第 33 卷 第 2 期 计 算 机 工 程 2007 年 1 月
Vol.33 No.2 Computer Engineering January 2007
·软件技术与数据库· 文章编号：1000—3428(2007)02—0053—03 文献标识码：A 中图分类号：TP311.12

一种基于分布式哈希表的 Web 服务目录系统

于守健[1]，朱 勤[1,2]，乐嘉锦[1]
(1. 东华大学计算机科学与技术学院，上海 200051；2. 南通大学计算机科学与技术学院，南通 226007)

摘 要：分析了集中式 UDDI 注册中心存在的缺点。结合 P2P 技术，基于分布式哈希表提供的高效的数据定位功能，提出了一种分布式 Web 服务目录系统，讨论了该系统下 Web 服务的发布与发现过程以及目录系统的维护。在该服务目录系统中，服务的描述信息分布在各个节点上，能够克服集中式 UDDI 注册中心的缺陷。

关键词：Web 服务；分布式哈希表；目录系统；UDDI

A Kind of Web Service Catalog System Based on Distributed Hash Table

YU Shoujian[1], ZHU Qin[1,2], LE Jiajin[1]
(1. College of Computer Science and Technology, Donghua University, Shanghai 200051;
2. College of Computer Science and Technology, Nantong University, Nantong 226007)

【Abstract】This paper analyzes limitations of typical UDDI registry. Utilizing peer-to-peer technology, it proposes a novel distributed Web services catalog service based on the efficient data location ability of distributed hash table. The paper discusses the procedure of Web service publication and discovery under this system, as well as the system evolution. This system enables distribution of description information to each peer, which will overcome the shortcomings of centralized UDDI registry.

【Key words】Web service; Distributed hash table; Catalog system; UDDI

在典型的 Web 服务架构中，集中式的 UDDI 注册中心被用来存储服务的描述信息。服务提供者定义服务的描述，并把描述信息发布到注册中心，然后，服务请求者就可以到注册中心中搜索适合自己的 Web 服务。集中式的注册中心能有效地保证注册到其中的所有服务均能被发现，但是该架构也不可避免地具有传统集中式系统的缺陷，诸如性能瓶颈和单点失败等问题。

从Web服务提供者来看，服务是分布在松散耦合的各个网络节点上，某些服务提供者，相对于另一些服务提供者而言，也是服务的请求者。从这一角度看，Web服务和P2P(peer to peer)计算环境具有较多的相似之处和共同特征[1]。因此，本文结合Web服务和P2P技术，提出了基于DHT的分布式Web服务目录系统。

1 集中式 UDDI 注册中心的缺点

集中式的 Web 服务注册中心会导致很多缺陷。当很多服务发布或发现请求同时被提交给注册中心服务器时，集中式的注册中心将会成为提供 Web 服务的瓶颈，导致与此同时很多其它服务提供者的发布请求或服务请求者的需求被闲置。即使当本地服务提供者能满足服务请求者的请求时，却仍然要进行不必要的全局服务查询，以决定绑定哪个服务提供者。

集中式注册中心另一个突出的问题是单点失败。UDDI 注册中心只有一个中心服务器，因此当该服务器发生故障时，便会造成整个 Web 服务系统的瘫痪。为了克服这些缺陷，一些 IT 厂商(如 Microsoft、IBM、HP 和 SAP)联合提出了建立备份服务器的解决方案。不过这种方法开销比较大，在 UDDI 用户有限的情况下，这种方法可以暂时改善性能。但是，随着服务数量的剧增，备份服务器越多，数据的一致性越难以保证。

此外，私有、半私有注册中心得到了广泛应用，这使得注册中心本身就是分布式的[2]。UDDI最初的目的是作为一个公共的Web服务目录，但是大量的电子商务应用和电子集市都是采用私有、半私有注册中心，每个注册中心用于特定应用领域服务的发布与发现。因此，服务的发布与发现不得不面对众多的服务注册库，在成百上千个注册库中搜索特定服务，变得异常困难，而且效率低下。采用将私有注册中心的内容复制到公共注册中心的方法，又难以保证数据的一致性。

2 基于 DHT 的分布式 Web 服务目录系统

当一个系统组合了几千个节点，如果能根据服务请求内容，选择出可以满足该请求的节点，并把请求直接发送给这些节点，那么就可以大大提高服务发现的效率。这需要有类似服务目录系统的支持，服务目录系统能根据查询的内容，决定哪个节点应该接受查询。本节介绍支持 DHT 的 Chord 协议，以此为基础，提出了基于 DHT 的分布式 Web 服务目录系统。

2.1 Chord 协议

近年来，许多研究者在设计可扩展的查找机制方面做了大量的研究工作，最新的成果是基于DHT的分布式查找和路由算法。Chord是一个著名的DHT协议[3]，它实现了这样一种

作者简介：于守健(1976－)，男，博士，主研方向：企业应用集成，Web 服务，数据库与数据仓库；朱 勤，博士生；乐嘉锦，教授、博导

收稿日期：2006-01-19 **E-mail：**jackyysj@dhu.edu.cn

操作：给定一个关键字(Key)，将Key映射到某个节点。给对等网络应用的每个数据都分配一个Key，Chord协议采用了相容哈希函数来为节点分配关键字。相容哈希有几个很好的特点：一方面，哈希函数可以做到负载平衡，也就是说所有的节点都可以接收到基本相同数量的关键字；另一方面，当第N个节点加入或者离开网络时，只有 1/N的关键字需要移动到另外的位置。图 1 说明了相容哈希函数的特点。

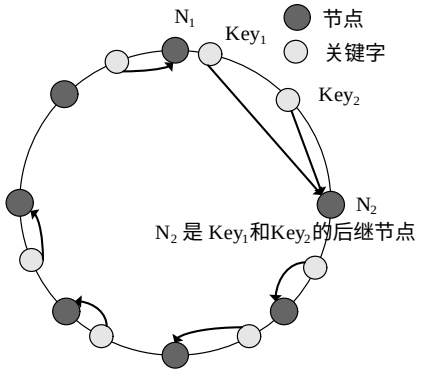

图 1 Chord 环

在Chord协议中，节点并不需要知道所有其它节点的信息。在由N个节点组成的网络中，每个节点只需要维护其它O(logN)个节点的信息。同样，每次查找只需要O(logN)条消息。当节点加入或者离开网络时，Chord需要传递$O(\log^2 N)$条消息来更新路由信息。

2.2 分布式 Web 服务目录系统

我们利用 Chord 技术，初步建立了一个完全分布式的 Web 服务目录系统，其方法是：Chord 协议根据每个服务提供者和服务请求者(节点)的 IP 地址，为其分配一个标识符，并将所有服务提供者和服务请求者连接到一起；服务提供者节点负责为发布的 Web 服务建立目录信息，然后 Chord 协议根据目录的关键字，将目录信息分配到相应的节点上。系统中的每个节点承担一部分服务目录信息，称之为本地服务目录。所有节点的本地服务目录构成全局服务目录系统，从而实现 UDDI 注册中心的功能。图 2 说明了分布式服务目录系统的架构。

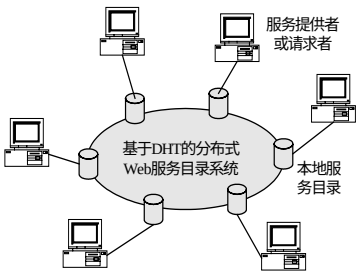

图 2 基于 DHT 的分布式 Web 服务目录系统

根据 WSDL 对 Web 服务的描述，我们将 Web 服务的名字作为待发布服务的关键字，而服务的操作以及操作的消息参数集作为与关键字相对应的摘要信息(Summary)。据此，服务提供者在发布服务时，建立如定义 1 所述的服务目录信息。

定义 1 Web 服务目录 C 具有这样的结构：C={(Key, Summary,N)}，其中 Key 表示从 Web 服务描述中抽取出的关键字，每个 Web 服务只有一个关键字；Summary 是与关键字 Key 相关联的摘要信息；N 是发布 Web 服务的节点。

基于 DHT 的分布对象定位机制，服务发现请求能够被直接发送到存储相关服务的本地服务目录系统中。然后在该目录系统中根据服务的摘要信息，再做进一步匹配，进而发现合适的服务。具体来讲，该目录系统具有如下优点：(1)目录存放的后继节点的确定，意味着相关 Web 服务信息的确定。Chord 协议查找后继节点的复杂度为 O(log n)，因而 Web 服务查询的复杂度也为 O(log n)；(2)目录中关键字的选定有很大的灵活性。Chord 协议的关键字可以是任意字符串，因此可以根据具体应用选择目录关键字，从而满足不同关键字的查询，这使得服务目录系统有较好的可扩展性和实用性；(3)目录的分配依赖于 Chord 协议，而 Chord 协议解决了网络中负载均衡的问题，因此服务目录系统不需要再考虑这一问题。

3 Web 服务的发布与发现

3.1 Web 服务发布

用N_i表示服务提供者，当N_i要发布服务时，首先为待发布的Web服务WS_j建立目录信息$C_{i,\ WSj}$ = {(Key_i,$Summary_i$, $_{WSj}$, N_i)}。然后，Chord协议根据服务关键字Key_i，将目录$C_{i,\ WSj}$分配到相应节点上。根据Web服务目录的定义，服务的名字应作为目录信息的Key，形如“ $op(para_1,para_2,\ldots,para_n)$ ”的操作以及消息参数集合，表示与Key相关联的Summary，其中包括了一个Web服务中所有操作的集合。

通过例子来进一步说明分布式 Web 服务目录下 Web 服务发布的过程。着眼于纺织企业应用，考虑 4 种服务：针织(Knitting)、机织(Weaving)、染整(Dying)和成衣加工(Slotwork)。表 1 中的 4 个服务提供者各自实现了其中的一部分服务。假设各服务提供者的服务名如表 1 中的“服务名”列所示，因此该列被作为服务目录的关键字。

表 1 服务提供者及其发布的服务

服务提供者	提供的 Web 服务	服务名
N_1	Knitting service	Knitting
	Weaving service	Weaving
N_2	Knitting service	Knitting
	Dying service	Dying
	Slopwork service	Slopwork
N_3	Dying service	Dying
	Slopwork service	Slopwork
N_4	Knitting service	Knitting
	Slopwork service	Slopwork

服务目录信息包括关键字和与关键字相关联的摘要信息，表 1 已给出了各个服务目录的关键字，因此还需要定义各个服务的摘要，也就是服务的操作以及操作的消息。表 2 说明了服务提供者N_1和N_2的Knitting服务的摘要信息，即操作以及操作的消息列表。

表 2 服务提供者N_1和N_2的Knitting服务中的操作

N_1的操作及消息参数	N_2的操作及消息参数
SearchByName(Name; Price)	SearchByPrice(Price)
GetDetail(No;Color, Weight,Price)	Delivery(LotNo)
Purchase(LotNo)	

在定义了服务描述的关键字和摘要之后，就可以定义服务的目录信息。以N_1的Knitting service服务为例，其目录$C_{1,Knitting}$={Key_1,$Summary_{1,Knitting}$, N_1}，其中：Key_1＝Knitting，$Summary_{1,Knitting}$＝{SearchByName(Name;Price), GetDetail(No; Color,Weight,Price), Purchase(LotNo)}。根据DHT的哈希算法，假定得到如表 3 表示的服务目录系统。

表 3 各节点上的服务目录信息

各节点上的服务目录信息	
N_1	Slopwork {$S_{2,\ Slopwork}$, $S_{3,\ Slopwork}$, $S_{4,\ Slopwork}$}
N_2	Knitting {$S_{1,\ Knitting}$, $S_{2,\ Knitting}$, $S_{3,\ Knitting}$}
N_3	--
N_4	Dying {$S_{2,\ Dying}$, $S_{3,\ Dying}$}; Weaving {$S_{1,\ Weaving}$}

从表 3 可以看出，关键字Knitting及其摘要信息被哈希到节点N_2，关键字SlotWork及其摘要信息被哈希到节点N_1，关键字Dying、Weaving及其摘要信息被哈希到节点N_4。每个节点存储部分服务目录信息，整个系统节点的服务目录构成了分布式Web服务注册中心。

3.2 Web 服务发现

服务发现与服务发布是互逆的过程。为了能够准确发现服务，服务请求者首先应根据目的服务的功能，定义目的服务的名字、操作以及消息参数列表，从而创建目的服务的目录信息。服务发现过程可以用 4 个步骤来概括：

(1)Chord 协议的哈希算法，根据用户提供的关键字，计算存储该关键字及其摘要信息的节点；

(2)Chord 协议根据各个节点的路由表，定位该节点；

(3)把服务发现请求转发至该节点。然后，根据查询请求的摘要信息，与该节点的本地服务目录进行匹配，找到合适的服务后，返回发布该服务的节点；

(4)服务请求者直接与该节点联系，获取 Web 服务的技术文档，进而调用 Web 服务。

图 3 给出了Web服务发现的一个例子。假设节点N_3是服务请求者，Knitting作为查询的关键字。经过哈希算法计算，关键字Knitting的服务目录信息位于节点N_2上，因此服务请求被转发至节点N_2。经过服务请求与节点N_2本地服务目录的匹配，假设仅有$S_{1,\text{Knitting}}$满足查询条件，N_1是唯一满足服务请求的节点，因此N_2返回节点N_1给N_3。最后N_3直接与N_1联系，获得进一步的细节信息，进而调用该Web服务。

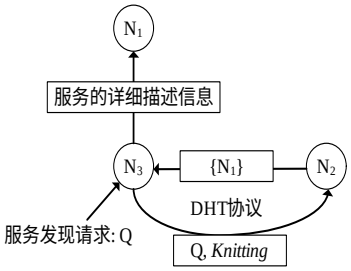

图 3 Web 服务发现过程

4 Web 服务目录系统的维护

4.1 节点加入

当有新节点加入系统时，服务目录信息不需要修改。但是因为部分关键字的后继节点发生变化，一些目录需要从原来的节点移动到新的节点之上。另外，新节点创建的目录，也需要根据Chord协议被分配到系统中。假设节点N_m要加入系统，N_m为待发布的Web服务建立服务目录信息$C_m=\{(K_{m,i}, S_{m,i}, N_m)\}$。$N_m$加入系统具体包括 3 步工作，如图 4 所示。

(1)N_m与网络中任意节点N_c联系，Chord协议计算出标识环上N_s是N_m后继节点，N_m加入系统；

(2)N_m成为标识环上的一个节点之后，它将服务目录中的信息依照关键字分配到其它节点，Chord协议计算新的服务目录信息存储于哪些节点；

(3)N_m作为系统的一个节点，有义务分担一部分服务目录信息。因此，Chord对原来N_s节点上的服务目录信息进行重新定位计算，相应目录信息被转移至N_m。以图 4 为例，在N_m加入系统之前，以K_1、K_2、K_3为关键字的目录信息位于节点N_s上。N_m加入系统后，经过Chord计算，关键字K_1、K_2的后继节点应为N_m，因此K_1、K_2及其摘要信息从节点N_s转移到新节点N_m。

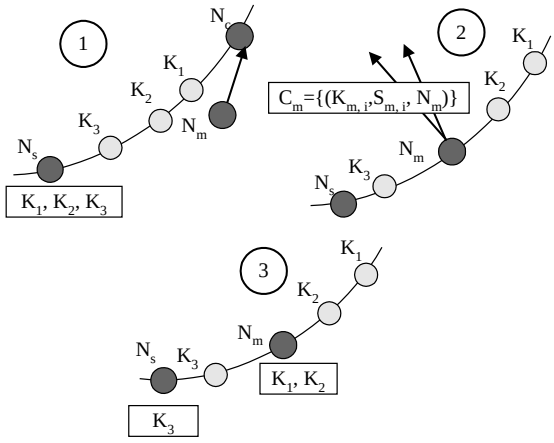

图 4 节点加入时目录系统的维护

4.2 节点离开与更新

当一个节点N_m离开系统时，根据Chord协议，N_m首先要把在它上面存储的服务目录信息转移给N_m的后继节点N_s。然后，它要通知存储节点N_m所发布服务目录信息的节点，将N_m所发布的服务目录信息删除。这些节点的确定，可以根据图 4 中步骤②计算每个关键字的后继节点的方法得出。

当服务提供者提供的服务发生改变时，服务的描述信息也需要做相应改变。这导致服务目录信息系统中的 Key 和 Summary 发生相应变化，因此需要更新网络中相应服务的目录信息。更新操作与节点加入操作的第 2 步相似，可以采用先删除原来的目录信息，再将新的服务目录信息，依照其关键字发布到其它节点。

5 结论

本文结合 P2P 技术，提出了基于 DHT 的分布式 Web 服务目录系统。系统中每个节点都承担一部分本地 Web 服务目录信息，汇集所有的本地 Web 服务目录，实现了传统 UDDI 注册中心的功能，从而克服了集中式 UDDI 注册中心的缺陷。随着 Web 服务成为下一代 Web 的主导技术，本文的 Web 服务组织方法将为各个商业组织提供更加灵活的Web服务发现形式，进而实现在完全分布式环境下的 Web 服务组合。

参考文献

1 Schattkowsky T, Loeser C, Müller W. Peer-to-Peer-based Web Services for Collaborative Engineering Environments[C]. Proceedings of the 3rd International Conference on Networking, Guadeloupe, France, 2004-03.

2 The Stencil Group. The Evolution of UDDI[Z]. 2002-07. http://www.uddi.org/pubs/the_evolution_of_uddi_20020719.pdf.

3 Stoica I, Morris R, Karger D, et al. Chord: A Scalable Peer-to-Peer Lookup Protocol for Internet Applications[J]. IEEE/ACM Transactions on Networking, 2003, 11(1): 17-32.