异步事件驱动的物联网服务模型

朱树人,谢勇

(广东财经大学 信息学院,广东 广州 510320)

摘 要:随着物联网应用的迅速发展,网络服务面对高并发请求,物联网服务和拥有大量在线用户的交通管理与信息网络服务器随时面临巨大访问量冲击,对网络服务质量形成威胁。本文提出了一种高性能网络服务模型。该模型基于EPCIS,用"阶段"划分网络服务,对象化的队列为主要构件,底层采用异步I/O技术,组件内部和组件间的通信采用事件驱动模型构建。设计和实现了一个基于事件驱动模型的异步Socket通信适配子层,用以屏蔽底层OS异步I/O细节。实验证明该模型对高并发请求服务具有较好的可伸缩性,对网络服务的业务流程变化具有较好的的敏捷性和适应性。

关键词:异步通信;事件驱动;网络服务;物联网

中图分类号:TP309.3;U298 **文献标志码**:A **文章编号**:1672-7029(2013)05-0093-06

Network service model of asynchronous event driven for the internet of things

ZHU Shuren, XIE Yong

(School of Information, Guangdong University of Finance and Economics, Guangzhou 510320, China)

Abstract: With the rapid development of the application of Internet of things, the network service in high concurrent requests, service for the Internet of things and a lot of online user traffic management and information network server facing huge amount of requests impact, pose a threat to the quality of service of the network. A high performance network service model was presented in this paper. The model was based on EPCIS, using the "stage" division of network service, the object queue as main component, the asynchronous I/O communication technique and component and inter component with event driven model. The design and implementation of a driven model of asynchronous Socket communication based on event adaptation sublayer was used to shield the OS asynchronous I/O details. The results show that the model has good scalability on high concurrent requests services, and business process change on the network service has good agility and adaptability.

Key words: Asynchronous communication; event driven; network service; internet of things

面向服务技术和经济全球化的日益发展,逐步形成了以独立、自治的服务供应和服务消费为特点、以网络为中心的商务结构趋势[1,2]。物联网服务和拥有大量在线用户的交通管理与信息服务系统等,不但需要承受大量持续的并发访问请求,还会遭遇巨量的尖峰负载[3-5]。高并发请求达到一定水平时,由于服务请求转化成数目更大的I/O和网络请求,需要消耗巨量的网络基础资源。研究网络服务能跟随负载同步增长的可伸缩服务能力,避免造成服务品质急剧下降具有十分重要的意义。现有的物联网服务平台尤其是交通管理与信息服务服务平台缺乏对高并发性、动态内容、连续的高可用性、对突发时变负载的自适应调节能力等需求的综合考虑[6],采用的编程模型缺乏完整的服务设计方法、缺乏保障服务整体性能的量化方法,对突发过量负载不具备自适应平滑能力[7]。本文基

收稿日期:2012-06-20

基金项目:广东省自然科学基金资助项目(10151032001000001)

作者简介:朱树人(1964-),男,湖南长沙人,教授,硕士研究生导师,从事信息系统与管理创新研究

于EPCIS(electronic product code information service),采用的阶段化异步事件驱动网络服务模型用"阶段"来划分网络服务,以对象化的队列为主要构成组件,底层采用异步I/O为基础,组件内部和组件间采用事件驱动模型通信,设计和实现一个基于事件驱动模型的异步Socket通信适配子层,用以屏蔽底层OS异步I/O细节,构造具有可伸缩性、连续的高可用性、可管理性和服务易构建性的网络服务模型。

1 事件驱动模型

Web服务器并发处理体系结构是影响服务器性能的重要因素[8-9],有多进程、多线程、事件驱动、核心类等服务并发处理。事件驱动模型是近年来逐步流行的并发管理方法[10],其目标是保持负载变化时的系统健壮性,当供应负载增大超出饱和值时,吞吐量只有很小的退化;当任务数增加时,服务器的吞吐量也增长直到瓶颈饱和;如果任务数进一步增加,过量的任务将被吸收在服务器的事件队列中。吞吐量在负载很宽的变化范围内保持恒定,而每个任务的延迟呈线性增长。Flash等Web服务器以及Harvest^Web cache采用这种模型[11-12]。这类服务器包含少量的线程,它们不停地循环,处理来自队列的事件。事件可以由操作系统或应用内部产生,并对应于网络和磁盘I/O的就绪和完成指示、定时器超时以及其他应用指定的事件。事件驱动方法在实现上把每个任务当成是一个有限状态机FSM(finite state machine),状态之间的转移由事件驱动。服务器对每个任务不依赖于线程上下文维护其连续状态。事件驱动服务器[13-15]只使用少量进程,不需要锁定和同步,也没有上下文切换开销,因此能获得更高的并发性。其缺点是开发这样的服务器没有合适的编程模型。所以适当组合使用编程简单的多线程模型将会简化开发难度。

2 异步事件驱动网络服务模型

2.1 异步事件驱动通信体系结构

异步事件驱动通信体系结构使用事件驱动模型和异步I/O技术的网络应用体系结构,主要包含底层的NIO,中间的异步通信子层,顶部的应用层3个层次[7]。模型中组件内部和组件间将组合使用事件驱动模型和多线程模型。在NIO层中,核心的组件是可层次扩展的变迁NIO,组件通过通信原语与其上层之间的数据交换。异步通信子层是一个适配层,它只有一个核心组件AsocketQ。AsocketQ是一个队列,所有与NIO和应用层之间的输入输出都缓冲在这个队列中,随后予以处理。与此类似,应用层中也包含一个队列appQ,缓冲并处理应用程序和异步通信子层的输入输出。接受1个客户请求时,该请求的服务映射为对CPU处理资源和对I/O资源的使用。请求到资源使用的1对多映射是1个同步粒度细化的过程,细化的结果是系统能更协调地使用资源,减少性能瓶颈产生的机会和持续时间。队列尽量吸收不能及时处理的请求,而不是在资源不足时简单地拒绝服务。Renew工具组件采用"类化"的方法描述多层次对象,AsocketQ和appQ可看作是同一个事件队列类的不同实例。

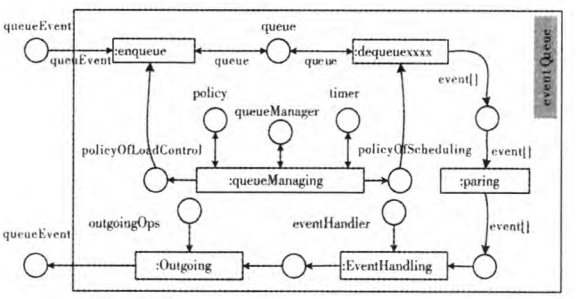

图1 异步通信队列类模型

Fig. 1 Asynchronous communication queue model

使用高级petri网描述的异步通信队列类模型(图1)给出了1个实用的队列模型,Queue是基本队列对象,定时器组件timer调用定期queueManager队列管理器,完成队列事件处理。policy是描述排队策略以及进行队列过载管理的预定义组件,变迁queueManaging使用policy实现事件调度策略,同时根据policy对队列进行过载管理,实现准入控制(admission control)等。当事件出队时,首先对队列进行分析,并通过eventHandler进行处理。例如,在AsocketQ实例中,当队列输入请求是1个TCP连接请求并调度出队时,parsing将该请求传递给EventHandling变迁,后者创建1个本地的连接代理proxy,由proxy通过outgoing变迁,在NIO上注册,通知NIO Selector监视连接操作,并使用连接请求附带的远程地址创建一个socket。当NIO完成实际连接时,在下一个轮询周期,将NIO Selector提供的连接完成事件重新入队,并将连接作为应用层事件回传给appQ队列。

图2所示为NIO的高级petri模型。NIO以注册(register)/查询(polling)方式提供异步TCP,UDP和LISTEN服务。图中9个变迁是提供给上层应用的服务接口。NIO层的工作过程如下:上层应用通过打开Selector和SelectableChannel建立可进行异步通信的环境。使用register对需要NIO监视的操作予以注册。完成注册后,上层可异步地进行connect,listen,read和write等操作[7]。NIO在完成I/O操作后,将通过SelectionKey指示。在事件处理队列一个轮询周期到达时,上层会通过调用select操作获取NIO完成的指示,并继而采取相应的动作完成异步I/O处理。

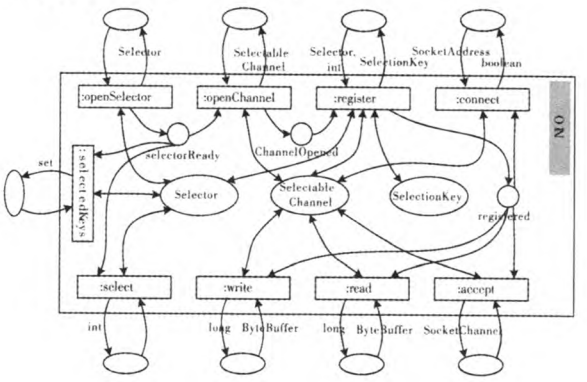

图2　NIO高级petri模型

Fig. 2 NIO Senior Petri model

2.2　服务的阶段化管理

在服务模型中使用"阶段"作为服务软件控制流程的构成组件,并以此作为设施资源分配、调度和其它管理的基本单位。设计时,服务开发者使用一系列设计模式将控制流程分解为一组"阶段",服务定义成由"阶段"所组成的图构成。"阶段"是一个自包含的软件组件,由事件处理器、可管理的事件队列、具有反馈能力的资源控制器以及驱动线程池组成。"阶段"通过独立工作的事件队列减少部件间的耦合度,使用动态资源管理技术适应突发负载的时变特征。驱动各"阶段"执行的活动实体有各自独立的线程池通过事件处理程序和线程调度部件消除耦合。服务逻辑从线程调度等核心逻辑中分离了出来,控制事件控制器的执行实现资源管理策略。应用系统内部组织是由多个事件队列连接起来的"阶段互连图"构成,"阶段"间通过队列通信,因此,具有独立的、明确的控制边界。

服务管理用业务流程处理语言(BPEL),事件驱动技术方法分析和实现。用面向过程的语法定义服务的控制和组合,在命令-控制模型上通过BPEL执行引擎实现BPEL。BPEL构建完全抽象了商务流程逻辑的服务组合层次,因此,不会对面向服务结构(SOA)造成任何影响。BPEL允许一个流程使用服务进一步封装,或由其他的服务流程组合,所以,任何一个商务流程并不限于它所处的层。通过点击的机制将事件、发布者和订阅者联系起来,执行环境的实现独立于控制引擎。事件驱动机制提供了连接孤立环境的方式,事件驱动的消息模式设计方法使得服务之间的通讯更加畅通。

2.3　ESB全局数据空间

设计1个处理简单对象访问协议(SOAP)的消息队列基础框架,SOA在HTTP层网络基础通过web service技术就能实现事件驱动机制。企业服务总线(enterprise service bus,ESB)构架提供了一种可以将消息队列和Web Service技术结合起来的途径。ESB允许事件广泛订阅,适合作容纳已发布的商务事件的容器。将ESB放入企业的全局数据空间,区域自治的多条服务总线产品结合起来提供ESB的服务,这样不管使用什么技术开发的应用在任何时间和地点都能访问事件信息。图3演示了一个企业级的服务总线是如何通过运行多个域,构造整个组织全局数据空间的。

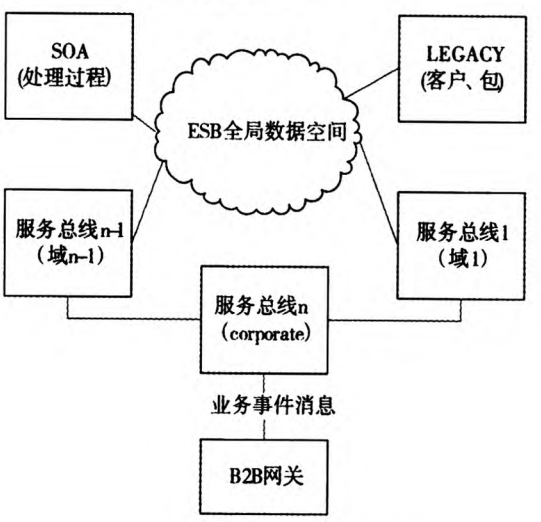

图3　ESB结构及全局数据空间

Fig. 3 ESB structure and global data space

2.4　事件驱动服务敏捷性与独立性

按需服务的商业模式要求服务提供者对外部环境的刺激(事件)快速做出反应。但供应与需求分离程度的扩大产生了对于敏捷性的需求,当组织的结构不断变化时,只有通过服务之间的松耦合才能支持业务不受阻碍的连续发展。为实现服务响应的敏捷性,网络服务系统必须不受机构的各种重

组变更的影响。服务业务流程适应机构变化的敏捷性必须不受支持它们的IT系统的限制。机构敏捷性基础是降低耦合的服务更容易在不干扰现有IT支持系统连续性的基础上改造组织的结构,因此,松耦合可以满足这些服务需求。EDA的发布-订阅模式建立了松耦合基础,作为连接边界、允许各个边界保持解耦合的状态,实现跨边界通信。服务应用搭建的灵活性可以通过定义良好的功能边界,使用共享的服务来完成。敏捷性依赖于为了重新设计商务流程而对应用进行重构增加敏捷性的能力,事件驱动能够增加服务的敏捷性。与同步、命令-控制消息交换模式的传统分布式架构不同,EDA采用基于异步、发布-订阅模式,发布者完全不需要知道订阅者的存在,订阅者也无需知道发布者的细节。服务的松耦合只共享消息的语义,EDA在联合的、自治的处理环境中,为商务流程中各个步骤之间的独立性提供支持。消息框架利用异步消息模式来实现的请求-应答通讯。EDA通过消息框架处理流程链中各个层之间的水平通讯、需要跨越功能单位边界的流程以及需要跨越物理单位界限的流程。通过将请求的发布者和应答的消费者分离,将请求的消费者和应答的发布者分离,让系统松耦合。鉴别跨越功能边界的应用,是潜在的"敏捷性瓶颈",对需要跨越机构的外部边界的应用给予较高的优先级,是解耦合服务边界的基础。

2.5 松耦合和"解耦合点"

松耦合意味着独立,松耦合的服务对彼此的依赖没有紧耦合的服务密切,功能和数据同样适合这个规则。服务相关的数据定义在服务功能边界内时,服务间的耦合度就会增加。这种设计会增加服务间访问数据的彼此依赖概率。当以数据复制、避免保证数据传输和语义一致性的共享层之外的其他层共享等机制实现的数据冗余被允许的情况下,服务之间的耦合程度会降低。EDA支持在冗余环境中的数据一致性,保证跨越解耦合边界的数据冗余让松耦合更加强健。找到"解耦合点"被证明是有效使用EDA的关键。"解耦合点"就是某些经常同时出现的商务流程,它们会一起出现在任何一个组织单位中,具有强烈的内聚性和原子类型的业务功能,即在各个业务功能之间的边界商业流程的功能构成变得清楚的位置。如果原子交易跨越了解耦合边界,那么回滚交易就需要在这些解耦合点实现。无论在这些解耦合点上的服务是连接还是断开,都不会影响被连接的系统,不同域之间的所有的数据交换都发生在这些点。在一个可重用的域中,EDA的粒度越小,可重用的域也就越小,系统的灵活性就越好。在解耦合点使用基于SOAP的Web服务技术,结合通用的ESB企业服务总线基础架构,可以很容易实现与SOAP技术封装的ERP系统、其它外部系统的网关等异构系统的无缝连接。

3 服务模型的设计实现

3.1 系统的体系结构

网络服务软件在访问网络或文件时的典型做法是使用阻塞式(blocking)I/O操作,并采用多线程复用多个I/O通道[11]。其主要缺陷是当I/O通道数目很大时性能下降严重,系统缺乏良好的伸缩性。另一种伸缩性更好的方法是使用非阻塞式(non-blocking)I/O,例如熟知的UNIX select()系统调用。为避免陷入复杂的调度决策和底层操作细节,对高层编程模型提供支持,在Java JDK NIO和编程模型之间加入异步socket适配子层Asocket,封装和屏蔽底层细节。图4描述了ASocket的体系结构,低部是JDK NIO;中间是异步通信子层,由hipserse. aFrame. Asocket和hipserse. aFrame. Asocket. nio构成;顶层是编程模型,Hiperse. aFrame. api是编程模型的API接口,hiperse. aFrame. core支持服务模型的基本运行环境,是编程模型aFrame的核心部分[8]。

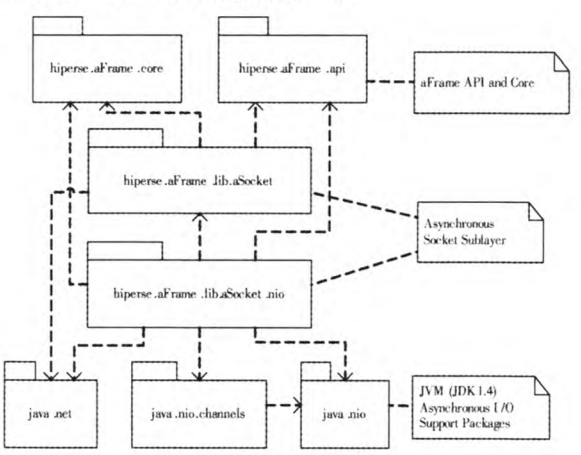

图4 Asocket异步通信体系结构

Fig. 4 Asocket communication architecture

3.2 基于EPCIS的服务实现

EPCIS是物联网的核心模块,在物联网的整体

结构中,它不同于应用层事件(Application Level Event,ALE)规范和RFID阅读器,EPCIS处于系统上层位置,它是基于ALE和RFID读取设备的管理中间件[9]。EPCIS定义了对象事件、聚合事件、统计事件和交易事件等四种事件类型。服务系统在物联网定义了捕获服务、查询服务等一系列的服务,为建立一个标准的物联网服务系统有重要的作用[10]。捕获服务的功能是捕获应用层事件,实现对标签状态间接监听。查询服务分为核心查询服务、查询订阅服务和查询控制服务。为用户提供一系列的订阅规则,服务可以根据用户的订阅规则,周期性地产生查询报告并发送给客户,便于客户了解感兴趣的信息。控制客户端的查询请求,处理例外情况或完成异常处理。EPCIS模块是一种层次结构,底层定义基础数据格式,上层提供的服务接口和绑定技术实现。在EPCIS服务器端,定义物联网的所有核心服务,允许用户开发自己的客户端来调用相应的EPC服务模块,实现对EPC系统的快速开发。

在EPCIS访问应用层嵌入Socket通信适配子层就能系统的高并发访问,增强服务系统的吞吐量,EPC服务之间的交互,需要使用对象命名服务(Object Naming Service,ONE)组件,它可以使2种服务无缝连接到一起,从而实现企业间信息交互中间件的开发,系统服务实现的数据流图见图5。ONS将一个EPC映射到一个或者多个URI,在URI中可以查找到物品的更多的详细信息,通常对应着EPCIS的地址信息,也可以将EPC关联到与物品相关的web站点或者其他Intemet资源。服务系统包含N个相对独立的EPC系统,EPC系统之间的交互枢纽是ONS模块。EPC系统包括EPCIS事件监听模块、数据持久化模块、即时查询模块、定制查询模块和本地ONS查询模块。

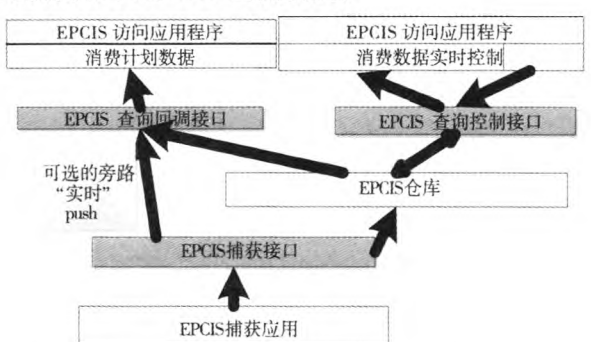

图5　基于EPCIS的异步事件驱动网络服务数据流

Fig.5 Network service data flow based on EPCIS

为了检验新模型的性能,仿真实验环境服务器采用IBM System x3850 X5(7143i19),在服务请求数量小于1 000时,系统性能没有什么变化,但当服务请求数量达到2 000时,新模型系统表现稳定,而非异步事件驱动的服务模型响应速度开始变慢,并开始出现阻塞现象。仿真实验说明,在新模型系统中请求到资源使用的1对多映射是一个同步粒度细化的过程,细化的结果是系统能更协调地使用资源,减少性能瓶颈产生的机会和持续时间。

4　结论

(1)基于EPCIS,用"阶段"划分网络服务,使用"阶段"作为服务软件控制流程的构成组件,并以此作为设施资源分配、调度和其他管理的基本单位。

(2)通过组合异步化事件驱动、阶段化编程模型等技术来满足高性能服务器高并发、可伸缩等方面的需求。并设计了1个异步Socket适配层来封装和屏蔽底层细节。整个模型采用的HLPN网来描叙。

(3)在高性能通信软件开发过程中将来需要继续完成的工作是立建模系统的性能模型,并通过性能评估发现可能的瓶颈。

参考文献:

[1] 李戈. 云环境中典型应用的I/O优化策略研究[D]. 上海:复旦大学,2012.
LI Ge. I/O Performance optimization for typical applications in cloud environment [D]. Shanghai: Fudan University,2012.

[2] 龚奕利,雷迎春,张文,等. MEANS:基于微线程结构的网络服务器[J]. 计算机研究与发展,2010,47(8):1466-1480.
GONG Yili, LEI Yingchun, ZHANG Wen, et al. MEANS: A micro-thread architecture for network servers [J]. Journal of Computer Research and Development, 2010, 47(8):1466-1480.

[3] 朱树人,彭妮. 一种具有可伸缩服务能力的高性能web代理服务器[J]. 中南大学学报:自然科学版,2005,36(6):1064-1068.
ZHU Shuren, PENG Ni. A high performance web proxy server with flexible service ability [J]. Journal of Central South University: Science and Technology, 2005, 36(6): 1064-1068.

[4] Apache Software Foundation. The Apache web server [EB/OL]. http://www.apache.org.

[5] Jack van Hoof. SOA and EDA: Using events to bridge decoupled service boundaries [EB/OL]. http://searchsoa.techtarget.com/tip/SOA-and-EDA-Using-events-to-bridge-decoupled-service-boundaries.

[6] 黄冬泉,张敏,徐振亚,等.高并发事件驱动服务器研究[J].2007,29(1):138-142,148.
HUANG Dongquan, ZHANG Min, XU Zhenya, et al. A study of highly concurrent even-t driven servers[J]. Computer Engineering & Science, 2007,29(1):138-142,148.

[7] 彭妮.阶段化异步事件驱动高性能WEB服务器[D].长沙:长沙理工大学,2006.
PENG Ni. The staged asynchronous event-driven high performance web server[D]. Changsha: Changsha University of Science & Technology, 2006.

[8] 夏卓群,朱树人,彭妮.基于事件驱动和异步通信体系结构的Web服务器设计[J].长沙电力学院学报:自然科学版,2005,20(1):49-51.
XIA Zhuoqun, ZHU Shuren, PENG Ni. The design of web server based on the construction of event-driven and asynchronous communication [J]. Journal of Changsha University of Electric Power: Natural Science, 2005, 20(1):49-51.

[9] LIU Zhiqing. Multimedia networking improvements of an embedded system: a case study [J]. The Journal of China Universities of Posts and Telecommunica, 2005, 12(3):26-32.

[10] 朱树人,刘雪峰.一种基于MPLS-DiffServ流量工程的故障恢复模型[J].铁道科学与工程学报,2006,3(4):88-92.
ZHU Shuren, LIU Xuefeng. A model for fault restoration based on MPLS-DiffServ traffic engineering[J]. Journal of Railway Science and Engineering, 2006,3(4):88-92.

[11] 贾冰.基于语义的物联网服务架构及及关键算法研究[D].长春:吉林大学,2013.
JIA Bing. Research on semantic-based service architecture and key algorithms for the internet of things[D]. Changchun: Jilin University, 2013.

[12] 魏强,金芝,李戈,等.物联网服务发现初探:传统SOA的可行性和局限性[J].计算机科学与探索,2013,7(2):97-113.
WEI Qiang, JIN Zhi, LI Ge, et al. Preliminary study of service discovery in internet of things: feasibility and limitation of SOA [J]. Journal of Frontiers of Computer Science and Technology, 2013,7(2):97-113.

[13] Guinard D, Trifa V, Karnouskos S, et al. Interacting with the SOA-based Internet of things: discovery, query, selection, and on-demand provisioning of Web services [J]. IEEE Transactions on Service Computing, 2010, 3(3): 223-235.

[14] HAO Yanan, ZHANG Yanchun, CAO Jinli. Web services discovery and rank: an information retrieval approach [J]. Future Generation Computer Systems, 2010,26(8):1053-1062.

[15] WANG Ting, WEI Dengping, WANG Ji, et al. SAWSDL-iMatcher: a customizable and effective semantic Web service matchmaker[J]. Web Semantics: Science, Services and Agents onthe World Wide Web, 2011, 9(4): 402-417.