Journal of Computer Applications
计算机应用,2014,34(7):1983 - 1987
ISSN 1001-9081
CODEN JYIIDU
2014-07-10
http://www.joca.cn

文章编号:1001-9081(2014)07-1983-05
doi:10.11772/j.issn.1001-9081.2014.07.1983

基于对等架构的 Web 服务注册系统

龙运坚[1],何 倩[1,2*],王 勇[1],王小峰[2]
(1. 认知无线电与信息处理教育部重点实验室(桂林电子科技大学),广西 桂林 541004; 2. 国防科学技术大学 计算机学院,长沙 410073)
(* 通信作者电子邮箱 treeqian@gmail.com)

摘 要: 针对传统的集中式架构的 Web 服务注册系统存在性能瓶颈、单点故障的问题,设计并实现了基于结构化对等协议的 Web 服务注册系统。系统包括配置、调度分发、对等通信、权限验证、JUDDI 和网络资源监控等六大模块,提出了基于 Pastry 的服务注册系统调度和通信算法,设计了相应的 Web 服务注册和发现的流程,采用 SoapUI 和 LoadRunner 对系统进行测试。实验结果表明,该系统能够支持大规模服务请求,具有动态可扩展性,在模拟多并发实验中,服务注册和发现的响应速度都提高了 1 倍。

关键词: Web 服务; 服务注册; 统一描述、发现和集成; 结构化 P2P; Pastry
中图分类号: TP393 **文献标志码:** A

Peer-to-peer based Web service registration system

LONG Yunjian[1], HE Qian[1,2*], WANG Yong[1], WANG Xiaofeng[2]
(1. *Key Laboratory of Cognitive Radio and Information Processing of the Ministry of Education (Guilin University of Electronic Technology), Guilin Guangxi* 541004, *China*;
2. *College of Computer, National University of Defense Technology, Changsha Hunan* 410073, *China*)

Abstract: Since the traditional centralized architecture Web service registry system suffers from such problems as performance bottleneck, single-point-of-failures, a structure Peer-to-Peer (P2P) based Web service registry system was designed and implemented. The registry system consists of six modules including configuration, schedule and distribution, peer-to-peer communication, rank validation, JUDDI, and network resources monitoring. The pastry based system scheduling and communication algorithms were proposed, and the corresponding Web service registration and discovery process was designed. The Web service registration system was tested and analyzed using SoapUI and LoadRunner. The experimental results show that the system can support large-scale accessing and has dynamic scalability. In the multi-concurrent simulation experiments, the response speeds of Web services registration and discovery are increased 1 times.

Key words: Web service; service registration; Universal Description Discovery and Integration (UDDI); structured Peer-to-Peer (P2P); Pastry

0 引言

面向服务体系架构(Service-Oriented Architecture,SOA)是继面向对象、基于构件开发之后的一种新型软件开发、部署和集成模式,为软件开发提供了灵活的设计和开发方案,是云计算对外提供访问的主要方式[1]。SOA 主要由服务提供者、服务注册中心和服务请求者三个角色组成。在整个 SOA 的架构当中,服务注册中心是实现服务组合的基础。统一描述、发现和集成(Universal Description Discovery and Integration,UDDI)是 Web 服务相关信息的描述规范,同时也包含了 Web 服务信息注册中心的标准规范。

传统的服务注册系统是遵从 UDDI 规范的私有服务注册库,采用的是集中式的架构。文献[2-3]实现了基于 UDDI 的单一服务注册中心。为了满足不同的应用需求,文献[4]通过对传统的 UDDI 注册中心进行扩展,以支持 Web 服务图。然而传统的服务注册系统,均存在集中式架构系统共有的性能瓶颈、单点失效且不易扩展等缺点,无法适应大规模的服务注册和查询。对此,有研究者提出引入分布式架构,利用分布式架构减轻注册中心的负担,提高系统效率[5-7]。

随着点对点(Peer-to-Peer,P2P)技术的发展,P2P 网络能够为各私有服务注册库之间提供相应的互通机制,可以避免服务孤岛等问题,利用 P2P 网络的优点来解决集中式架构的性能瓶颈以及单点失效等问题已经成为研究趋势。文献[8]针对非结构化 P2P 网络,设计了一种称为 I-Wander 的 Web 服务发现方法,提高了 Web 服务查找的效率。然而,非结构化 P2P 网络的服务查询机制通过洪泛完成,具有一定的盲目性,占用大量的网络带宽。针对非结构化 P2P 网络存在路由效率低、可扩展性差、负载不均衡等问题,文献[9]提出基于结构化 P2P 网络的服务发现方法,利用结构化的 P2P 系统在节点间使用信息路由来代替这种洪泛机制。

本文拟在 JUDDI[10] 系统上进一步扩展,引入 Pastry 结构化 P2P 网络技术用来组织、协调多个 JUDDI,以解决性能瓶

收稿日期:2014-01-20;修回日期:2014-02-26。 基金项目:国家自然科学基金资助项目(61201250,61163057,61163058);国家科技支撑计划项目(2012BAH18F00);广西自然科学基金资助项目(2012GXNSFBA053174,2012GXNSFAA053230)。

作者简介:龙运坚(198 -),男,湖南邵阳人,硕士研究生,主要研究方向:通信软件、分布式计算; 何倩(1979 -),男,湖南安仁人,教授,博士,CCF 会员,主要研究方向:分布式计算、信息安全; 王勇(1964 -),男,四川阆中人,教授,博士,主要研究方向:计算机网络、信息安全、计算智能; 王小峰(1982 -),男,助理研究员,博士,CCF 会员,主要研究方向:网络安全、分布式计算。

颈、单点故障等问题。所实现的基于 Pastry 的 Web 服务注册系统(PUDDI)可以实现动态可扩展的服务注册,PUDDI 系统通过 SoapUI 和 LoadRunner 验证和分析了其扩展性。

1 基于 P2P 的服务注册系统框架

在 UDDI 注册中心引入 Pastry 这种结构化 P2P 协议,根据 P2P 协议的特点,在系统伸缩性设计上,节点的数目可以任意扩大,系统伸缩性好可扩展性强,系统不因节点的扩大而降低性能,具有良好的稳定性;在网络的负载均衡性能上,Pastry 网络上的每个节点只保存属于自己管理的资源,网络上的服务被大致均匀地分发到网络上的各个节点,有利于网络的负载均衡,增大系统吞吐流量,支持更多的并发用户量访问;在系统可靠性上,当有大量服务请求集中于某一节点或者某节点突然失效时,服务请求自动被 Pastry 网络转发到相邻节点,避免单点失效问题。基于以上诸多的优点,我们引入 Pastry 协议,利用 Pastry 网络的优点来提高系统的动态可扩展、负载均衡等性能。基于 Pastry 的 Web 服务注册系统网络架构图如图 1 所示。服务请求者可以向 PUDDI 上的任意 UDDI 节点发出查询请求,根节点是创建 Pastry 的第一个节点,节点和节点之间基于 Pastry 协议实现路由和通信。

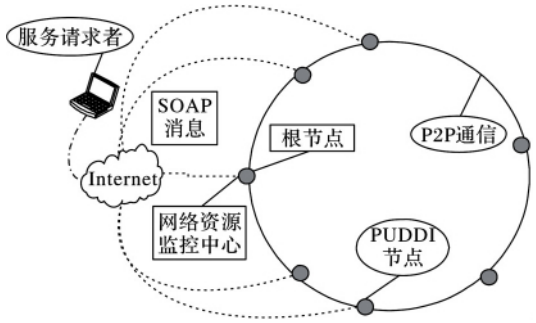

图 1 PUDDI 系统网络架构

PUDDI 系统的功能模块主要包括配置、调度分发、对等通信、权限验证、JUDDI、网络资源监控六大模块,模型如图 2 所示。

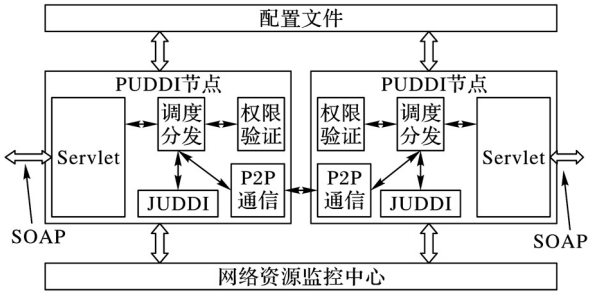

图 2 系统功能架构

1) 配置模块。每个节点部署相同的应用程序,根据配置参数的不同,其中一个节点为根节点,发起 Pastry 环,其他节点相继加入。节点可以根据需求动态加入,实现系统动态可扩展。Pastry 的启动,首先通过调用 PastryIdFactory,对公钥进行哈希,如果没有公钥就对 IP 地址进行哈希,得到 nodeId。

2) 调度分发模块。该模块主要用于拦截服务请求消息,控制操作流程。该模块是实现 JUDDIV3 和 Pastry 算法相结合的关键,负责判断对消息是否需要转发到其他的节点。

3) P2P 通信模块。它包括发送模块和接收模块。发送模块主要是转发客户端发起的简单对象访问协议(Simple Object Access Protocol,SOAP)请求,调用 route 方法发送消息,由 Pastry 的路由机制定位到保存该服务的节点上。接收模块主要是接收其他节点发过来的消息,并作相应的处理,当收到属于本节点管理的消息时,Pastry 将会调用 deliver 方法。

4) 权限验证模块。Web 服务认证是 Web 服务访问控制和安全调用的前提[11],PUDDI 系统的访问控制和安全调用由权限验证模块来保证,该模块对用户身份和操作进行安全认证,若用户没有被赋予相关的权限,则终止操作,返回提示的错误信息;若权限认证通过,则继续相关的操作。

5) JUDDI 模块。JUDDI 模块主要是节点服务注册查询,保存服务注册信息,提供查询。服务提供者可在 JUDDI 上发布服务,而服务请求者可在 JUDDI 发现所需的服务。

6) 网络资源监控中心。系统监控的主要功能是监控服务在整个 P2P 网络上的服务分布情况。各个节点采用 socket 通信,将自身保管的服务情况通知根节点。

2 基于 Pastry 的系统调度与通信算法

系统调度算法和系统通信算法是实现 JUDDIV3 和 Pastry 结合的关键算法。调度算法负责分发任务,控制流程,协调多个 UDDI 节点的服务请求。系统通信算法是实现多个 UDDI 节点之间交换服务信息的关键所在,是 UDDI 节点之间的纽带。

2.1 Pastry 协议

Pastry 是微软研究院提出的完全分布式的、可扩展的、自组织的结构化 P2P 协议。与其他结构化 P2P 算法相比,Pastry 考虑了网络位置信息,根据类似于 IP 路由的 hop 数的标量距离度量,尽量减少消息传输的距离[12],用于协调多个 UDDI,将有利于提高服务请求者选择较近的 UDDI 访问。

节点在加入 Pastry 网络时随机分配一个唯一的节点标识(NodeID),每个节点都需要维护 3 个状态表:叶子节点集(L),路由表(R)和邻居节点集(M)。当发送消息时,其路由机制是:节点首先检查消息关键值 K 是否在叶子节点集合 L 范围内,如果是,则直接把查询消息发给 L 中节点标识和 K 值最接近的节点,否则就从路由表 R 中根据最长前缀优先的原则选择一个节点作为路由目标,转发消息。如果不存在这样的节点,当前节点将会从其维护的所有邻居节点集合 M 中选择一个距离 K 最接近的节点作为发送消息的目标。详细介绍参见文献[12]。

2.2 系统调度算法

系统调度算法具体流程是:首先判断消息来源,如果是 P2P 网络发过来的服务请求消息,则直接调用本地的 JUDDI 接口,进行相应的服务注册查询操作。如果是 P2P 网络发过来的回复消息,则将消息返回给客户端。如果是客户端直接发起的 SOAP 消息,则进行权限认证,如果认证不通过,则返回错误提示消息给客户端。如果认证通过,查询缓存服务中有该服务则直接返回该服务,否则取出消息中的信息(如 *tmodelName*),由 SHA-1 算法生成消息关键字 K,将 K 与 UDDI 节点 ID 进行比较,如果服务由本 UDDI 节点管理,则直

接调用本地的 JUDDI 接口，如果不是，则调用 Pastry 的通信接口，进行转发消息。等待消息回复，如果收到回复消息，则返回给客户端，如果等待超时，则返回超时提示信息，如果还没有超时，则继续等待。

系统调度算法具体流程的伪代码描述如下，为了方便介绍，给出如下记号:

K: 服务请求消息关键字。*nodeId*: 本节点 ID 号。

```
1)  if isPastrySentMessage then          //P2P 转发来的请求消息
2)    Invoking JUDDI API;                //调用本地 JUDDI 接口
3)  if isPastryBackMessage then          //服务回复消息
4)    if timeout then { Putting into register; }
5)    else { Return to client; }
    else                                 //客户端的服务请求
    {
6)  if isPassed {                        //权限认证通过
7)    if storedInRegister then           //缓存服务中有该服务
8)      Return storedMessage;
      else
    {           //取出消息中 tmodelName，由 SHA-1 算法生成 K
9)    K←SHA-1( Message. tModelName);
10)   if mostSimilar then
      // K 与 nodeId 最相近，服务由本 UDDI 节点管理
11)     Invoking JUDDI API;
      else                               //服务由其他本节点管理
      {
12)     Create PastrySentMessage;        //创建消息
        //调用 Pastry 的通信接口，发送消息;
13)     Invoking Pastry API;
14)     Wait;                            //等待一段时间
15)     if receivedPastryBackMessage then
16)       Return to client;
17)     else if timeout then
18)       Return timeoutMessage;
19)     else Goto 14);
      }
    }
  }
    else                                 //权限认证没有通过
20)   Return error; }
```

2.3　系统通信算法

系统通信算法的具体流程是: 首先将需要发送的服务消息存入消息缓存器中，选择管理器根据消息关键字 K 与 UDDI 节点 ID 进行比较匹配，如果在与本节点相近的 UDDI 节点集合中存在有 UDDI 节点 ID 比本节点 ID 最接近 K，则转发到该 UDDI 节点，如果发送成功且有 ACK 确认回复，则转向下一个服务消息的发送，否则重新发送。如果在与本节点相近的 UDDI 节点集合中找不到 UDDI 节点 ID 比本节点 ID 最接近 K，则计算本节点 ID 号与消息关键字具有公共前缀的长度 m，如果在路由表 m 行 i 列(i 是 K 的第 m 位数值) 中存在 UDDI 节点 ID，则转发到该 UDDI 节点，如果不存在，则转发到路由表中公共前缀长度为 m，且比本节点更接近 K 的 UDDI 节点。

系统通信算法具体流程的伪代码描述如下，为了方便介绍，给出如下记号:

K: Pastry 消息的关键字; *nodeId*: 本节点 ID 号。

R_l^i: 路由表中 l 行($0 \leqslant l < 128/b$) 第 i 列($0 \leqslant i < 2^b$) 的 UDDI 节点。

L_i: 与本节点相近的 UDDI 集合 L 中第 i 个 UDDI 节点($-|L|/2 \leqslant i \leqslant |L|/2$) 。

K_l: 服务请求消息关键字 K 的第 l 位数值。

```
1)  Put Message into buffer;             //存入缓存器
2)  SelectorManager. Match( K )          //选择管理器匹配消息关键字
3)  if L_{-|L|/2} ≤ K ≤ L_{|L|/2} then
      //K 在与本节点相近的 UDDI 集合中
    {     //直接转发到相近 UDDI 节点集合中更接近 K 的节点
4)    Forward UDDI_L,( UDDI_L ∈ L ∩ min( |L_i - K| ) );
5)    if ACK == true then Goto 1);       //有确认回复，转 1)
6)    else Goto 2);
    }
    else                                 //使用路由表发送
    {
7)    l = prefix( K,nodeId);             // 公共前缀长度
8)    if R_l^{Kl} ≠ NULL then            //该 UDDI 节点存在
      {
9)      Forward UDDI_R,( UDDI_R == R_l^{Kl} );
10)     if ACK == true then Goto 1);
11)     else Goto 2);
      } else                             //该 UDDI 节点不存在，或者不可到达
      {
      /* 转发到路由表中公共前缀长度为 l，且比本节点更接近 K 的 UDDI 节点* /
12)   Forward UDDI_R,( UDDI_R ∈ R ∩ min( |R_l^i - K| <
          |nodeId - K|) );
13)     if ACK == true then Goto 1);
14)     else Goto 2);
      }
    }
```

以 NodeID 为 74BA2F 的节点发送 K 为 D617FA 的消息为例，经系统通信算法，其消息转发过程如图 3 所示。消息在节点间的每一次转发将更加接近目标节点，最终被转发到 NodeID 与 K 值最接近的节点，即 NodeID 为 D615AB 的节点上。

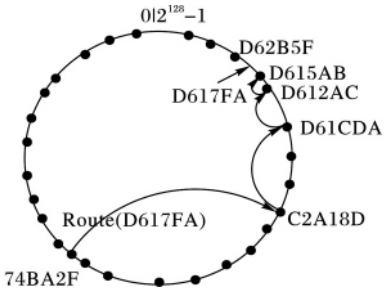

图 3　系统消息路由

3　基于 PUDDI 的 Web 服务注册与发现

服务提供者提供服务的具体实现，同时负责将服务发布到注册中心。服务请求者可在注册中心发现服务描述，并能获取绑定信息，利用该绑定信息，可以绑定到服务提供者，从而调用服务提供者提供的服务。

服务提供者向 PUDDI 注册服务的具体步骤为:

1) 服务提供者读取服务的 Web 服务描述语言(Web Services Description Language，WSDL) 文件的统一资源定位符(Uniform Resource Locator，URL) ，并解析该 WSDL 文件，生成服务注册时需要的相关信息(如，绑定信息等) 。

2) 生成相关的 SOAP 消息，通过 http 请求，发送到 P2P 网

络上某个已知节点上。

3) 该节点接收 SOAP 消息后，由调度分发模块转给权限认证模块，对服务操作进行权限认证；调度分发模块根据权限认证模块返回的结果判断，若权限认证没有通过，则返回提示的错误信息；若权限认证通过，则继续往下执行。

4) 解析信息中的服务相关信息，通过 SHA-1 算法生成散列值；节点根据该散列值，判断是否为本节点所管理，若是，则跳步骤 f；若不是，则将该 SOAP 消息进行 Pastry 协议封装，转交给 P2P 通信模块，转发到属于管理该散列值的节点上。

5) P2P 通信模块根据 Pastry 的路由机制，最终将消息转发到管理该服务的节点上。P2P 通信模块将消息转交给调度分发模块进行处理。

6) 调用 JUDDI 的注册应用程序编程接口(Application Programming Interface，API)，将服务的 WSDL 文件的解析信息以及一些相关信息注册到本节点的 JUDDI 中。

7) 将注册的结果信息返回给客户端。

服务注册流程，如图 4 所示。服务请求者向 PUDDI 查询发现服务步骤与注册的步骤类似，只是发起的请求消息不同，在此不再描述介绍。

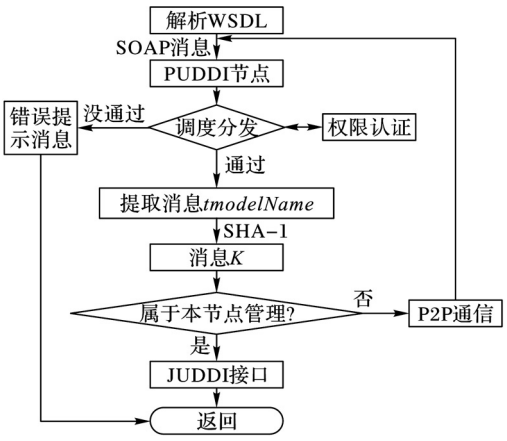

图 4 服务注册流程

4 实验与分析

该实验在百兆以太网的网络环境中进行测试，包括 3 台物理主机和 3 台虚拟机。物理主机是联想万全 R510 服务器(Intel 4 核 2.80 GHz 处理器、1 GB 内存)，虚拟机部署在 1 台曙光 W5801-G10 服务器(2 个 Intel 12 核 2.30 GHz 处理器、64 GB内存)上，分配 3 台虚拟机(单 CPU、1 GB 内存)。操作系统均采用 centos 6.2，软件环境有 jdk 1.6 的 Java 平台、mysql5.5 数据库、Apache Tomcat6 Web 容器。

使用 SoapUI [13] 对 PUDDI 系统进行功能测试。对于 Web 服务注册，以 save _ Tmodel 为例，注册成功后，会返回 tModelDetail 的 SOAP 消息，该消息中包含了刚刚注册的信息以及 tModelKey。对于 Web 服务查询，服务请求者查询到服务的 WSDL 的地址，然后再根据该 WSDL 生成客户端，从而可以调用该服务，甚至可以组合成新的服务。UDDI BindinTemplate 中的 accessPoint 元素包含了服务的 WSDL 的地址。以 find_binding 为例，查询服务的 WSDL 的地址。实验中对已经注册好了的服务进行查询，利用 find_binding 嵌套了 find_tmodel 进行复合查询，返回消息中的 accessPoint 元素包含了该服务的 WSDL 地址(如：http://buy.96511.com/Lottery/WebService/MultiBuyService.asmx?wsdl)，则查询服务成功。另外，对 PUDDI 系统还测试了节点发生故障的情况下，系统功能仍然正常，所提供的服务注册和发现的功能将可以自动地由相邻节点接管。

衡量一个软件系统性能的常见指标有：响应时间、吞吐量、并发用户数等。其中响应时间就是用户感受软件系统为其服务所耗的时间，包括服务端响应时间、网络响应时间以及客户端响应时间。响应时间能够直观地反映系统的处理能力，因此，本文根据响应时间来衡量 JUDDIV3 和 PUDDI 系统的性能。下面使用 LoadRunner [14] 分别对 JUDDIV3 系统和部属 6 个节点的 PUDDI 系统进行服务注册和发现的性能测试。

4.1 服务注册

针对原始的 JUDDIV3 和 6 个节点的 PUDDI 系统，用 LoadRunner 模拟多个用户进行服务注册，每 10 秒新增 5 个用户，直到达到 100 个并发用户，持续运行一段运行时间。JUDDIV3 和 PUDDI 的服务注册平均响应时间如图 5 和表 1 所示。由图表可知，随着并发用户的增加，服务注册响应时间也随之增大，但 JUDDIV3 系统的响应时间增大的幅度远远大于 PUDDI 系统。JUDDIV3 系统比 PUDDI 系统的注册响应时间时间多两倍有余。PUDDI 系统标准差比 JUDDIV3 系统小得多，这说明 PUDDI 系统受用户增加的影响小，系统更稳定。

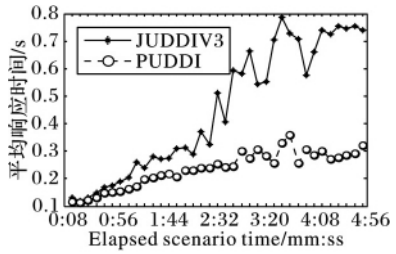

图 5 平均响应时间对比

表 1 注册响应时间对比 s

测试对象	平均值	最小值	最大值	中间值	标准差
JUDDIV3	0.451	0.120	0.787	0.405	0.230
PUDDI	0.234	0.113	0.360	0.243	0.064

4.2 服务发现

在 JUDDIV3 系统和 PUDDI 系统服务相同，且已经注册了 2 000个不同的服务情况下，模拟 100 个并发用户，每个用户查询的服务都是随机的。每 10 秒启动 5 个并发用户，直到达到 100 个并发用户，持续运行一段时间。JUDDIV3 平均响应时间和 PUDDI 系统平均响应时间对比关系如图 6 和表 2 所示。

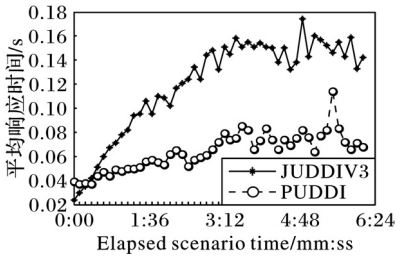

图 6 查询响应时间对比

从实验结果可以发现，随着并发用户的增加，服务发现响

应时间也随之增大,但 JUDDIV3 系统的服务发现时间几乎比 PUDDI 多两倍。PUDDI 系统标准差比 JUDDIV3 系统小得多,这说明 PUDDI 系统受用户增加的影响小,系统更稳定。

表 2　查询响应时间对比　　s

测试对象	平均值	最小值	最大值	中间值	标准差
JUDDIV3	0.120	0.024	0.174	0.134	0.039
PUDDI	0.064	0.037	0.114	0.066	0.015

5　结语

海量的 Web 服务管理需要一个可扩展的分布式服务注册系统。本文设计了基于结构化 P2P 的服务注册系统,提出了基于 Pastry 的服务注册系统调度和通信算法。系统在高并发情况下,提高了服务注册和发现的效率,可以解决单台注册系统带来的性能瓶颈和单点故障问题,能够适应大规模 Web 服务的注册和访问,可以作为云计算服务环境的重要支撑,具有很好的应用前景。下一步我们将在 PUDDI 系统的基础上进行语义扩展,实现 Web 服务的模糊查询。

参考文献:

[1] REN Q. Research on verification of Web service based on abstraction refinement and combination technology[D]. Suzhou: Soochow University,2011. (任强. 基于谓词抽象与精化技术的 Web 服务验证研究[D]. 苏州: 苏州大学, 2011.)

[2] WANG M. Research and implementation of a highly avaliable Web services platform[D]. Xi'an: Xidian University, 2012. (王萌. 一种高可用 Web 服务平台的研究与实现[D]. 西安: 西安电子科技大学, 2012.)

[3] LIU J X, LIU J, CHAO L. Design and implementation of an extended UDDI registration center for Web service graph[C]// ICWS 2007: Proceedings of the 2007 IEEE International Conference on Web Services. Piscataway: IEEE, 2007: 1174 - 1175.

[4] TAMILARASI K, RAMARKRISHNAN M. Design of an intelligent search engine-based UDDI for Web service discovery[C]// ICRTIT 2012: Proceedings of the 2012 International Conference on Recent Trends In Information Technology. Piscataway: IEEE, 2012: 520 - 525.

[5] YAO Y, CAO J, LIU B, *et al*. Scalable mechanism for semantic Web service registry and discovery[J]. Journal of Southeast University: Natural Science Edition, 2010, 40(2): 264 - 269. (姚燚, 曹玖新, 刘波, 等. 基于语义的可扩展 Web 服务注册与发现机制[J]. 东南大学学报: 自然科学版, 2010, 40(2): 264 - 269.)

[6] GUO M-Q, HUANG Y, LUO X-G, *et al*. Design and implementation of a distributed Web service directory in SOA-oriented urban spatial information sharing platform[C]// PACIIA 2009: Proceedings of Asia-Pacific Conference on Computational Intelligence and Industrial Applications. Piscataway: IEEE, 2009, 1: 127 - 130.

[7] DU Z, HUAI J. Research and implementation of an active distributed Web service registry[J]. Journal of Software, 2006, 17(3): 454 - 462. (杜宗霞, 怀进鹏. 主动分布式 Web 服务注册机制研究与实现[J]. 软件学报, 2006, 17(3): 454 - 462.)

[8] ZHANG C, CAO Y, LIU D, *et al*. I-Wander: A Web service discovery method for unstructured P2P network[J]. Transactions of Beijing Institute of Technology, 2008, 26(6): 521 - 525. (张常有, 曹元大, 刘丹, 等. I-Wander: 一种面向非结构化对等网络的 Web 服务发现方法[J]. 北京理工大学学报, 2008, 26(6): 521 - 525.)

[9] SIOUTAS S, SAKKOPOULOS E, MAKRIS C, *et al*. Dynamic Web service discovery architecture based on a novel peer based overlay network[J]. Journal of Systems and Software, 2009, 82(5): 809 - 824.

[10] JUDDI[EB/OL]. [2014-01-17]. http://juddi.apache.org/.

[11] HE Q, WANG F, CHAI H, *et al*. Survey of Web services authentication technology[J]. Journal of Guilin University of Electronic Technology, 2013,33(3): 246 - 252. (何倩,王芳,柴华昕,等. Web 服务认证技术综述[J]. 桂林电子科技大学学报,2013,33(3): 246 - 252.)

[12] ROWSTRON A, DRUSCHEL P. Pastry: Scalable, decentralized object location, and routing for large-scale peer-to-peer systems[C]// Middleware 2001, LNCS2218. Berlin: Springer, 2001: 329 - 350.

[13] LUO Z, ZHU Y, CHENG M. Web service testing tool SOAPUI and its analysis[J]. Computer Applications and Software, 2010, 27(5): 155 - 157. (罗作民, 朱燕, 程明. Web 服务测试工具 SOAPUI 及其分析[J]. 计算机应用与软件, 2010, 27(5): 155 - 157.)

[14] Mercury Interactive, LoadRunner[EB/OL]. [2014-01-10]. http://www.mercury.com/us/products/loadrunner/

(上接第 1977 页)

[9] YANG X, XIONG Y, JIA G. Fast acquisition of primary synchronization signal in LTE system [J]. Journal of Applied Sciences, 2012, 30(1): 14 - 18. (杨秀梅,熊勇,贾国庆. 一种 LTE 系统的主同步信号快速捕获方法 [J]. 应用科学学报, 2012,30(1): 14 - 18.)

[10] KIM Y B, CHANG K H. Complexity optimized CP length predecision metric for cell searcher in the downlink of 3GPP LTE system [C]// Proceedings of the 2009 Personal Indoor and Mobile Radio Communications. Piscataway: IEEE, 2009: 895 - 899.

[11] LIN Y, YUE G, YIN C. Improvement of CP type decision and SSS detection algorithm in cell search for LTE system [J]. Journal of Jilin University: Engineering and Technology Edition, 2012, 42(2): 499 - 504. (林雁,乐光新,尹长川. LTE 小区搜索中循环前缀类型判决和辅同步信号检测算法的改进 [J]. 吉林大学学报: 工学版,2012,42(2): 499 - 504.)

[12] 3GPP TS 36.211 v9.1.0. Evolved Universal Terrestrial Radio Access (E-UTRA); Physical Channels and Modulation (Release 9) [EB/OL]. [2013-10-10]. http://www.3gpp.org.

[13] TSAI Y H, SANG T H. A new timing synchronization and cell search procedure resistant to carrier frequency offsets for 3GPP-LTE downlink [C]// Proceedings of the 1st IEEE International Conference on Communications in China. Piscataway: IEEE, 2012: 334 - 338.