ISSN 1000-9825, CODEN RUXUEW
Journal of Software, Vol.17, No.3, March 2006, pp.454−462
DOI: 10.1360/jos170454

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

主动分布式 Web 服务注册机制研究与实现*

杜宗霞+, 怀进鹏

(北京航空航天大学 计算机学院,北京 100083)

Research and Implementation of an Active Distributed Web Service Registry

DU Zong-Xia+, HUAI Jin-Peng

(School of Computer Science and Engineering, BeiHang University, Beijing 100083, China)

+ Corresponding author: Phn: +86-10-82327634 ext 868, E-mail: duzx@act.buaa.edu.cn, http://www.buaa.edu.cn

Du ZX, Huai JP. Research and implementation of an active distributed web service registry. *Journal of Software*, 2006,17(3):454−462. http://www.jos.org.cn/1000-9825/17/454.htm

Abstract: In SOA (service oriented architecture), the service registry takes place an important role which complies with UDDI (universal description, discovery and integration) specification. However, some tough problems are still in the way of present UDDI registry. For example, current registry has to replicate all web service publications in all UBR (universal business registry) nodes, and thus becomes impractical for a large number of web services. Furthermore, present UDDI registry is a passive directory and cannot guarantee the real-time validity of the services. In this paper, an active distributed architecture named as adUDDI is proposed for federated web service publication and discovery among multiple registries. With the distributed architecture, the service information is published within one or more adUDDIs so as to avoid the performance bottlenecks in centralized configuration. With the active monitoring mechanism, the service information is updated automatically and then the service requestor may find the latest service information. Finally, comprehensive simulations are evaluated and the results show that it outperforms the existing approaches.

Key words: UDDI(universal description, discovery and integration); Web service; active monitoring; distributed architecture

摘 要: 服务注册库作为 SOA(service oriented architecture)结构中的重要组成部分,目前主要使用的 UDDI (universal description, discovery and integration)及基于 UDDI 的 UBR(universal business registry)的实际使用情况较差.主要原因在于:UBR 的注册信息在各注册节点间完全复制,导致其随着服务数量的增加变得较难管理和维护;另一方面,目前 UDDI 只能提供被动的目录服务,而统计研究发现,很少有组织或个人在发布服务后主动更新信息,这就造成了其上服务信息的有效性差.提出了一种主动分布式服务注册机制(adUDDI),利用服务主动监测机制提高注册库中服务信息的实时有效性;利用分布式结构减轻统一注册中心的负担.通过实验分析说明,此方法可提高 SOA 结构的可用性,为基于 Web 服务构建可靠业务中间件提供了基础.

关键词: UDDI(universal description, discovery and integration);Web 服务;主动监测;分布式结构

* Supported by the National Natural Science Foundation of China under Grant No.90412011 (国家自然科学基金); the National High-Tech Research and Development Plan of China under Grant No.2001AA113030 (国家高技术研究发展计划(863))

Received 2005-06-30; Accepted 2005-10-11

中图法分类号: TP393 文献标识码: A

为了适应 B2B 电子商务的需要,企业 IT 架构正逐步转向面向服务的结构(service-oriented architecture,简称 SOA)[1].服务提供者利用 WSDL[2]描述服务接口,然后将 Web 服务发布到服务注册库中,以便服务请求者获取该服务信息.可见,服务注册库是 SOA 结构中的重要组成部分,Web 服务中定义服务注册库的主要标准是 UDDI 系列规范[3].UDDI 的主要目标是实现全球统一的服务目录(universal business registry,简称 UBR),以维护公共可见的服务信息.目前,UBR 由 IBM,Microsoft,SAP,NTT 四家公司维护,每天各节点相互复制信息.由于业务服务的多样性、复杂性以及安全性等原因,Web 服务数量的增加导致 UBR 的维护变得不可操作.事实也表明,目前使用较广泛的不是 UBR,而是遵从 UDDI 规范的私有服务注册库.但由于各私有服务注册库间没有相应的互通机制,导致新的服务孤岛的形成.

Kim 等人在文献[4]中研究了 2003 年~2004 年公共 Web 服务的使用情况,其数量并没有明显地增加[5],大约有近 1 200 个服务注册在 UBR 上,而其中只有 34%的服务是可用的,这里的可用是指注册信息中包含了 WSDL 的信息,但其中还有很多可下载的 WSDL 实际上是错误的(主要包括语法错误,或者缺少必须元素),因此实际可用的服务信息很少.另一方面,数据表明很少有服务提供者在注册了服务信息后会主动更新服务信息,而且由于网络问题导致的服务失效也不可能由提供者主动更新.总体来看,目前 UDDI 的主要问题包括:1) UDDI 采用大集中模式,随着服务数量的不断增加,其维护、管理将变得很困难;2) UBR 的目标是要存储所有的服务信息,这些不区分行业、类别的信息混杂在一起,导致服务查找的效率较低;3) 目前广泛使用的私有 UDDI 之间缺乏互通机制,导致了新的服务孤岛的出现;4) 目前,UDDI 是被动的服务目录,被动等待服务请求者查找服务,被动等待服务提供者注册、更新服务信息,而在实际检测中发现,每周大约有 16%的已注册可用的 Web 服务已经失效.这就导致了我们从 UDDI 中查找到的服务信息的实际可用性较差.

针对以上问题,本文提出了主动分布式的服务注册机制(active distributed UDDI,简称 adUDDI),通过分布式结构合理组织私有或半私有服务注册库,使其方便地互通,将业务服务信息转到各行业服务注册库上,避免大集中模式导致的执行瓶颈以及信息维护困难等问题.通过主动监测机制,使得服务注册库中记录的服务信息实时、有效,提高了构建在 SOA 结构之上的应用服务的可用性.

1 主要相关工作

灵活的资源发现和管理是跨企业协作的基础.传统的集中式资源管理的可扩展性较差[6].近年来,分布式结构的可扩展性和灵活性得到了越来越多的重视[7,8].

UDDI 系列规范是目前广泛使用的 Web 服务注册库规范,2004 年 10 月推出的 UDDIv3.0.2 承认全球统一的服务注册库不能解决 SOA 结构的所有问题,提出了附属注册中心的概念,用于指代目前广泛存在的私有或半私有服务注册中心.但其附属注册中心的服务信息是对 UBR 信息的部分复制,而 UBR 仍然记录所有的公共信息,附属注册中心主要作为信息的本地缓存存在,其大集中的模式没有改变.

IBM 提出的服务发布和发现协议 ADS(advertisement and discovery of services protocol)[9]利用 UDDI crawler 从 Internet 中自动拉回服务信息广播,而不需要人工将服务提供信息注册到注册库中,即建立了主动的服务注册机制.它借用 Web 页面搜索引擎技术,在 Web 服务器的根目录中放置 svcsadvt.xml 文件,crawler 看到该文件后将采集该 Web 站点的所有服务信息.ADS 利用 crawler 的主动采集技术,去除了每个服务均需注册自己的负担.但 Web 服务信息不同于 Web 页面信息,Web 页面搜索引擎技术中的 crawler 能自动收集信息,很关键的一点是通过收集该 Web 服务器页面文件中的超链接,进入下一级节点.而 Web 服务的描述信息通常仅描述该服务自身的功能接口及特征,没有与其他 Web 服务的链接关系,因此 crawler 的自动扩散很困难.UDDIe[10]是英国 Cardiff 大学提出的针对 UDDI 的扩展,它主要提出了服务的租赁时间和属性包概念.其中服务租赁时间是指服务可定义其租赁 UDDI 空间的时间,在服务租赁的时间内,UDDI 确信服务的信息是实时有效的,过了租赁时间或没有定义租赁时间,则 UDDI 不提供信息有效性保障.但其有效性保障主要是依靠 UDDI 与服务提供者之间的

信任关系建立,并没有提供相应的监测机制来确认服务的实际有效性.

MSWDI(METEOR-S Web service discovery infrastructure)[11,12]是美国 Georgia 大学提出的分布式 UDDI.它是一个可扩展的在多个注册库之间执行联合服务发布及发现的框架,主要考虑语义 Web 服务的发布、发现.提出了注册库 Ontology 来维护各注册库间的关系,由于 Ontology 的维护是另外一套语义维护体系,增加了实现和使用的难度.另外,该研究不解决 UDDI 中信息有效性的问题.荷兰 Tilburg 大学和德国的 Fern 大学试图将 Web 服务与 P2P 网络融合在一起[13],提出了结合 P2P 网络的服务注册系统,主要考虑在某服务领域内部将服务注册于统一的注册库中,然后加入感兴趣的联合体,每个联合体内部是 P2P 的结构.其服务信息仍主要发布在统一的注册库中,无法有效解决执行瓶颈问题.该研究也不能解决 UDDI 中服务信息的可用性问题.

2 主动分布式服务注册机制

adUDDI 主要通过对服务的主动监测机制保证 adUDDI 上注册的服务信息的可用性,通过分布式 adUDDI 的互连结构,减轻中心节点执行瓶颈问题,提高服务注册库的实际可用性.以下分别介绍 adUDDI 的主动监测机制以及分布式结构.

2.1 主动监测机制

UDDI 实用的一个重要前提就是其注册信息的实时有效性,即需要保证服务请求者在其上找到的服务信息是实际可用的.如前所述,由于很少有组织在发布服务信息后主动进行更新,因此必须提供服务的主动监测机制,由服务注册中心主动向服务提供者发起监测请求,监测服务的当前状态,以提高注册中心服务信息的可用性.本文提出的主动监测机制为定时监测机制,即 adUDDI 定时向注册于其上的服务提供者发送监测请求.

单个 adUDDI 服务器的状态图如图 1 所示,其中正常状态(normal)是指 adUDDI 被动等待触发事件的状态;监测状态(monitor)指 adUDDI 主动向服务提供者发起监测请求的状态;更新状态(update)是指 adUDDI 根据服务提供者的返回信息更新服务注册库的状态.处于正常状态的 adUDDI 由定时触发事件触发,进入监测状态,运行主动监测算法,如果被监测服务没有更新,则 adUDDI 由监测状态回到正常状态,等待其他事件触发;如果需要更新,则通过更新信息触发进入更新状态,运行服务信息更新算法,更新结束后,adUDDI 回到正常状态,等待其他事件触发.adUDDI 是对 UDDI 规范的扩充,它仍然提供接口接受服务提供者对服务的更新请求,在正常状态下的 adUDDI 也接收服务提供者的更新请求,由更新请求触发进入更新状态,运行服务信息更新算法,更新结束返回正常状态.

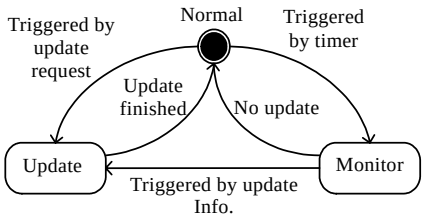

Fig.1 The state chart of adUDDI

图 1 adUDDI 状态图

图 2 显示了 adUDDI 监测单个服务时的交互协议 AMP(active monitoring protocol),AMP 涉及的角色包括 adUDDI 和服务提供者(是指负责执行 Web 服务功能的服务器,简称 SP),我们用三元组⟨*sender*,*receiver*,*message*⟩来描述 AMP 协议中的单条消息,其中 *sender* 表示消息的发送者角色,*receiver* 表示消息的接收者角色,*message* 表示实际发送的消息,这样我们可将 AMP 协议描述如下:

1) adUDDI 向服务提供者发送监测请求,即⟨adUDDI,SP,'Monitor'⟩,其中"Monitor"消息中包含目前 adUDDI 中注册的服务名称、键值以及当前服务版本号;

2) 服务提供者按照服务名称、键值以及版本号与自身服务的基本信息进行匹配,若完全一致,则返回"nonUpdate"消息,即⟨SP,adUDDI,'nonUpdate'⟩;

3) 若不完全一致,则将新的服务信息通过 UDDI 提供的接口"save_Service"返回给 adUDDI,即⟨SP,adUDDI,'save_service'⟩,其中更新后的服务信息作为 save_service 消息的参数返回;

4) adUDDI 若收到"nonUpdate"消息,则结束监测线程;adUDDI 若收到"save_service"消息,则执行更新服务算法.若在规定时间内未收到返回消息,则说明服务当前无效,adUDDI 进入更新状态,将该服务更新为(unavailible)无效服务.

值得注意的是,无效服务可能是因为网络的异常或服务提供者服务器的短期失效引起的,也有可能是因为服务本身已经被取消引起的,因此我们不能简单地删除无效服务的注册信息,而应通过服务监测策略来决定如何处理无效服务.例如,我们可设定策略为:若连续 10 次监测都无法获得服务信息,则将该服务彻底删除;若某次监测得到了正确返回结果,则根据返回结果更新服务信息,并将该服务状态恢复为可用状态.服务请求者查询、请求服务时,仅在状态为可用的服务中进行查询、选择等.

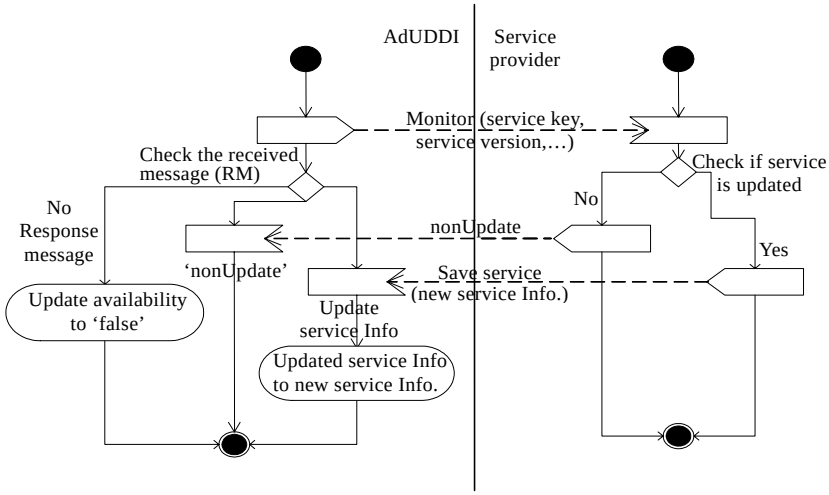

Fig.2 The interaction process of active monitoring

图 2 adUDDI 主动监测交互协议

为了实现 AMP 协议,服务提供者需要提供相应的功能接口 monitor(serviceName,serviceKey,service Version),用于接收 adUDDI 发来的监测请求,比较、判断服务信息是否已更新,并返回结果,完成 AMP 协议.以上操作给服务提供者增加了一定负担,但其负担增加很小:首先,由于服务提供者本身需要对外提供操作接口,因此接收、发送 SOAP 消息不额外增加负担;第二,monitor 操作负责进行服务名称等内容的字符串匹配,功能实现所带来的负担很小;第三,通过该部分功能扩充可使得服务提供者的信息自动在注册库中得到更新,反而减少了服务信息更新带来的负担.因此,AMP 协议通过较少的额外负担提高了服务注册库中信息的可用性.

2.2 分布式结构

为了提高服务注册库的有效性,本文提出了分布式结构来解决以上问题,结构如图 3 所示.

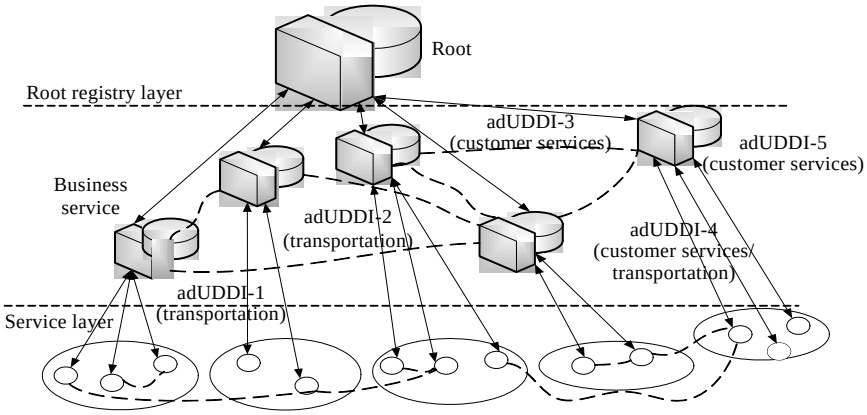

Fig.3 The distributed architecture of adUDDI

图 3 adUDDI 分布式结构示意图

结构分为 3 层:顶层为根注册中心,注册各 adUDDI 服务,不接受业务服务的注册与查找,以减轻根注册中心

的负担;中间层为 adUDDI 层,各 adUDDI 按照 GICS 全球行业分类标准[14]分类建立,一个 adUDDI 中可以注册一类或几类行业服务.每个 adUDDI 作为特殊的 Web 服务注册于根注册中心;底层为服务层,每个业务服务注册于某 adUDDI 上,adUDDI 负责管理、维护注册于其上的服务信息.为了提高服务查询效率,每个 adUDDI 中建立服务信息缓存空间,以记录请求者查询过的服务信息.缓存空间有限,长期没有被查询的信息会被新的缓存信息替代.

在分布式结构中,我们必须考虑 adUDDI 的加入与退出机制、服务的注册、更新机制以及服务查找机制.

1) 新 adUDDI 的加入

新 adUDDI 加入时,首先需要将自身基本信息注册到根注册中心;然后,在根注册中心查找与其属于同一行业的其他 adUDDI;根据查找结果,与全部或部分同行业其他 adUDDI 分别建立连接,请求与其建立邻居关系;如果被请求方同意,则双方均在自己的邻居表中记录对方信息,邻居关系建立.邻居关系建立后,业务服务的发布和查找将仅在中间 adUDDI 层进行,而不涉及根注册中心.新 adUDDI 加入的协议如图 4 所示,通过 adUDDI 之间邻居关系的建立,可降低根注册中心的负担.

2) adUDDI 的退出

adUDDI 的退出包含两方面问题:1) 是否向根注册中心发出退出请求;2) 是否对注册于自身的服务进行相应处理.根据这两个问题的不同处理方法,可产生 4 种 adUDDI 的退出机制:第 1 种是放弃注册于自身的服务,直接关闭 adUDDI,不通知根注册中心;第 2 种是简单放弃注册于自身的服务,并向根注册中心更新自身信息为无效;第 3 种将自身服务委托给邻居 adUDDI,直接关闭 adUDDI,不通知根注册中心;第 4 种是将自身服务委托给邻居 adUDDI,并向根注册中心更新自身信息为无效.以上 4 种方法处理复杂度依次增加,第 4 种的处理复杂度最高,但却是最有利于根注册中心的管理并保证服务提供者利益的一种处理方法.由于 adUDDI 的退出可能由不同原因引起,因此可以使用不同的退出方法.例如,adUDDI 使命结束,正常退出时,可能选择第 4 种方法;而由于网络的原因导致 adUDDI 无法与根和其他服务建立连接时,则只能选择第 1 种处理方法.

如果 adUDDI 主动向根注册中心请求更新其信息为无效,则根注册中心直接删除该 adUDDI.若 adUDDI 未通知根注册中心,而由根通过主动监测机制发现 adUDDI 无法连接时,则根据根注册中心监测策略,先将其信息置为无效(unavailable),而不能直接删除.另一方面,待退出的 adUDDI 若简单放弃注册于其上的服务信息,则无须作任何处理,关闭 adUDDI 即可.若 adUDDI 要将其服务信息委托(transfer)给其他 adUDDI,则需要征得对方的同意,并需通知服务信息所属的服务提供者.具体交互协议如图 5 所示.

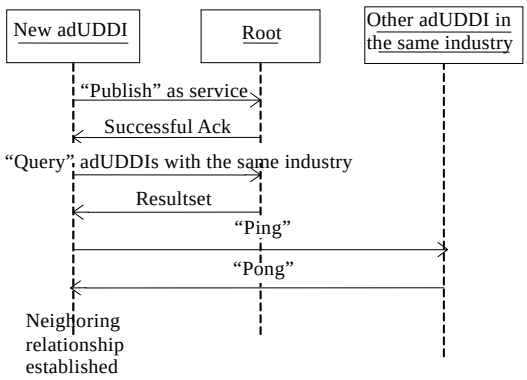

Fig.4 Adding an adUDDI

图 4 新 adUDDI 加入的交互过程

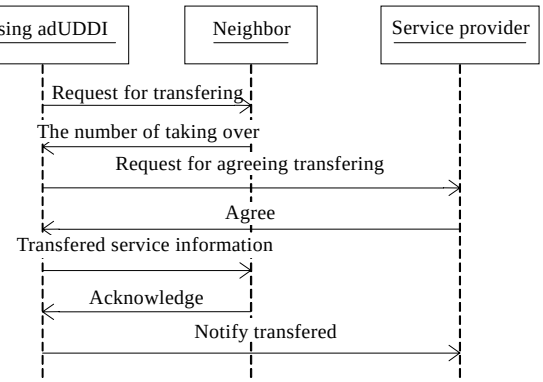

Fig.5 Closing an adUDDI

图 5 adUDDI 节点退出的交互过程

3) adUDDI 邻居关系的维护

新的 adUDDI 加入时,通过加入协议建立了 adUDDI 间的邻居关系.而由于各 adUDDI 的自治性,其更新或退出不一定会通知邻居,因此需要建立邻居关系维护机制.邻居关系的维护主要利用根注册中心的主动监测机制来实现.根注册中心主动监测 adUDDI 服务的状态,发现 adUDDI 信息变化时,将该 adUDDI 的信息广播至同

行业的其他 adUDDI,以更新邻居表,其中的广播机制直接利用 UDDI v3 中提出的订阅机制.

4) 服务的发布

服务提供者根据自身所属行业类别(industry classification),注册为该行业的某 adUDDI 服务器的用户,该 adUDDI 负责管理、维护服务信息.若服务提供者不了解相关行业的 adUDDI 信息,则首先需要请求根注册中心提供相关 adUDDI 服务器信息,然后根据自身需求选择 adUDDI 并注册.服务的实际发布过程直接使用 UDDI 中的 save_business,save_service 等 API 实现.

5) 服务的发现

各 adUDDI 负责管理注册于其上的服务信息,并接收服务请求者发来的服务查询请求.为了向请求者提供更多、更好的服务选项,adUDDI 不仅在本地服务目录中查询,还同时将请求信息广播给它的邻居,以取得更多的服务选择.

图 6 显示了服务请求者与 adUDDI 的交互协议.服务请求者首先在连接到的 adUDDI 缓存中查找服务信息,如果缓存中有所需服务信息,则将结果返回,查找过程结束;如果本地无缓存,则 adUDDI 在本地查询服务,同时将查询请求信息广播给邻居 adUDDI,并等待本地和邻居的查询结果返回;将返回的结果信息加入缓存中,最后将结果信息按请求者要求返回.

6) 服务信息扩散同步机制

上节描述的服务主动监测机制保证 adUDDI 上记录的服务信息是实时有效的,但在分布式结构下,由于缓存信息的存在,需要将单个 adUDDI 上服务信息的更新扩散至缓存了该信息的邻居 adUDDI 上,以同步缓存信息.由于同步更新机制的存在,使得上节描述的 adUDDI 的主动监测状态图扩展为图 7,在服务信息完成更新后,需要向邻居 adUDDI 广播更新信息,邻居 adUDDI 利用服务信息扩散同步机制实现缓存中相关服务信息的更新.

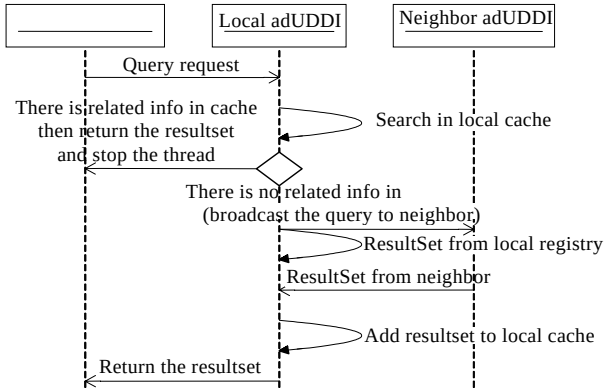

Fig.6 The interaction protocol of service query

图 6 服务查找请求交互过程

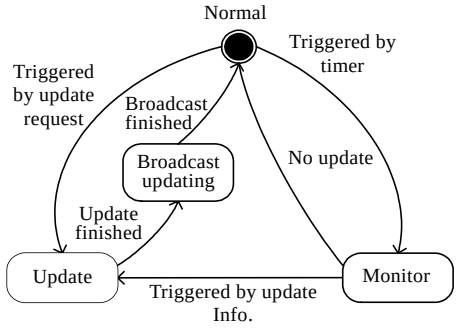

Fig.7 The extended state chart of adUDDI

图 7 扩展的 adUDDI 状态图

3 仿 真

我们设计了系列仿真实验来仿真 adUDDI 结构以及相关的服务提供者、服务请求者,并在其上通过服务查询请求的成功率、响应时间、网络流量等度量标准来评估 adUDDI.第一,我们利用 BRITE[15]拓扑产生工具产生了 2 000 个节点的网络拓扑图,每两个节点间的网络延迟依据节点间的最短路径算法计算;第二,建立仿真服务结构,每个服务至少包含服务名称、服务版本、服务类别、服务存取点(仿真网络结构中的节点号)等基本信息.我们生成不同个数的服务随机地部署到网络拓扑结构中;第三,建立仿真 adUDDI 结构,一个 adUDDI 可注册多个行业的服务,一个行业的服务也可以注册到多个 adUDDI 上.我们将 adUDDI 分布到有限的行业中,并将服务按照服务类别发布到 adUDDI 上.根注册中心为特殊的 adUDDI 节点,仅注册各行业 adUDDI 的服务信息,不接受业务服务的发布信息.各 adUDDI 生成后,也需要映射到网络拓扑结构中;第四,建立仿真请求,为了避免大量无

效请求(注册库中根本没发布过的服务的请求)影响评估结果,我们将仿真请求限制在已有的服务类别范围内,仿真主要考虑已发布的服务变化对于服务查询请求的影响.服务请求同样需要映射到网络拓扑结构的节点中.

3.1 度量参数

adUDDI 的主要目标是更好地为用户找到可用的服务.为了评估 adUDDI,我们使用了下列度量标准:

有效率(available rate):请求者在 adUDDI 上找到的第 1 个符合功能条件的服务就是可用服务的请求个数占总发起请求个数的比例.

成功率(success rate):服务请求者在 adUDDI 上找到可用的符合条件的服务的请求个数占总发起请求个数的比例.即无论花费多长时间,检查多少个备选服务,最终能找到可用服务的请求个数占总请求个数的比例.

平均响应时间(average response time):服务请求者在 adUDDI 上找到第 1 个当前可用的服务时间为一个请求的响应时间,其中包括了服务的查找时间以及服务可用性确认时间.多个请求的响应时间的平均值即为平均响应时间.

网络负载(total traffic cost):请求者的服务请求带来的网络流量,包括了请求、响应的流量.在 adUDDI 结构中,还包括后台主动监测带来的流量和信息后台同步更新的流量等.

3.2 仿真结果分析

本节我们将 adUDDI 与现有的没有主动监测机制的集中式 UDDI 实现方法进行比较,分别进行了 4 组仿真实验,并通过上节提到的度量参数来进行 adUDDI 的评估.

第 1 组仿真实验的目的是为了说明主动监测机制的加入对 adUDDI 上获取的信息可用性的影响,如文献[4]中报告,UBR 中每周大约有 16%的 Web 服务失效.我们以此为依据仿真服务变化情况,同时每 3 天发出 10 000 个请求来检查从 adUDDI 上找到的第 1 个满足条件的服务的可用性,其结果如图 8 所示.可以看出,随着时间的推移,没有主动检测机制的 adUDDI 上查询请求的有效率单调下降,随着时间的流逝,有效率会变得很差;而加入主动监测机制可对查询有效率有很大的改观,主动监测频率越高,查询有效率也就越高.

第 2 组仿真实验是为了检验从 adUDDI 上找到可用服务信息所需时间的分布情况,我们生成 1 000 个服务及 10 个 adUDDI 注册中心,将服务随机分布在 2 000 个网络节点上,并按照行业注册到相应的 adUDDI 上.仿真运行时,我们将 10 000 个查询请求随机分布在 30 天的时间范围内,仿真结果如图 9 所示.横轴时间是从服务请求发起时间到找到第 1 个满足条件的可用服务的时间.由图可见,没有主动监测机制时,在 1.9s 内,有 69%的请求找到可用服务,而另外有 15%的请求根本无法找到可用服务.加入主动监测机制后,在 1.9s 内找到可用服务的请求比例大幅增加,例如,监测频率为每天监测时,96%的请求可在 1.9s 内找到可用服务.可见,主动监测机制可促进快速获得可用服务.

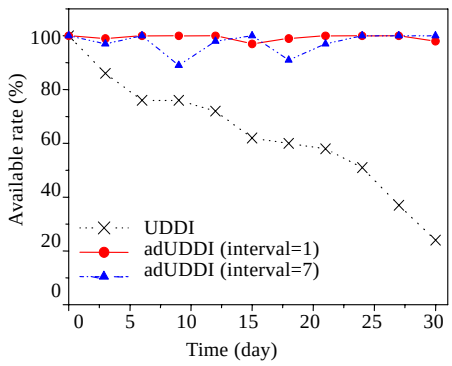

Fig.8 Available rate vs time

图 8 查询有效率随时间的变化曲线

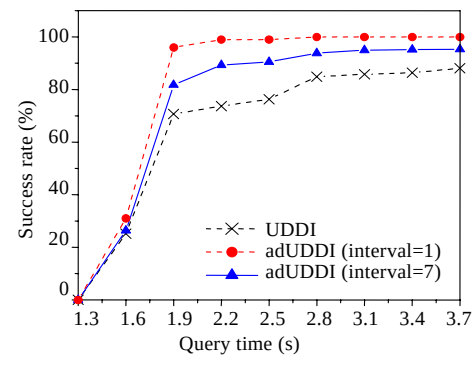

Fig.9 Success rate vs query time

图 9 查询成功率比查询执行时间

第 3 组仿真是为了观察服务节点个数以及主动监测频率对服务查找请求的平均响应时间及网络流量的影

响.我们将 10 000 个请求随机分布在 30 天的范围内,通过仿真实验,请求的平均响应时间并不会随着服务数的增加有明显的变化,因为服务数主要影响注册库的规模,而在 100 条记录和 1 000 条记录中查找信息的时间差别很小.另一方面,如图 10 所示,监测频率越高,平均响应时间就越短.这主要是因为监测频率越高,越能快速找到有效的服务,因此可减少检验服务有效性的次数,进而缩短平均响应时间.

图 11 显示了网络流量在不同服务节点数下随监测频率变化的情况,仍然取 10 000 个请求分布在 30 天的时间,10 个 adUDDI 记录服务,仿真结果记录 30 天所有使用的相关网络流量,对于没有监测机制的情况,仅考虑服务查找带来的流量,而对于有监测机制的情况,需要同时考虑定时监测机制带来的流量.在监测频率很高时,监测开销很大,而没有监测机制的情况下,节省了监测流量,但由于找到有效服务可能需要更多次的检查,因此服务请求的流量又会增加,网络负载的曲线不是单调变化的,而是在监测频率为 1 周时,得到最小的网络流量.综合网络流量和图 8 所示的查询有效率可知,主动监测的频率并不是越高越好,而将监测频率设定为 1 周是较好的选择.更进一步地,我们也可为不同的服务设计不同的监测频率,对于业务功能重要,客户愿意多付费的服务,可设置较高的监测频率;而对于不重要或不愿多付代价的服务,可设置较低的监测频率.

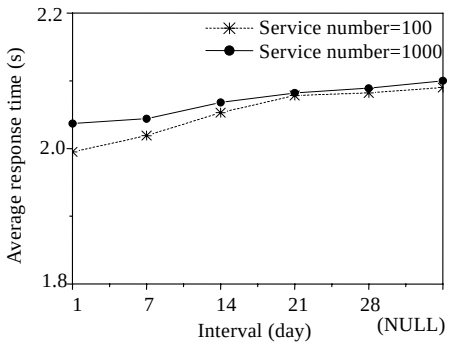

Fig.10 Average response time vs interval

图 10 平均响应时间比监测间隔

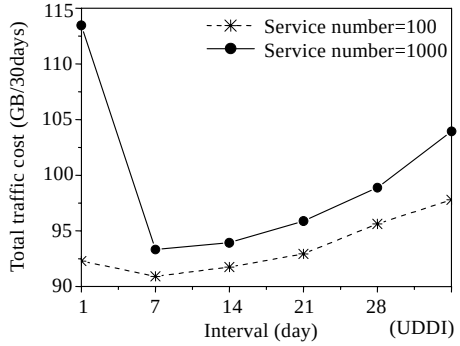

Fig.11 Total traffic cost vs interval

图 11 网络负载比监测间隔

第 4 组实验主要检验系统规模对 adUDDI 的影响,图 12 显示了平均响应时间随系统规模的增长而发生变化的情况,结果显示 adUDDI 在服务节点增加时,具有较好的可扩展性.而随着候选结果集的增大,系统的整体流量会缓慢增加(如图 13 所示).

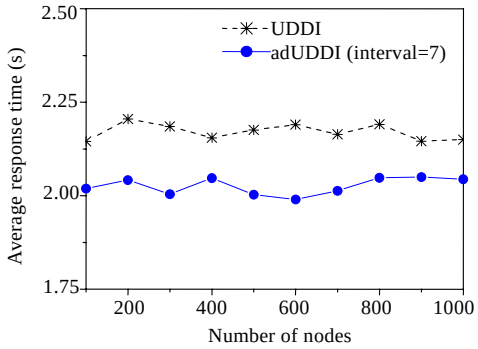

Fig.12 Response time vs number of services

图 12 平均响应时间比服务个数

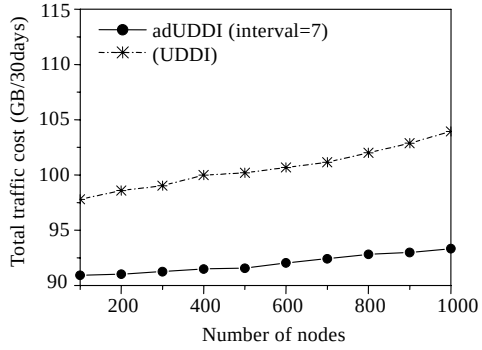

Fig.13 Total traffic cost vs number of services

图 13 网络负载比服务个数

4 结 论

针对目前公共 UDDI(UBR)的利用率低、服务信息有效性差等问题,我们提出了主动分布式 UDDI 结构,用于为服务请求者提供有效的服务信息.本文主要贡献包括:1) 在分布式结构中,根注册中心仅管理各行业 adUDDI,使其管理负担大幅下降,提高了根注册中心的实用性;2) 各行业 adUDDI 的建立及连接方式,为目前广

泛存在的私有或半私有 UDDI 的互连提供了基础,使得各服务孤岛可以方便地连接在一起;3) 主动监测机制使得从 adUDDI 中查询到的服务信息的实时有效性大幅提高.

更进一步地,Web 服务的有效性还包括 QoS 方面的管理监测等.如何在不大幅增加服务提供者及服务注册库负担的条件下更好地提高服务有效性,是我们下一步的主要工作.

References:

[1] Endrei M, Ang J, Arsanjani A, Chua S, Comte P, Krogdahl P, Luo M, Newling T. Patterns: Service_Oriented architecture and Web services. 2004. http://www.redbooks.ibm.com/ redbooks/pdfs/sg246303.pdf

[2] Christensen E, Curbera F, Meredith G, Weerawarana S. Web services description language (WSDL) 1.1. 2001. http://www.w3.org/TR/wsdl

[3] Clement L, Hately A, Riegen CV, Rogers T. Universal description discovery & integration (UDDI) 3.0.2. 2004. http://uddi.org/pubs/uddi_v3.htm

[4] Kim SM, Rosu MC. A survey of public Web services. In: Feldman SI, Uretsky M, Najork M, Wills CE, eds. Proc. of the 13th Int'l Conf. on the World Wide Web (WWW 2004). New York: ACM Press, 2004. 312−313.

[5] Kim SM. Population of public Web services. 2005. http://nclab.kaist.ac.kr/~smkim/ws_survey/index.html

[6] Cai M, Frank M. RDFPeers: A scalable distributed RDF repository based on a structured peer-to-peer network. In: Feldman SI, Uretsky M, Najork M, Wills CE, eds. Proc. of the 13th Int'l Conf. on World Wide Web (WWW 2004). New York: ACM Press, 2004. 650−657.

[7] Hong W, Lim M, Kim E, Lee J, Park H. GAIS: Grid advanced information service based on P2P mechanism. In: Livny M, ed. Proc. of the IEEE Int'l Symp. on High Performance Distributed Computing 2004 (HPDC 2004). Hawaii: IEEE Computer Society, 2004. 276−277.

[8] Xiao L, Zhang X, Xu Z. On reliable and scalable peer-to-peer Web document sharing. In: Ferreira A, Gottlieb A, eds. Proc. of the 16th Int'l Parallel and Distributed Processing Symp. (IPDPS 2002). IEEE Computer Society, 2002.

[9] Nagy W, Curbera F, Weerawaranna S. The advertisement and discovery of services (ADS) protocol for Web services. 2000. http://www-128.ibm.com/developerworks/library/ws-ads.html?dwzone=ws

[10] ShaikhAli A, Rana OF, Al-Ali RJ, Walker DW. UDDIe: An extended registry for Web service. In: Chang C, Murai J, eds. Symp. on Applications and the Internet Workshops (SAINT 2003). IEEE Computer Society, 2003. 85−89.

[11] Verma K, Sivashanmugam K, Sheth A, Patil A, Oundhakar S, Miller J. METEOR−S WSDI: A scalable P2P infrastructure of registries for semantic publication and discovery of Web services. Journal of Information Technology and Management, 2005,6(1):17−39.

[12] Oundhakar S, Verma K, Sivashanmugam K, Sheth A, Miller J. Discovery of Web services in a multi-ontology and federated registry environment. Int'l Journal of Web Services Research, 2005,2(3):1−32.

[13] Papazoglou MP, Kramer BJ, Yang J. Leveraging Web-services and peer-to-peer networks. In: Eder J, Missikoff M, eds. Proc. of the Advanced Information Systems Engineering, 15th Int'l Conf. (CAiSE 2003). LNCS 2681, Klagenfurt: Springer-Verlag, 2003. 485−501.

[14] GICS structure and sub-industry definitions. 2005. http://www.msci.com/equity/GICS_map2005.xls

[15] Medina A, Lakhina A, Matta I, Byers JW. BRITE: An approach to universal topology generation. In: Agrawal DP, ed. Proc. of the 9th Int'l Workshop on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS 2001). Cincinnati, 2001. 346−356.

杜宗霞(1975－),女,山西应县人,博士,讲师,主要研究领域为网络中间件,可信软件生产.

怀进鹏(1962－),男,博士,教授,博士生导师,CCF 高级会员,主要研究领域为计算机软件与理论,网络安全,网格计算.