使用DNS作为REST Web服务的发现机制

REST架构的应用在公开Web服务与企业集成两个方面取得较稳定的发展势头。然而，在面向服务架构的某一方面没有取得足够的重视：服务发现机制。

在这篇文章中，我将描述现存的Web技术是如何促使在RESTful Web服务实现服务发现机制。

寻找服务发现的解决方案

一个想要与某一服务交互的客户端需要一个初始的URI用来进入所提供的应用。那如何来获取这个用来进入应用的URI呢？有以下三种不用的方式：

1. 在客户端对这个URI配置进行硬编码；
2. 创建一个专用的基于HTTP的解决方案（一个服务注册应用）；
3. 利用现有的一些服务发现机制。

如今，硬编码或者将初始的URI配置写入客户端应该是最常见的手段。然而，坏处是期间所引入的耦合将使得服务的提供者难以重新部署服务，或者为服务配置某种形式的负载均衡。

通过创建专用的Web应用来提供基于HTTP的服务注册机制、同时关联标准媒体类型是一种切实可行的选择，但是这只是在没有现存可用技术时的最后选择。

使用现存以部署的技术显然是最好的解决方案。我们无需到处寻找这种大规模普适的可靠解决方案，域名系统（DNS）就是我们所需的。DNS主要是一个用来查找对于主机的IP以及提供例如查找域名对应邮件服务器等其他服务的分布式数据库。

这篇文章将描述如何利用标准的DNS功能来实现服务发现机制。

服务发现机制的需求

服务发现机制的解决方案应当具有怎样的能力？这是一个很常见的场景：一个客户端从实现一个特点功能的服务列表中查询并挑选一个进行交互。通常这个客户端在某个场景下限制了这个查询。例如，一个客户端可能不需要所有的购物服务，它只需要www.examplebooks.com网站下的购物服务。

我们可以把提炼出以下的需求：一个服务发现机制应该是客户端有能力

1. 获取特定种类下所有服务的列表；
2. 获取一个特定服务示例的入口URI；
3. 获取关于这个服务的元数据；
4. 限定查找的边界（如，查找domain example.org下所有的搜索引擎）

我们会马上发现，现存的DNS支持上述所有的需求。接下来我们先环顾并了解一下RESTfull系统中服务类型的概念。

RESTful系统的服务类型

服务发现机制的需求之一就是使得它能够获得期望的容量。通常，一个客户端从指定种类的服务列表中基于附带的元数据或者随机地选取一个可用的服务。

但不幸的是，服务类型这个概念还没有出现在REST Web服务领域中。为了让基于类型的服务发现机制成为可能，我们必须首先创造一个能详细说明服务类型的方法。

哪里适合定义服务类型及其名称呢？考虑到RESTful Web系统唯一能使用的说明文档是媒体类型及其类型关系的规范，那将其定义在媒体类型规范中就很自然了。

那它是如何运作的呢？让我们来看一下博客服务与搜索服务这两个例子。

定义REST Web服务类型的例子

Atom出版协议

Atom出版协议规范定义了一系列的可用于实现出版接口的超文本类型以及链接关系。它最开始的用途是用来出版发布网络日志，但它同样可以用来为任何一个管理条目、组织条目的系统提供服务接口。

为Atom出版协议定义一个服务类型简单到只需在其规范中为这个服务类型指定一个名字。然后客户端，用这个名字通过服务查询机制，查询这个实现了Atom的服务。为了方便，我们之后都将这个Atom服务名字取为*_atom_http*。（不用担心这个名字中的下划线或是名字的格式，这只是一个DNS转换。我会在后面解释这个。）

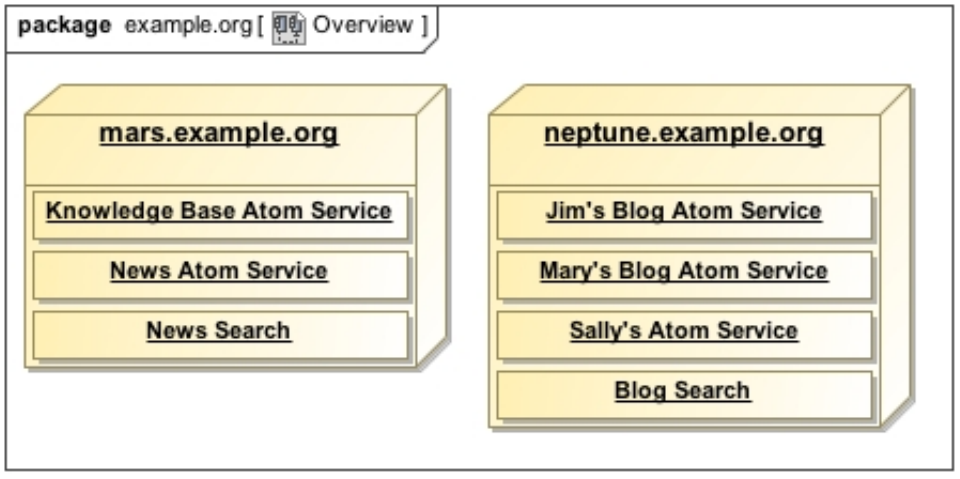
OpenSearch

OpenSearch规范规定了“OpenSearch描述文档用来描述一个能够被搜索客户都所使用的搜索引擎。”我们不必深究此处的服务类型：搜索引擎。我们不如把它转变成一种名称为*_search_http*的Atom服务。

通过在OpenSearch规范中加入这个服务名称，我们能够用它来做基于DNS的服务发现。

域名example.org

在这篇文章中，我将用*_atom_http*以及*_search_http*这两个实际的服务类型作为例子来说明我们将要讨论的内容。下面是一张关于example.org域名的图样。其中的两个主机*mars.example.org*与*neptune.example.org*都运行了Atom协议以及搜素类型的服务。



*mars.example.org*主机上提供了知识库以及新闻订阅两个Atom服务。同时还有一个用于查找新闻归档的搜索服务。*neptune.example.org*上运行了一些博客以及与之关联的搜索服务用于

查找其中的条目。

根据我们之前列举的需求，一个服务发现机制需要解决如下几个问题：

1. 其中哪些服务是支持Atom协议的？
2. 其中哪些服务是支持OpenSearch规范的？
3. 如何定位到指定的服务？
4. 指定的服务中又有哪些可用的元数据？

另外，它还需要限制example.org域名返回结果的数量。

DNS（十分）笼统的介绍

在互联网早期，每个主机上都保存了记录主机名与对应的ip地址的一个静态文件（hosts.txt）。随着互联网上主机数目的增长，由于规模问题更新这个文件变成了一个难题。DNS通过非中心化地管理这个数据库，解决了这个规模性的问题。去中心化的管理通过委托的方式来实现，这就意味着一个特定索引路径块（一个域名）的拥有者可以将这个域名的部分所有权委托给其他组织来代替管理。

DNS是一个路径索引的分布式数据库，它具有以下的能力：

1. 分布：一个DNS数据库可以分布在互联网上很多台名称主机上。
2. 委托：一个特定索引路径块的拥有者可以将部分域名所有权委托给其他人。
3. 类型资源记录：DNS可以关联域名的若干种数据类型记录。在域名查询时可以指定所需的类型。
4. 缓存：考虑到性能，DNS自带缓存。
5. 容错性：DNS可以复制若干个，使得当单个名称服务器不运行时仍继续提供服务。

域名的管理者通过修改名称服务器来控制DNS查询所返回的结果。通常，名称服务器拥有这个域名一些部分（区段）的完整信息。

当名称服务器启动时，它会从一个文件中或通过其他名称服务器加载一个或多个区段，然后才能够响应对于这些区段的查询。

当一个DNS客户端查询一个指定域名的信息时，它与它配置所指向的名称服务器建立联系，依次查询其他名称服务器直到请求的资源记录被定位，然后返回给客户端。

使用DNS实现服务发现

在DNS中，一个域名能够关联若干种资源记录。除了我们比较熟悉的A记录（用于地址查询）或者是MX记录（用于邮件服务查询），DNS还定义了其他一些资源记录类型如SRV（用于服务定位）、TXT（用于任意文字数据）以及PTR（用于表达其他域名的引用）。我们可以通过同时使用后面三种来用DNS提供服务发现能力。

SRV资源记录

SRV资源记录用来描述域名区段中可用服务的主机与端口。以下是一条SRV配置的示例：

```
_ldap._tcp.example.org.  IN   SRV  0 0 389 venus.example.org.
```

上面的这条配置声明了一个在 venus.example.org 主机的389端口上 可用的 _ldap 服务 。

备注：其中用0填充哪些字段用作负载分布；_tcp和_udp是SRV资源记录的规范，用于标识所用的协议，加在服务类型与域名之间。

通过是用域名解析软件如nslookup，我们可以向DNS请求获得在example.org上的可用ldap服务。

```
$ nslookup -q=srv _ldap._tcp.example.org.
Server:         xxxx
Address: ip

_ldap._tcp.example.org      service = 0 0 389 venus.example.org.
```

上面的这个调用是用来查询_ldap._tcp.example.org上所有可用的SRV记录。（-q=srv这个命令行选项告诉域名解析器查询SRV记录 ）

如果还有更多可用的LDAP服务，这个区段的配置可能如下：

```
_ldap._tcp.example.org.   IN   SRV  0 0 389 venus.example.org.
_ldap._tcp.example.org.   IN   SRV  0 0 389 mercury.example.org.
_ldap._tcp.example.org.   IN   SRV  0 0 389 earth.example.org.
```

如果按照上面的配置，那么查询的结果将会变成

```
$ nslookup -q=srv _ldap._tcp.example.org.
Server:         xxxx
Address: ip

_ldap._tcp.example.org      service = 0 0 389 venus.example.org.
_ldap._tcp.example.org      service = 0 0 389 mercury.example.org.
_ldap._tcp.example.org      service = 0 0 389 earth.example.org.
```

这个返回结果的 意思是example.com域名有三个LDAP服务，并且分别陈列了它们的位置（主机与端口）。

DNS服务发现（DNS-SD）

由于SRV记录不能被用来配置一个服务类型对应的命名示例，并且对于任何指定的主机及端口，只配置一个服务，SRV记录在所需的服务发现需求上受到了极大的限制。举个例子来说，我们无法让SRV记录指向运行在同一web应用的两个搜索服务（他们共享同一个主机与端口）。另外SRV记录不支持为一个特定的服务示例配置元数据。

DNS服务发现规范已解决了这个限制。DNS-SD通过同时使用SRV、PTR和TXT资源记录来迎合所有的服务发现需求。用法如下，

- PTR被用来将服务类型映射到被命名的服务实例上
- SRV被用来提供服务示例的位置以及端口号
- TXT用来提供服务实例中附加的元数据

使用PTR记录配置服务实例列表

DNS-SD吧服务类型域名与PTR记录结合起来用来把服务类型名映射为服务实例名。这样就可以通过指定的服务类型检索到所有实例的列表。

下面的dns的区域配置行说明了这个理念：

```
# Note that all names are relative to example.org, for example,
# _atom_http._tcp is really _atom_http._tcp.example.org.
_atom_http._tcp          PTR KnowBase._atom_http
_atom_http._tcp          PTR News._atom_http._tcp
_atom_http._tcp          PTR JimBlog._atom_http._tcp
_atom_http._tcp          PTR MaryBlog._atom_http._tcp
_atom_http._tcp          PTR SallyBlog._atom_http._tcp
_search_http._tcp        PTR NewsSearch._search_http._tcp
_search_http._tcp        PTR BlogSearch._search._tcp
```

每个PTR的左边是服务类型域名，右边是这个类型对应的实例。

DNS服务发现指定使用下面服务实例命名约定：

<Instance>.<ServiceType>.<Protocol>.<Domain>

例如，Jim的博客日志服务的完整实例名是：

JimBlog._atom_http._tcp.example.org

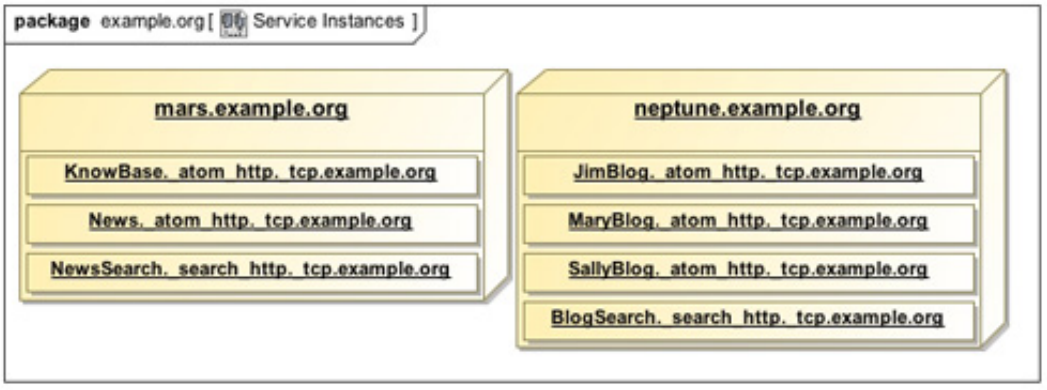
现在，对example.org上的所有类型为_atom_http._tcp的服务的查询将被表示为（注意-q选项的srv将取代为ptr）：

```
$ nslookup -q=ptr _atom_http._tcp.example.org
Server:         xxxx
Address: ip

_atom_http._tcp.example.org    name = MaryBlog._atom_http._tcp.example.org.
_atom_http._tcp.example.org    name = SallyBlog._atom_http._tcp.example.org.
_atom_http._tcp.example.org    name = KnowBase._atom_http.example.org.
_atom_http._tcp.example.org    name = News._atom_http._tcp.example.org.
_atom_http._tcp.example.org    name = JimBlog._atom_http._tcp.example.org.
```

DNS-SD将把这个PTR查询解释为对查询域所指定的服务类型（这儿是_atom_http._tcp.example.org)的实例列表的查询。因此得到的结果的右边是检索到的服务实例名。

系统使用服务实例名对example.org域所构建的设置图现在看起来如下：



配置服务实例化数据-PTR-和TXT记录

DNS-SD明确使用PTR记录，并使能够让DNS客户端获得一个服务的列表。服务实例反过来被

描述为使用SRV和TXT记录。SRV记录提供了关于主机和端口的服务实例信息。TXT记录提供了额外的元数据，用于修饰或说明该实例。

DNS-SD为TXT记录指定了一个通用键值格式（key1=val1,key2=val2,...）,并把一个给定的服务类型作为类型的规范。

对于基于http的服务，例如*_atom_http* 和 *_search_http*在构建入口URI服务实例的时候,至少有一个路径键是必要的。下面是Jim博客中的一个服务例子：

```
JimBlog._atom_http._tcp      SRV 0 0 80
mars.example.org.
                             TXT path=/blogs/jim
```

这个配置表示服务实例可以访问JimBlog._atom_http._tcp.example.org mars.example.org上的80端口。TXT记录指定了一个路径参数，客户端必须使用 这个参数构建服务器的入口http uri。这个路径参数事实上必须是满足服务器类型规范的一个http URI。因此原子服务使用https将需要一个不同的服务类型,例如原子https。

关于Jim的博客实例信息检索服务可以像这样:

```
$ nslookup -q=any JimBlog._atom_http._tcp.example.org.
Server: xxxx
Address: ip

JimBlog._atom_http._tcp.example.org service = 0 0 80 neptune.example.org.
JimBlog._atom_http._tcp.example.org text = "path=/blogs/jim"
```

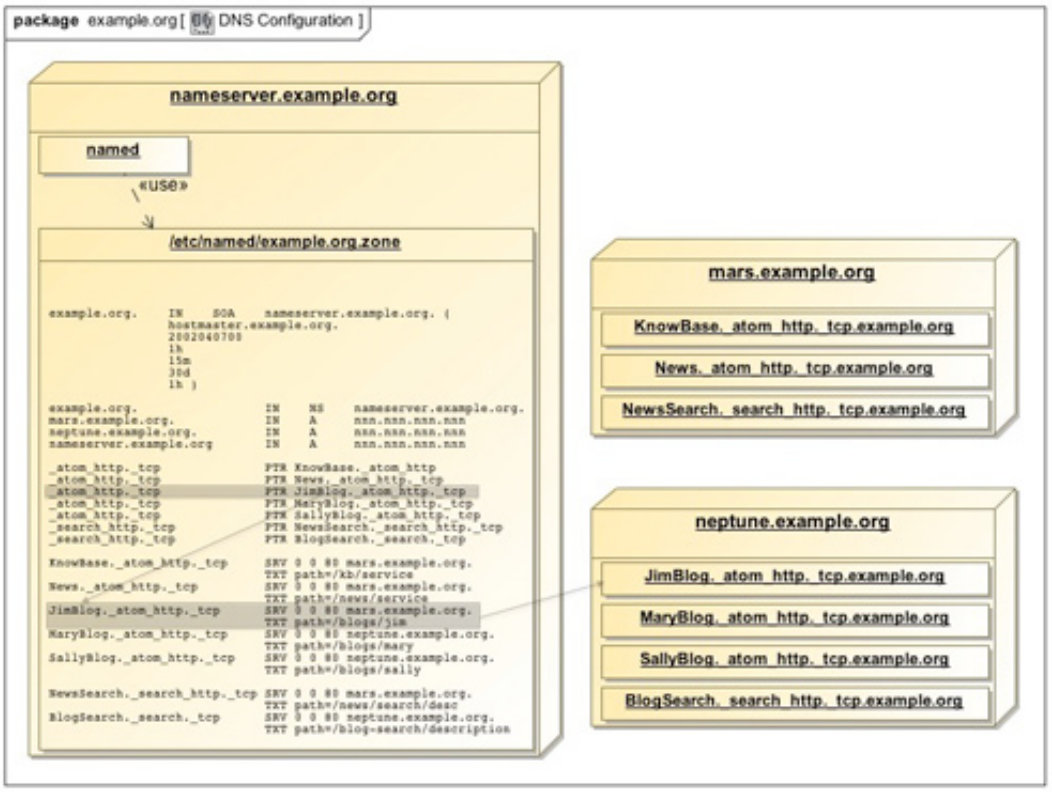
从这里，根据服务类型规范规则，客户端可以构造服务的输入URL。给定的例子生成的服务URI将是下面的样子：

```
http://neptune.example.org:80/blogs/jim
```

下面的图展示了*mars.example.org* 和*neptune.example.org*服务主机以及名称服务器主机*nameserver.example.org*。大量文本工件显示了完整的区域配置文件为*example.org*。

在最顶端是SOA，表明在给定区域（SOA="启动授权"）服务是经过授权的，下面的这个实例中，我们可以看到几行定义了区域中的命名服务器（NS记录）和主机的ip地址（A-record）。

其余的文件包含了讨论Jim博客服务中完整的配置细节。Jim博客服务的配置项展示了他们是如何彼此引用并最终在neptune.example.org进行服务实例化的。



完成交互实例

我已经在讨论过了,DNS-SD是怎么使用PTR-, SRV-, 和TXT 资源的记录来启用DNS基服务发现的. 简而言之, 下面显示的是一个完整查的找过程例子.

1. 从example.org获取Atom出版协定列表.

```
$ nslookup -q=ptr _atom_http._tcp.example.org
Server:         xxxx
Address: ip

_atom_http._tcp.example.org    name = MaryBlog._atom_http._tcp.example.org.
_atom_http._tcp.example.org    name = SallyBlog._atom_http._tcp.example.org.
_atom_http._tcp.example.org    name = KnowBase._atom_http.example.org.
_atom_http._tcp.example.org    name = News._atom_http._tcp.example.org.
_atom_http._tcp.example.org    name = JimBlog._atom_http._tcp.example.org.
```

2. 获取配置好例子的位置和meta数据.

```
$ nslookup -q=any MaryBlog._atom_http._tcp.example.org.
Server:         xxxx
Address: ip

MaryBlog._atom_http._tcp.example.org   service = 0 0 80 neptune.example.org.
MaryBlog._atom_http._tcp.example.org   text = "path=/blogs/mary"
```

3. 基于type_*atom_http*服务的定义,构造服务等级URL.

```
http://neptune.example.org:80/blogs/mary
```

4. 以HTTP方式进入服务并获取初始化应用状态.

```
$ curl http://neptune.example.org:80/blogs/mary
200 Ok
Content-Type: application/atomsvc+xml

<service>
...
</service>
```

在上面的例子中, 服务端可以获取首次请求中的任何服务实例事件. 只有服务端不去区分这些事件的时候(比如服务端发送一个检查或者是交互的功能)才是最合适的. 然而,多数情况下,只有在基于事件所处理的数据的情况下,这些事件才会有重大作用.在这种情况下,我很期待服务事件配置能显示出配置好的服务事件名,这样就能让客户端稍后按照事件名将其绑到服务中去 .

使用dns的整体优势

应用已有的技术，尤其它已被广泛应用十多年，有众多的优势，诸如：

- 大范围的良性测试的实现是可用的，许多已被开源啦。
- 知识渊博的开发人员与管理员被广泛的应用。
- 现有的广泛使用可以保护技术投资。
- 至少dns本身有下面的特别优势：
- 通过复制的可靠性
- 内置缓存
- 行政责任与负担的委托
- 容易配置
- 良好的支持(DNS-SD 是以苹果Bonjour协议为基础的)

总结

服务发现是面向服务架构的一个重要方面，因为它避免了在早期将客户端绑定到特定服务实例。消除这个耦合为整个系统的重新配置带来了更高的灵活度。

通过利用DNS-SD所指定的标准DNS机制，服务发现可以很容易的被引入RESTful的Web服务系统。考虑到无处不在的DNS名称服务和解析器实现的可用性，基于DNS的服务发现任何人都可以获得，无论他处于什么样的系统环境。

互联网技术已经在全球范围成功部署了超过十年，通过给DNS启用服务发现，你可以给互联网技术增加性能及可靠性。

本文地址：http://www.oschina.net/translate/rest-discovery-dns
原文地址：http://www.infoq.com/articles/rest-discovery-dns
