ISSN 1000-9825, CODEN RUXUEW
Journal of Software, Vol.17, No.3, March 2006, pp.638−648
DOI: 10.1360/jos170638

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel/Fax: +86-10-62562563

可扩展和可配置事件通知服务体系结构*

汪 洋[1,2+], 魏 峻[1], 王振宇[2]

[1](中国科学院 软件研究所,北京 100080)

[2](武汉数字工程研究所,湖北 武汉 430074)

An Architecture for Extensible and Configurable Event Notification Service

WANG Yang[1,2+], WEI Jun[1], WANG Zhen-Yu[2]

[1](Institute of Software, The Chinese Academy of Sciences, Beijing 100080, China)

[2](Wuhan Digital Engineering Institute, Wuhan 430074, China)

+ Corresponding author: Phn: +86-27-87889707, E-mail: yangwang@public.wh.hb.cn, http://www.iscas.ac.cn

Wang Y, Wei J, Wang ZY. An architecture for extensible and configurable event notification service. *Journal of Software*, 2006,17(3):638−648. http://www.jos.org.cn/1000-9825/17/638.htm

Abstract: Event notification services, specially based on Publish/subscribe infrastructures, are used in a large spectrum of distributed applications as their basic communication and integration infrastructure. With their recent popularization, event notification services are required to support various specific application domains. The state of practice is that, in the development of distributed applications, developers face the dilemma of choosing between application-specific or general purpose notification servers. In this paper, a flexible solution is proposed—a dynamically customizable and extensible architecture for notification services, which is presented based on configuration management and meta-service mechanisms with the ability to customize and extend the subscription, event description, interaction protocol and configuration languages. It allows dynamical extension and customization of the notification service to satisfy not only functional requirements but also non-functional features from application domains.

Key words: event notification service; publish/subscribe system; dynamic architecture; event-based middleware

摘 要: 基于发布/订阅模式的事件通知服务,作为基本的通信与集成基础设施已广泛应用于分布式应用系统.由于日益增长的应用需求,事件通知服务需要应对各种来自不同应用领域的新需求.但在开发基于事件中间件的分布式应用时,开发者面临特殊化与通用化通知服务的两难选择.针对这种现状,提出了一种灵活的解决方法,设计一个动态可扩展和可配置的事件通知服务体系结构,允许针对不同应用领域定制不同的通知服务.该体系结构基于 XML 的可扩展订阅、事件、协议和配置语言,以配置管理和元服务管理的机制,提供了通知服务功能的动态扩展和定制以及非功能性特征的满足.

关键词: 事件通知服务;发布/订阅系统;动态体系结构;基于事件中间件

* Supported by the National Natural Science Foundation of China under Grant No.60203029 (国家自然科学基金); the National High-Tech Research and Development Plan of China under Grant No.2004AA112010 (国家高技术研究发展计划(863)); the National Grand Fundamental Research 973 Program of China under Grant No.2002CB312005 (国家重点基础研究发展计划(973))

Received 2004-02-08; Accepted 2004-06-10

中图法分类号: TP301 文献标识码: A

近些年,大量分布式应用的实现以基于事件的发布/订阅服务作为基本的通信和集成基础设施,如用户和软件监控[1]、群件应用[2]、工作流管理系统和移动应用[3]等,这些发布/订阅服务是以事件通知服务器的中间件形式出现的.由于日益增长的应用需求,事件通知服务需要应对各种来自不同应用领域的新需求.

现有的研究和商业系统,要么采用通用化的方法,通过提供非常全面的特征来满足新应用需求,例如CORBA[4]通知服务或 READY[5],这样能支持比较广的应用;要么是通过特殊化定制通知服务器,适应应用相关的需求,提供新颖但特殊的功能,例如 khronica[6]和 CASSIUS[7]用于支持群件和感知类应用,Yeast[8]专门进行局域网应用的事件处理,GEM[9]专门用于分布式应用监控的事件处理.即使像 Siena[10]这样为特殊领域设计的系统,也努力在订阅语言的表达力和特殊性之间平衡.

基于以上情况,在基于事件中间件开发分布式应用中,开发者面临特殊化与通用化的两难选择:或者使用通用化的基础设施,能支持和集成不同应用,但这样可能不能为特定应用领域提供所有必需功能;或者为每个应用领域使用一种定制的基础设施,但这样就又可能失去单一解决方案的一致性和集成性.例如,在开发感知(awareness)应用时,有的可能使用通用方案,如 CORBA 通知服务,这样可以提供非常全面的服务;有的可能使用适用特定领域的方案,如CASSIUS,为感知应用提供服务.像CORBA通知服务这样通用的基础设施,即使在功能方面非常全面,也不能为所有应用领域的各种需求提供直接支持,如感知应用中不能提供事件源的浏览和发现.而 CASSIUS 这样特定定制的系统提供了这项功能,能很容易发现信息源和来自不同组件的事件订阅.

当前可用的基于事件通知服务的另一个问题是,对要提供服务的选择和定制能力很弱.这对运行在资源有限计算设备上的应用影响很大,如移动应用.如果通用的事件通知服务用于这类应用,则服务所有拥有的特征将无论应用需要与否都提供,这势必影响通知服务在这类应用的可用性和性能.而且,现有基于事件的基础设施普遍缺乏功能可扩展机制,扩展通常是理解后改变源代码,或由应用自身实现扩展功能.加之在事件或订阅模型上的局限,增加新功能是一件非常困难的任务.结果是在大型组织结构中,为了支持不同领域应用,特征不同的事件通知服务需要并存,它们通常相互不兼容,不易互操作.

针对以上问题,我们探讨了可扩展和可配置通知服务体系结构的设计框架,为通用化和特殊化的两难选择提供了一种解决方案.该设计框架对事件模型、通知模型、订阅模型和资源模型提供了可扩展机制,同时支持事件通知服务的特殊化、扩展和定制,满足不同应用的需求.在该设计框架中,扩展性基于可扩展的事件、通知、订阅和协议语言,通过元服务(meta-service)管理来提供;可配置性是通过灵活的配置管理获得的.而通过配置管理器和元服务管理器,允许在运行时增减和替换服务,以提高系统的动态性.

本文第 1 节给出各种应用领域驱动的可扩展和配置需求及其分析.第 2 节从多维度视角给出事件通知服务可扩展和可配置的设计框架.第 3 节给出事件通知服务可配置和可扩展体系结构的设计.第 4 节讨论实现情况和相关工作.最后给出结论并说明将来的工作.

1 事件通知服务可配置和可扩展的需求及其分析

1.1 应用领域的需求

前面谈到了基于事件中间件开发分布式应用的两难局面,为了更深入地理解事件通知服务的扩展需求,需要对应用领域各种需求有全面的了解.这里,从功能和非功能性两方面分析事件通知服务 3 个最具代表性的应用领域的需求,它们分别是应用监控、感知应用和移动应用.

基于事件的应用监控基础设施,例如 EDEM[1]和 EBBA[11],具有一些共有的特征,包括事件序列检测;事件组合,即基于事件模式组合新的事件;基于内容事件过滤,即基于事件属性和类型定义订阅条件;事件源和事件类型的浏览;事件持久化.所有这些特征都是这类应用的功能性需求.

感知应用通常需要事件持久化和类型化、事件时效检查(TTL)、事件序列检测和不同的事件发送机制.而

且,这类应用的一个特征是可浏览事件源,并订阅将发布的各类事件.Khronika[6]和 CASSIUS[7]提供这种服务.

移动应用是另一个有特定需求的领域.为了能支持移动应用,通知服务器需要提供:推和拉通知策略(发布者和订阅者)、事件持久化(应付拉策略和断连操作)和所有基本功能性需求.非功能需求如漫游(由于迁移改变局部地址)和安全也需要.移动本身也可认为是非功能性需求.JEDI[3]是提供了这些特征的通知服务器.

1.2 分析框架

在文献[12]中,Rosenblum 和 Wolf 提出了一个设计框架,给出了设计事件通知服务最需关注的方面.在该框架中,对象模型描述接收通知和产生事件的组件,即订阅者和发布者;事件模型描述事件的表示和特性;通知模型关注事件传送给订阅者的方式;观察模型描述用于表示感兴趣事件出现的机制;时间模型关心事件之间的因果和时序关系;资源模型定义了在分布式系统结构中,事件观察和通知在哪里进行以及如何安置这些功能;命名模型关心系统中对象、事件和订阅的寻址.

这些模型从不同方面刻画了基于事件观察与通知的分布式应用通信和集成中的主要活动和相互关联.在对事件通知服务的可扩展和可配置能力进行扩充时,就是要针对这些反映不同方面的模型进行分析,考虑各种模型的可扩展和可配置机制.与 Cugola 等人在文献[3]中采用的方法相似,我们在设计事件通知服务体系结构时,对该框架进行了定制.

尽管 7 个模型之间相互关联,如命名模型与对象、事件、观察、通知模型都紧密相关,但考虑事件通知服务核心的活动是事件观察和通知,在我们的模型中,命名和观察模型统一考虑,两者合为订阅模型.在这种情况下,标识各种资源成为订阅描述语言的一部分.另外,考虑应用需求的增加变化,除了基本的订阅/发布消息通信机制以外,还应考虑其他交互机制,如增加安全控制的交互、移动支持的交互等.因此,增加了交互模型,允许服务器支持新的交互协议.其他方面的考虑仍然采用文献[13]中提出的模型.

2 事件通知服务可扩展和可配置的设计框架

为了使事件通知服务可扩展和可配置,系统体系结构中需要有对第 1.2 节描述的我们定制的设计框架的各个模型变化和扩展的支持.鉴于事件通知服务核心活动是:(1) 描述要观察的事件和事件模式;(2) 观察事件的发生和关联相关事件认识事件模式;(3) 通知应用被观察的事件发生,本节重点讨论事件模型、订阅模型、通知模型、交互模型以及资源模型,分别给出每个模型扩展使用的策略.

2.1 事件模型

事件最普遍的形式是元组、记录和对象.系统使用的事件模型影响系统可能的扩展.例如,类型检查是基于记录或对象的事件模型的特征,基于元组的则没有.因此,选择其一就决定了系统在事件描述与特性方面被定制的需求.

事件模型与事件通知服务中的核心组件——事件分发器紧密相关,而且应与订阅描述语言相匹配.例如,若 Siena[10]被用作系统的事件分发器,则系统的事件模型是基于元组的.基于内容的 Siena 订阅需要根据 Siena 事件来描述.在以 XML 描述订阅时,订阅解析过程还必须提供与事件描述元素匹配的特殊处理功能,以处理该语言中的标签.

为了扩展事件模型,事件描述语言和事件分发器的适配器必须定制.例如,扩展基于元组的订阅/发布内核提供类型支持.在这种情况下,类型是基于元组的事件的特殊属性,事件分发器的适配器需要增加事件类型声明管理和强制类型检查.

2.2 订阅模型

为了订阅模型可扩展,首先,事件订阅描述必须是可扩展的.订阅描述必须对事件过滤策略、事件序列组合模式、事件观察划分和分布策略具备可扩展能力.通过定义基本标签集的 XML 模式描述订阅信息,可以使该模式在多方面可扩展.例如,在事件序列组合模式方面,可以在语言中增加新元素:“(A xor B)”映射为特殊标签,表示为

〈XOR〉
〈EVENT〉 A 〈/EVENT〉
〈EVENT〉 B 〈/EVENT〉
〈/XOR〉

在订阅信息描述方面扩展后,需要针对扩展的 XML 标签——代表不同的事件序列模式、过滤策略、检测算法等,创建并通过元服务管理注册这些解析、过滤、检测算法服务.在订阅中出现该新标签将指示订阅管理通过元服务管理来查找,实例化其对应服务.

2.3 通知模型

该模型的扩展类似于订阅模型的方式.在订阅规范 XML schema 中,加入新的通知机制扩展标签和相关参数,并在元服务管理中注册扩展标签对应的通知机制服务.当通知管理获得订阅实例的通知机制信息后,通知管理通过元服务管理获得该元服务,并实例化装载.通知机制包括基本的推或拉发送、事件持久.

2.4 交互模型

增加交互模型主要是考虑体系结构在交互协议方面的扩展,主要机制是协议描述语言的可扩展以及通过配置管理,根据配置需求规范和运行时需求来选择装配交互协议.

交互协议描述为一组以 XML 定义的原语消息及其序列关系.对每个交互协议,定义一个交互服务来协调和处理该消息.在交互的认证协议扩展中,认证服务将对每个用户、系统的事件和订阅集合,负责系统用户、授权用户数据库、访问的拒绝或让步的验证.

2.5 资源模型

一些应用领域可能需要订阅活动部分在订阅方或发布方进行.这样可以将订阅处理分布,减少与中心服务器的消息交换.这对移动应用、应用监控或大规模基于事件处理的应用是非常重要的.在我们的资源模型中应对这样需求的方法是:允许选择性地在订阅方或发布方进行订阅匹配处理的部分执行.

在体系结构设计中,资源模型主要包括订阅方代理(subscriber proxy)和发布方代理(publisher proxy).在订阅方和发布方向通知服务器注册时,配置管理根据订阅方或发布方的请求,以动态代理技术[13]包装各种元服务,建立订阅方代理和发布方代理.这两种代理中介应用与通知服务的交互.

2.6 设计框架小结

以下列表呈现了在系统设计中要解决的主要设计维度,以及它们各自的扩展机制,也举例了可加入的特征.

设计维度	扩展机制	扩展特征举例
事件模型	• 可扩展的事件描述语言 • 为所使用的事件分发器提供特定的适配器 • 为事件适配器相关的事件描述语言提供处理功能	基于元组、基于类型化记录并事件、基于对象
订阅模型	• 可扩展的订阅描述语言 • 订阅描述对应的特征处理服务	事件聚合、事件组合模式、复合事件检测、事件过滤
通知模型	• 订阅描述语言通知策略定义部分可扩展 • 不同通知元服务	推模式、拉模式与事件持久化
交互模型	• 可扩展交互协议描述语言 • 处理不同交互协议的交互服务,配置管理器 • 根据配置信息静态或动态部署	安全认证的交互协议 支持移动的交互协议
资源模型	• 配置描述语言和配置管理器,允许订阅处理分布 • 动态代理机制包装处理元服务	订阅方处理部分订阅、发布方处理部分订阅

3 事件通知服务可配置和可扩展体系结构的设计

针对第 2 节给出的设计框架,在本节要设计一个可扩展和可配置的事件通知服务体系结构,关键的挑战是刻画一个基础设施体系结构和关键机制,允许增加替换事件处理功能(功能性需求),如事件过滤、复合事件检

测、新的事件组合模式等,同时允许引入非功能方面的处理,如安全控制交互和移动操作.

对中间件进行功能性或非功能性特征扩展,通常有两类策略:

(1) 服务策略:实现在中间件内核之上的服务组件,这些服务一般遵循中间件的服务组件规范,这种方法屏蔽了中间件底层,在可移植性和互操作性方面较好,但在性能上可能受损.

(2) 集成策略:对中间件核心组件进行直接改动,这种方法可以实现对应用的完全透明,一般会提供更好的性能,但是需要注意可移植性和互操作问题.

考虑事件通知服务是以发布/订阅系统为核心的特点,针对第 2 节给出的设计框架中各个模型(方面)的扩展机制,本文采用集成策略和服务策略相结合的方法来设计该系统的体系结构.

3.1 体系结构设计概述

系统设计采取 3 层结构的概念体系结构,如图 1 所示.系统可配置性体现在系统的配置管理器组件的能力上.系统在初始化或应用请求时,配置管理器根据系统配置描述信息或请求信息,装配和装载各种组件.而系统可扩展性一方面体现在以 XML 表示的事件、订阅和消息和各种策略描述的可扩展能力;更重要的是,系统通过元服务管理和各种控制组件,根据需求描述信息动态地增加、去除、替换具体的服务.

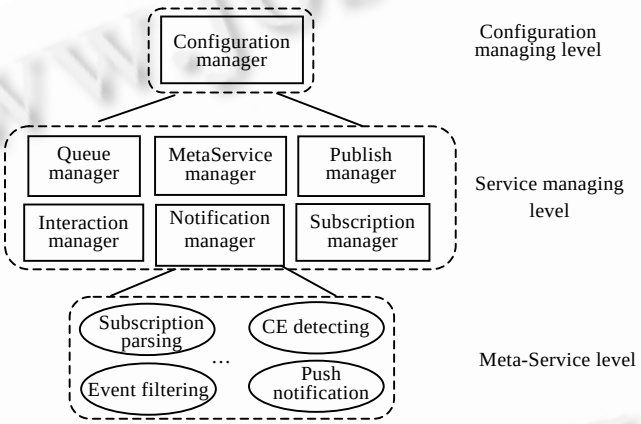

Fig.1 Three level based conceptual architecture of the extensible and configurable event notification service

图 1 事件通知服务可配置和可扩展的 3 层结构的概念体系结构

针对概念体系结构,我们给出可扩展和可配置事件通知服务的体系结构,如图2所示.

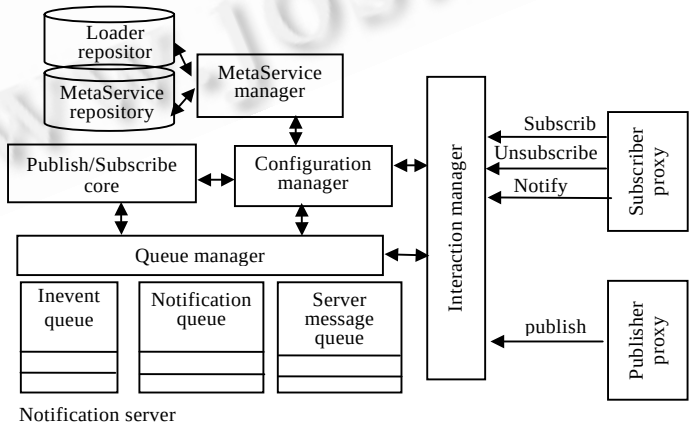

Fig.2 The architecture of the extensible and configurable event notification server

图 2 可扩展和可配置事件通知服务器体系结构

设计方法采用集中策略与服务策略相结合来进行.在该体系结构中,高层次增加配置管理器起到对系统组件全局管理和装配的作用;元服务管理器是以服务策略为主的功能扩展管理机制.而由于基本订阅/发布内核涉及事件模型、订阅模型、通知模型的扩展以及相互关联,我们不得不运用集成策略改变主要组件的设计,详细设计下文给出.

配置管理器管理整个系统组件的安置,在系统初启时,根据系统组件配置描述装载组件,协调 pub/sub 内核里的元服务管理器进行元服务的动态增减和替换,为资源模型提供部署时和运行时的改变能力.元服务管理器注册管理事件通知中的各种基本服务,响应配置管理器的管理,控制元服务生命周期.交互管理器提供了交互扩展点,使得系统不仅能处理订阅/发布消息,而且可以有其他通信机制,例如与安全认证和移动相关的协议.

3.2 体系结构的配置管理

配置管理器管理整个系统组件的安置.在系统初启时,根据系统组件配置描述(XML 格式)如环境变量、库路径、操作系统信息、组件装载顺序、依赖关系等装配组件;在运行时,与元服务管理器协调配合,进行元服务的注册与注销,而且还与其他管理组件一起,包括发布管理器、订阅管理器和通知管理器,进行各种功能服务的初始化、增减和替换.

结合灵巧代理(smart proxy)[14,15]技术,配置管理器可以静态配置与动态请求分析方式,在订阅方或发布方需要功能扩展时,创建增加部分订阅处理的订阅方代理或发布方代理,提供部署时和运行时改变资源模型的能力.

3.3 元服务管理

元服务管理器负责管理各种扩展和替换原功能模块的元服务,并注册管理生成元服务的工厂和服务类装载器.Pub/Sub 内核中发布管理器、订阅管理器、通知管理器都通过元服务管理器生成扩展原功能的元服务.元服务管理器除了自身的管理功能以及与 Pub/Sub 内核管理组件协调工作以外,它还直接管理两个组件:

- 服务装载器库保存管理元服务装载器,不同元服务装载器装载服务所需的物理资源,如二进制文件、设备资源等.
- 服务注册库保存所有成功注册到元服务管理器的服务和对应工厂,这些服务所需资源由对应元服务装载器装载,元服务管理器负责初始化每个服务.

元服务的管理组件之间的结构关系和基本交互关系分别由图 3 和图 4 描述.

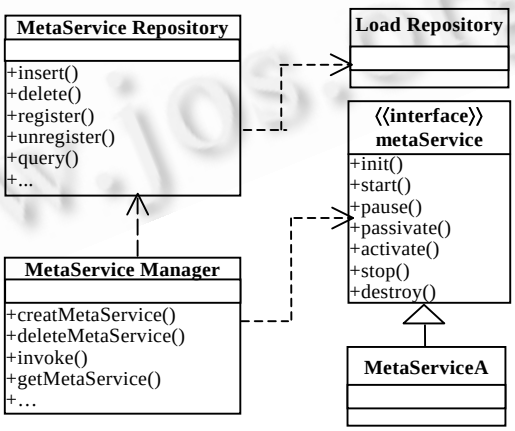

Fig.3 Meta-Service design pattern

图 3 元服务管理设计模式

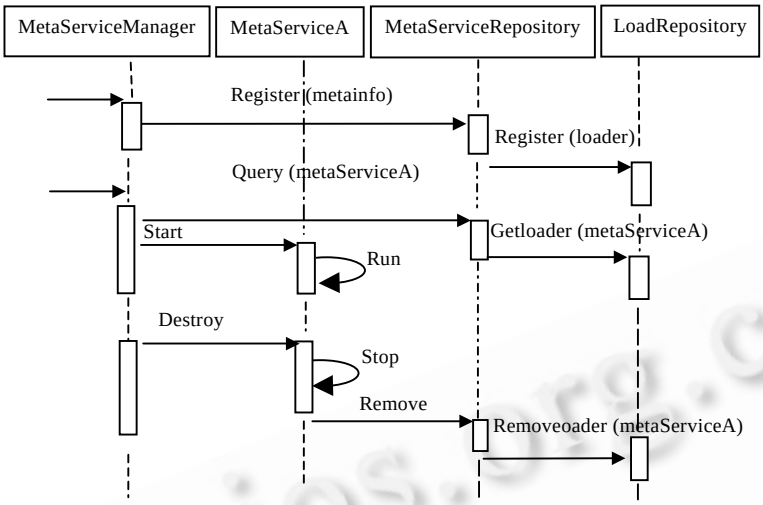

Fig.4 Example of interaction pattern of Meta-Service management

图 4 元服务管理组件交互模式例子

3.4 订阅/发布内核

订阅/发布内核基于传统的订阅/发布系统的基本功能,只是将事件分发、订阅处理、通知等功能各自由一个管理组件控制,而且这些控制管理组件与元服务管理器协调,使得既可以在系统部署时根据配置描述,由配置管理器装配各自功能处理服务,又可以在运行时动态替换、增加功能处理服务.可扩展和可配置的订阅/发布内核体系结构如图 5 所示,下面分别介绍各个组件.

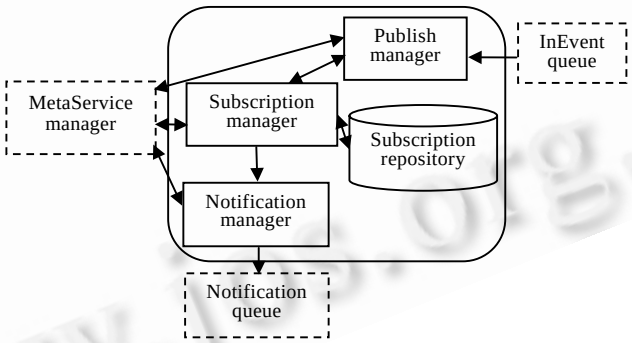

Fig.5 Architecture of the publish/subscribe core

图 5 订阅/发布内核的体系结构

3.4.1 订阅管理

订阅管理器负责对消费者需求的订阅进行处理.主要活动有:对订阅信息的管理,包括注册/注销、查询;对订阅的解析,分解订阅描述形成事件过滤、复合事件检测、事件通知需求的信息;在订阅解析后,订阅管理器还协调事件过滤和复合事件检测两个过程.图 6 描述了订阅管理涉及的主要组件以及订阅处理的主要流程,图中的有向箭头指示了流程中活动的关系.

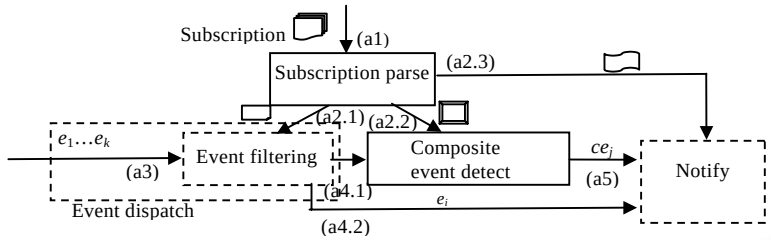

Fig.6 Sketch of subscription processing

图 6 订阅处理流程示意

• 订阅解析器.订阅以可扩展的 XML 描述语言定义,允许定义带卫士项的原子事件和复合事件以及通知选项.订阅解析器负责验证订阅,并解析出事件组合和通知选项,将各自信息转送到订阅管理器和通知管理器.

订阅解析器传送给订阅管理器的信息是事件过滤与复合事件检测所要使用的信息,解析形成的数据结构的形式决定了事件过滤和复合事件检测算法的选择.在我们设计的体系结构中,数据结构和相应算法是(动态)可配置的,由配置管理器决定.下面举例说明以树为数据结构的订阅解析.这里,简化 XML 表示的一个订阅为

(A [when $cond_1$ and $cond_2$]; B) [when $guard_A$ or $guard_B$]|(C & D)==>Notify[Pull] E

订阅解析器解析该订阅的结果如图 7 所示.其中:树叶节点代表原子事件,可附带卫士项;中间节点代表了复合事件,也具有卫士项结构;从根节点出发的节点代表通知事件并带有通知策略.

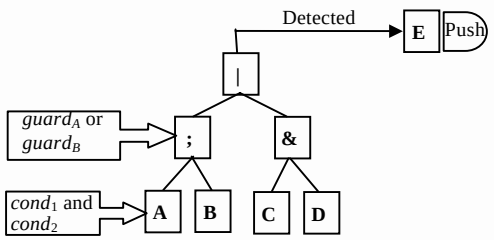

Fig.7 Tree-Based subscription parsing

图 7 基于树的订阅解析

• 订阅库.订阅库保存所有成功注册到订阅管理器的订阅信息和解析对应的数据结构,如树、自动机、Petri-net 等,以便重用.这些存储的数据结构可以减少订阅解析器对复杂订阅的重复计算.

• 复合事件检测器.复合事件检测是可扩展事件通知服务需求的主要特征之一.在我们的设计中,将多种数据结构对应的复合事件检测器作为元服务注册在元服务管理器中.当配置管理器通知订阅管理器使用何种方法解析订阅描述时,订阅管理器将装配相应复合事件检测器.复合事件检测广泛采用的方法有基于 Petri-net 的、基于树的、基于图的、基于扩展自动机的[16].目前,基于树和基于扩展自动机数据结构的复合事件检测方法显示了其有效和易用性,我们主要将这两种复合事件检测方法包装为元服务.

3.4.2 通知管理

相似于订阅管理器,通知管理器负责解释执行通知选项.在接收到订阅解析器传来的通知选项后,通知管理器分配合适的元服务进行通知的发送.目前,我们设计了 3 类通知元服务,即推模式通知服务、拉模式通知服务和事件持久化的拉模式通知模式,最后一种是为了满足移动应用需求中断连情况.例如,在图 7 中,推模式的通知被选中,则通知管理器选择推模式通知元服务来处理.

3.4.3 发布管理

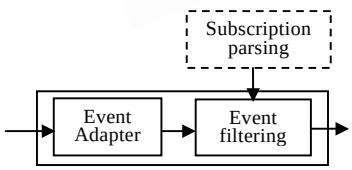

Fig.8 Event dispatcher

图 8 事件分发器结构

发布管理主要由事件分发器负责.一个简单的事件分发器提供基本的事件路由机制,决定通知服务器的事件模型.若提供一个基于内容的分发器,并且事件模型是基于元组的,则系统所有组件基于这个特征,事件描述语言也将构建在这个基础之上.

事件分发器的结构如图 8 所示.其中主要包括两个组件:事件适配器和事件过滤器.

• 事件适配器.主要根据事件模型进行相应适配包装工作,例如,扩展基

于元组的订阅/发布内核提供类型支持.在这种情况下,类型是基于元组的事件的特殊属性,开发支持事件类型声明管理和强制类型检查的事件适配器作为元服务,注册到元服务管理器.配置管理器根据需求,在部署时通知发布管理器装配相应事件适配器.

- 事件过滤器.事件过滤主要指原子事件的过滤,与事件模型紧密相关.基于主题的事件过滤与基于内容的事件过滤的复杂度实际是受事件模型的影响.在我们的体系结构中,主要考虑基于内容的事件过滤.

事件过滤器接收订阅解析器传来的带卫士条件原子事件的信息,通过事件分发器的协调与管理,选择性地装配基于内容的事件过滤算法,形成不同事件过滤器.例如,可以使用最基本的公平谓词算法,也可以使用针对树结构的 Gough 算法,以及快速紧凑的基于 BDD 匹配算法[17].

3.5 交互管理

交互服务的非功能性方面,例如安全控制和移动支持,通常需要以不同协议与服务器交互,而不仅仅是基于 pub/sub 协议.例如,安全通常需要认证协议,而移动可能需要服务器协议扩展支持 move-in/out 命令,如文献[3].支持这些和其他增加服务,需要一种机制来为服务器提供新交互机制.我们设计的交互管理器就具备这样的能力,它以消息路由器的方式工作,允许注册交互元服务处理每种协议.交互协议的选择由配置管理器决定.

3.6 订阅方代理与发布方代理

在体系结构设计中,通过灵巧代理技术扩展传统的订阅方代理与发布方代理,使之封装通知服务器的某些处理逻辑.

灵巧代理[14,15]是从服务器下载到客户端并与客户端应用链接起来的服务特定的代码,客户通过代理与服务器进行交互.灵巧代理取代传统的桩程序,为客户提供了以服务为中心的接口,在代理中可以加入服务器端的服务逻辑.灵巧代理既可静态装载也可动态装载.

订阅方代理(Sub_Proxy)或/和发布方代理(Pub_Proxy)在通知服务器处理客户端(订阅方或发布方)请求时动态生成,通过配置管理器创建相应灵巧代理,封装所需的事件处理服务.可封装的事件处理服务包括订阅解析、事件分发处理、复合事件检测、通知机制以及交互协议等.由于 Sub_Proxy 和 Pub_Proxy 是动态生成的,所以可以在运行时通过配置管理器与其他管理组件交互和协作来动态地装配 Sub_Proxy 和 Pub_Proxy 中的内容,从而实现动态配置和扩展.

图 9 显示了一个订阅方代理的内部结构.本地订阅管理器组件的使用使得订阅匹配计算在订阅方的空间进行.在客户端进行订阅处理的目的是只允许良构订阅到达服务器.

图 10 显示了一个发布方代理结构.发布方代理通过事件分发和复合事件检测服务,实现在本地的事件过滤.当条件满足时,发送原子或组合事件.该方法需要下载且在本地执行部分订阅.当被处理的事件大部分来自本地时,此发布方代理的使用非常有意义.

交互协议服务能用于实现客户端安全和移动服务,另一个选项是在本地使用通知服务.例如,发布方持久化在某些移动应用中是有用的,事件在断连时仍能被张贴发送,而当与通知服务器的连接重建后再被真正传送.

4 实现情况与相关工作

一个基于该设计的事件通知服务器体系结构原型正在基于 JAVA 实现,包括一个简单版本的体系结构配置管理器、元服务管理器、可扩展的 pub/sub 内核已经实现.

在原型中,我们现在使用 Siena 作为基于内容订阅/发布内核的事件分发器组件,利用了其简单的基于元组事件模型和基于内容的订阅匹配的能力.对基本 Siena 功能的扩展包括基于树的复合事件检测和事件规则,已经使用该框架实现.

可扩展和可配置软件体系结构的思想并不新颖,已成为计算机科学不同领域的研究主题.下面讨论一下促使我们开展此项研究的相关工作.

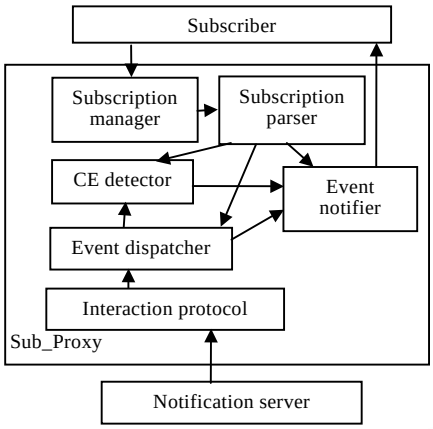

Fig.9 An example of Sub_Proxy
图 9 一个 Sub_Proxy 配置例子

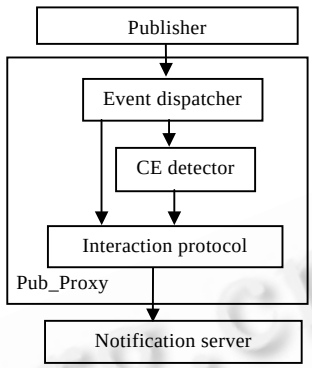

Fig.10 An example of Pub_Proxy
图 10 一个 Pub_Proxy 配置例子

文献[18]中定义了一个可插拔和可编程路由器的基础体系结构,用于定义灵活和模块化 Internet 路由器.在该体系结构中,软件模块能根据 IP 路由工作流来安排,允许表示不同策略和配置协调正确的 IP 包路由.Promile[19]是另一个系统,扩展了文献[18]中的可配置体系结构,增加了运行时改变的能力.它使用 XML 表示的图描述模块之间的互联.过滤器和策略实施设计为模块,以管道和过滤器体系结构风格安插到路由器主事件流中.一个图管理器控制组件在路由器中的动态增减.

软件体系结构和基于事件系统能组合起来,提供支持运行时重配置的框架.Oreizy 和 Taylor [21]提出使用 C2 体系结构风格支持这些改变.同样,事件处理语言如 GEM[9]和动态体系结构语言如 Darwin,能用于在应用级实现可配置分布式系统[21].

可配置中间件服务已被广泛探讨.例如,TAO[22]允许 CORBA ORB 的服务静态可配置或策略组件运行时改变.它也允许定义配置,解决移动设备的小覆盖区需求或特殊的实时约束.该工作的动机类似于我们的研究,处理特定应用领域的不同需求.

Apache Web 服务器使用可插拔体系结构,其中提供不同服务的模块可以增加.这些模块可以在请求、处理和响应过程等不同阶段被安装.这种方法与我们设计的通知服务器体系结构中的元服务相似.不同的是,Apache 使用了非常简单的管道模式(pipeline)的扩展模型,没有分布灵活性,每个请求和响应遵循同样的流程,也不允许在运行时增加新服务.

5 结束语和未来工作

在本文中,我们通过对事件通知服务体系结构可扩展和可配置框架的分析,给出了通知服务的一个可配置和可扩展体系结构的设计.该设计提供了特定应用领域的实现必需的特殊性,也提供了支持广泛应用需求的一般性.该体系结构提供了:(1) 整个事件通知服务的定制,满足不同应用的需要;(2) 支持增加新特征的可扩展性;(3) 允许在运行时引入特征的动态性.为了达到这些要求,体系结构基于 XML 的可扩展订阅、事件、协议和配置语言,以配置管理和元服务管理的机制,提供了系统功能的动态扩展和定制以及非功能性特征的满足.

未来的工作包括原型的改进,使用特定配置进行系统可扩展性和可配置性的测试,还要对具体订阅处理服务进行优化实验.引入时间模型对体系结构的影响,也是我们未来工作的一部分.

References:

[1] Hilbert-Redmiles D. An approach to large-scale collection of application usage data over the Internet. In: Proc. of the 1998 Int'l Conf. on Software Engineering. IEEE Computer Society Press, 1998. 136−145.

[2] Dourish P, Bly S. Portholes: Supporting distributed awareness in a collaborative work group. In: Proc. of the SIGCHI Conf. on Human Factors in Computing Systems. New York: ACM Press, 1992. 541−547.

[3] Cugola G, Nitto ED, Fuggeta A. The JEDI event-based infrastructure and its application on the development of the OPSS WFMS. IEEE Trans. on Software Engineering, 2001,27(9):827−849.
[4] OMG. Notification Service Specification v1.0.1. Object Management Group, 2002.
[5] Gruber RE, Krishnamurthy B, Panagos E. The architecture of the READY event notification service. In: Proc. of the 19th IEEE ICDCS Workshop on Electronic Commerce and Web-Based Applications. IEEE Computer Society Press, 1999. 108−113.
[6] Lövstrand L. Being selectively aware with the Khronika system. In: Proc. of the European Conf. on Computer Supported Cooperative Work (ECSCW'91). Kluwer Academic Publishers, 1991. 17−31.
[7] Kantor M, Redmiles D. Creating an infrastructure for ubiquitous awareness. In: Proc. of the 8th IFIP TC 13 Conf. on Human-Computer Interaction (INTERACT 2001). IOS Press, 2001. 431-438.
[8] Krishnamurthy B, Rosenblum DS. Yeast: A general purpose event-action system. IEEE Trans. on Software Engineering, 1995, 21(10):845−857.
[9] Mansouri-Samani M, Sloman M. GEM: A generalised event monitoring language for distributed systems. IEE/IOP/BCS Distributed Systems Engineering Journal, 1997,4(2):96−108.
[10] Carzaniga A, Rosenblum DS, Wolf AL. Design and evaluation of a wide-area event notification service. ACM Trans. on Computer Systems, 2001,19(3):332−383.
[11] Bates PC. Debugging heterogeneous distributed systems using event-based models of behavior. ACM Trans. on Computer Systems, 1995,13(1):1−31.
[12] Rosenblum DS, Wolf AL. A design framework for Internet-Scale event observation and notification. ACM SIGSOFT Software Engineering Notes, 1997,22(6):344−360.
[13] Sun Microsystems, Inc. Java Management Extensions White Paper. 2002.
[14] Santos N, Marques P, Silva L. A framework for smart proxies and interceptors in rmi. In: Proc. of the 15th ISCA Int'l Conf. on Parallel and Distributed Computing Systems (ISCA PDCS-02). 2002.
[15] Koster R, Kramp T. Loadable smart proxies and native-code shipping for CORBA. In: Linnhoff-Popien C, Hegering HG, eds. Proc. of the 3rd IFIP/GI Int'l Conf. on Trend towards a Universal Service Market. LNCS 1890, Springer-Verlag, 2000. 202−213.
[16] Liu Y. Event filtering for content-based Pub/Sub system. Technical Report, Technology Center of Software Engineering, Institute of Software, the Chinese Academy of Sciences, 2004 (in Chinese with English abstract).
[17] Zhang JF. Composite event detection of distributed system. Technical Report, Technology Center of Software Engineering, Institute of Software, the Chinese Academy of Sciences, 2004 (in Chinese with English abstract).
[18] Kohler E, Morris R, Chen B, Jannotti J, Kaashoek MF. The click modular router. ACM Trans. on Computer Systems, 2000,18(3): 263−297.
[19] Rio M, Pezzi N, Meer HD, Emmerich W, Zanolin L, Mascolo C. Promile: A management architecture for programmable modular routers. In: OPENSIG 2001 Workshop on Next Generation Network Programming. London, 2001.
[20] Oreizy P, Taylor RN. On the role of software architectures in runtime system reconfiguration. IEE Proc. of Software Engineering, 1998,145(5):137−145.
[21] Mansouri-Samani M, Sloman M. A configurable event service for distributed systems. In: Proc. of the 3rd Int'l Conf. on Configurable Distributed Systems. IEEE Computer Society, 1996.
[22] Schmidt DC, Cleeland C. Applying a pattern language to develop extensible ORB middleware. In: Rising L, ed. Design Patterns and Communications. Cambridge University Press, 2000.

附中文参考文献:

[16] 张菊芳.分布式系统的复合事件检测研究.技术报告,北京:中国科学院软件研究所,软件工程技术中心,2004.
[17] 刘昱.基于内容 Pub/Sub 系统事件过滤机制的研究.技术报告,北京:中国科学院软件研究所,软件工程技术中心技术报告,2004.

汪洋(1968－),男,湖北麻城人,博士,主要研究领域为软件工程,分布式系统,中间件技术.

魏峻(1970－),男,博士,副研究员,主要研究领域为软件工程,软件理论,网络分布式计算技术.

王振宇(1936–),男,研究员,博士生导师,主要研究领域为软件工程,软件理论与工具开发.