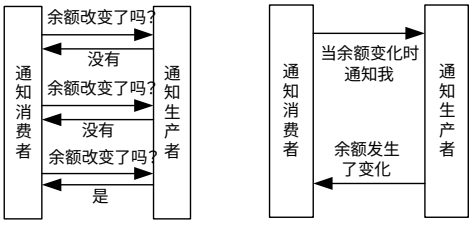
(5)使用回调接口订阅主题。把上一步中返回的引用传递给 subscribe。

```
Subscribe rq = new Subscribe();
rq.setUseNotify(Boolean.TRUE);
rq.setConsumerReference(epr);
rq.setTopicExpression(topicExpression);
```

(6)取消订阅。当通知结束后，通知消费者需要显式或者通过设定资源终止时间的方式来销毁订阅的资源。另外还需注销回调接口，然后停止对通知的监听。

4 基于 WSN 的事件通知模型与轮询模型的比较

通知是一种非常流行的软件设计模式，有时被称为观察者模式。设想一个软件有几个不同的组件，其中一个组件需要关注另外一个组件状态的变化。例如，通知消费者需要知道通知生产者的状态，从而进行相应的操作。如图 4 所示，最简单的办法是每隔一定的时间就查询服务器的状态。如果间隔较长，系统的可用性会大大降低，但是如果间隔较短，该模型又会占用大量的网络资源，同时需要不停地响应通知消费者的查询，因此，通知生产者的负担也会大大增加。



(a)轮询模型 (b)基于 WSN 的模型

图 4 轮询模型和基于 WSN 的模型的比较

(下转第 102 页)