基于 REST 架构的 Web2.0 的研究

李亚强
北京邮电大学计算机科学与技术学院， 北京（100876）
E-mail: liyaqiang82@gmail.com

摘 要：Web2.0 是近几年发展起来的替代 Web1.0 的一种 Web 应用模式。REST 架构的出现，满足了 Web2.0 发展的需求，同时 Web2.0 反过来促进了 REST 在 Web 中的应用。本文分析了 Web2.0 应用模型和 REST 架构的特点，并对开发基于 REST 架构的 Web2.0 应用进行了研究，最后提出了该应用中使用的主要技术模型。

关键词：REST，Web2.0，Ajax，SOAP
中图分类号：TP393

1. 引言

1989 年，英国科学家 Tim Berners-Lee 和比利时人 Robert Cailliau 在欧洲粒子物理研究所共同提议和构造了在因特网上使用超文本的方式来发布、共享和管理信息，此时宣告 Web 的诞生。Web 是 World Wide Web，即互联网的简称，是一个相互链接在一起、通过 Web 浏览器来访问的超文本文档系统。浏览器里看到的网页，可能包含文本、图像以及其它的多媒体，通过文档之间的超链接，可以从一个网页浏览到其它网页。随着 Web 的不断发展，目前它拥有最大量的用户，变成了最大的分布式计算系统，基于最广为接受的开放标准，不断推陈出新，成为一个动态生长着的开放系统。因此，可以说 Web 是一个正在生长着的、开放的、动态的分布式系统。

起初的 Web2.0 大都是基于 RPC 模型的 Web 服务的方式。在一个相对封闭的环境中，RPC 模型取得了较大成功。因为在封闭环境中，所有用户都是已知的，可以共享一个数据模型。用户直接调用发布接口的 API 来获得服务，服务器端的开发者也通过接口获得了很大的灵活性。但是随着 Web 规模的不断扩大，RPC 的方式严重的偏离了 Web 发展的初衷。因为使用 RPC 的方法会导致调用者和服务之间的紧密耦合，同时 RPC 系统在 Web 规模下很容易产生接口复杂性。

Web2.0 急需要满足它发展的 Web 架构。REST 的出现从很大程度上解决了这个问题。REST 定义了应该如何正确地使用 Web 标准，例如 HTTP 和 URI。如果你在设计应用程序时能坚持 REST 原则，那就预示着你将会得到一个使用了优质 Web 架构的系统。REST 在本质上是一个可以被许多不同技术实现的高层次的风格，而且可以被实例化。通过统一的 URI 来标识 Web 中的所有资源，并且使用 HTTP 通用的操作接口来操作这些资源，即将 HTTP 协议作为了信息交互的主要契约，真正发挥 HTTP 协议的各种功能。因此，REST 将繁琐的 Web 应用从 RPC 复杂的接口中解放出来，通过简洁和人性化地方式开发应用，方便共享 Web 的各种资源，使 Web 应用更加贴近用户，更加简约。

2. Web2.0

Web 2.0，是O’Reilly公司在 2003 年首先使用的一个词，之后 2004 年又召开了Web 2.0 大会，这个词迅速流行起来。它主要指基于Web 的下一代社区和托管服务，以博客、RSS、网摘、P2P、即时信息、社会网络、维基百科、大众分类等社会软件的应用为核心，依据六

度分隔、XML、Ajax等新理论和技术实现的互联网新一代模式，帮助Web用户协作和分享，以Google、MySpace、Facebook等网站为代表[1]。Web2.0 并不只是Web的升级换代，实际上是指软件开发人员和最终用户使用互联网平台的方式的变化，同时，在商业上，一直在探索如何才能在Web这个平台上获得成功。

2.1 Web2.0 实质

Web2.0 实质上是社会、商业、技术在 Web 的平台上的重新融合和进一步发展。

经过多年网络宽带的迅速发展，Web 用户迅速增加，大量普通用户需要需要更好用、更个性化、更加以他们为中心、更多样的内容、服务和应用，这成为了 Web1.0 向 2.0 迈进的最大动力。同时，长尾理论在 Web 上的成功商业应用大大的促进了 Web 的革新，这种长尾经济主要就是以如何让相互协作和分享的用户口碑相传来病毒式地营销，如何利用这种用户社区的传播和推荐能力来开拓大规模的个性化服务为特点，从大规模单一市场转向选择性极大并丰富的多样化长尾市场。技术上，开发者希望更加快捷和方便的开发满足用户的 Web 应用，因此能够获得更好的用户体验和更加轻量级的编程模式赢得了大量开发者的拥护。

社会、商业和技术三种因素融合式地共同发展，相互推动，经过几年全球范围的尝试和实践，新的生活方式、商业模式和技术范型不断涌现，Web 变得更加开放、动态，社区也更加去中心化，人们却又能够更加紧密地协作和分享，内容、服务和应用更加的多样化和个性化，商业模式也逐渐走向如何满足大规模的小众需求。

2.2 编程准则

首先，Web 2.0 倡导的是简单性，即软件容易使用、易于组合、易于扩展。Web 2.0 应用的最初吸引之处就是它的简单，避免并隐藏了那些多余的复杂性。从编程者的角度，Web2.0 的应用一定要简约而不简单，能够清晰构思你的目标。而从用户的角度，Web2.0 应用能够使用户在一个时候只关心一个一个特性，完成一个目标。这对传统软件，是一次颠覆性的革新，把传统软件从开发者花大量人力物力无开发和维护，并且要花大量的时间去训练用户使用软件转变到开发者的简单实现和用户的简单操作。

其次，Web 2.0 中链接是最基础的思想。这也是我们称之为 Web 的一个理由。链接是把 Web 中各种实体连接起来的最基本的元素。Web2.0 中的链接需要遵循一些规则：

1. Web 上的任何东西都是可以被 URI 或者是 URL 所连接的。
2. 把所有的链接都保存为他的原始出处，扩大共享的范围。
3. 链接必须是持久的，它不会在没有任何缘由的情况下被改变或者是消失。
4. 链接应该是可读的、稳定的、并且能够自我诠释的。

然后，Web2.0 奉行软件即服务的观念。只要用户付费即用，无需担心开发、安装、部署和运营维护，开发过程也极大程度地由用户驱动，用户需求及时反馈。还要符合长尾理论，通过口碑和网络效应来实现 Web 应用的推广。

最后 Web2.0 的应用应该满足以下几个架构模型特征：

1) 使用 REST 来表示和访问服务，每个网络资源可以用一个 URI 来唯一地表示和确定，并且使用 HTTP 的基本操作，包括 GET、POST、PUT、DELETE 等。

2) 数据被编码成 XML 文档或者 JSON 的方式来进行交换数据。

3) 基于 Ajax 的丰富用户体验。

3. REST 架构

REST 是英文 Representational State Transfer 的缩写，中文翻译为“表述性状态转移”，他是由 Roy Thomas Fielding 博士在他的论文《基于网络软件体系结构的构架风格与设计》中提出的一个术语。REST 本身只是为分布式超媒体系统设计的一种架构风格，而不是标准。它是 Fielding 博士对当前 Web 体系结构潜在设计原则的一种描述，也是对 Web 最成功要素的总结。REST 架构风格是全新的针对 Web 应用的开发风格，是当今世界最成功的互联网超媒体分布式系统架构之一，它使得人们真正理解了 HTTP 协议本来面貌。

3.1 REST 风格准则

基于 Web 的架构，实际上就是各种规范的集合，REST 也不例外。REST 正是在原来的各种 Web 规范基础上增加新的规范而形成的一种 Web 架构形式，并不是什么新的标准。REST 架构风格具有三个主要准则：

1) 网络上的所有事物都被抽象为资源，并且每个资源对应一个惟一的资源标识符 URI（Unified Resource Identifier）。一个 Web 应用总是使用固定的 URI 向外部世界呈现。REST 中的资源所指的不仅仅是数据，而是数据和表现形式的组合。比如一个电子商务公司可以用“HTTP://www.dianzishangwu.com/book/123_html”来表示一本书《童话》的介绍，它的形式是 HTML 文本形式，而“HTTP://www.dianzishangwu.com/book/123_text”也表示这本书的介绍，但是其形式是 text 的文本形式，所以被归为不同的资源，这也是 REST 为什么要这样命名的原因。在 REST 系统中，Web 上每一个资源都对应唯一的一个 URI。不管资源是图片，文字还是视频文件，甚至只是一种虚拟的 Web 服务，也不管你是 xml 格式、txt 文件格式还是其它文件格式，全部通过 URI 对资源进行惟一的标识。

2) 通过通用的连接器接口对资源进行操作。REST 是基于 HTTP 协议的，任何对资源的操作行为都是通过 HTTP 协议来实现。HTTP 不仅仅是一个简单的运载数据的协议，而且还是一个具有丰富内涵的网络软件协议。他不仅仅能对互联网资源进行惟一定位，而且还能告诉我们如何对该资源进行操作。HTTP 把对一个资源的操作主要是 GET、POST、PUT 和 DELETE 这四个方法，简称为 CRUD 操作。由于资源和 URI 是一一对应的，执行这些操作的时候 URI 是没有变化的，因此可以通过同一个 URL 地址对某个网络资源进行以上这些操作。

3)真正意义的无状态客户-服务器模式。从客户到服务器的每个请求都必须包含理解该请求所必须的所有信息。无状态性使得客户端和服务器端不必保存对方的详细信息，服务器只需要处理当前请求，而不必了解所有的请求历史。同时，无状态减少了服务器从局部错误中恢复的任务量，可以很容易的释放资源，因为服务器端不必在多个请求中保存状态。原来的静态网络应用时，Web 架构都是无状态的，但是随着融入式 Web 应用的兴起，Web 应用程序架构逐渐开始背离 REST 这个准则。主要是需要为个人用户提供更多个性化的内容和富应用程序模型。

REST 的目的是决定如何使一个良好定义的 Web 程序向前推进：一个程序可以通过选择一个带有链接的 Web 页面上的超链接，即状态变换，使得另一个 Web 页面(代表程序的下一个状态)返回到用户，使程序进一步运行。

3.2 REST 与 SOAP

Web 服务是一个崭新的分布式计算模型。Web 服务是一系列标准的综合，这些标准包括 XML, SOAP, UDDI, WSDL 和 WSFL 等。Web 服务利用这些标准提供了一个松散耦合的分布式计算环境。在 Web 服务的模型中，厂商将其服务分成一个个相对独立的 Web 服务，每个服务提供某类功能，客户可以通过绑定在 HTTP 之上的 SOAP 协议来访问这些服务。但不是说构建一个 Web 服务就必须使用 SOAP 和 WSDL。

SOAP(Simple Object Access Protocol，简单对象访问协议)技术有助于实现大量异构程序和平台之间的互操作性，从而使存在的应用能够被广泛的用户所访问。SOAP是Web服务中的基础协议，它采用的交互模型是基于RPC机制的，编码模式是XML[2]。目前，大多数的Web服务都是采用SOAP协议作为客户端和服务器通信的主要协议。以前，在一个相对封闭的环境中，RPC模型取得了较大成功。因为在封闭环境中，所有用户都是已知的，可以共享一个数据模型，用户直接调用发布接口的API来获得服务，服务器端的开发者也通过接口获得了很大的灵活性。但是，随着环境扩大到整个因特网后，用户如果还是通过API来获得服务，那样接口数量指数级地增长必然带来服务接口的繁琐性，大大阻碍了Web服务的提供。

REST是一种Web架构形式，而SOAP是基于HTTP的具体协议，两者并不是完全对立的。说两者存在差异性，主要因为目前SOAP是RPC模式的， SOAP和REST形式在下面几个方面有明显的区别[3]：

表 1：SOAP 和 REST 的比较

	SOAP	REST
地址模型	URI 对 SOAP 来说没有实际的意义，只是作为 Web 服务进入点的一个标识，即 SOAP 消息处理器的标识。资源和 URI 不是一一对应的。一个 URI 可能对应多个资源。如图()	URI 和资源一一对应，通过同一个 URI 可以对资源进行 CRUD 的操作。
接口方式	SOAP 不提供通用操作，每一个服务可以定义它自己的操作集，即定义自己的方法调用 API。	REST 强调使用标准通用的操作，也就是 HTTP 提供的 GET、PUT、POST 和 DELETE，通过这 4 个接口对基于 URI 标识的资源作统一的处理。
中间媒介	不能应用一些传统的Web中间媒介，因为其操作完全封装在 SOAP 消息体中，无法通过 URI 来操作网络资源。	能够充分利用 Web 中间媒介，比如缓存服务器，代理服务器等，主要是因为 REST 都是通过 HTTP 协议的标准操作，只需要中间媒介实现 HTTP 协议。
安全性	要访问的对象名称藏在方法的参数中，因此需要创建新的安全模型。	比较简单，只需匿藏 URI 就可以隐藏某个资源，还有以资源为中心的 Web 服务是防火墙友好的，可以充分利用防火墙。

4 基于 REST 的 Web2.0 应用

4.1 Ajax 客户端

Ajax是Web2.0 客户端的核心实现技术，因此基于REST的Web2.0 应用的客户端也需要采用Ajax。Ajax表示Asynchronous JavaScript加XML。它有时也称为XML的HTTP技术。在Ajax中，核心的技术主要是围绕实现与服务器的异步通信，而无需页面刷新。XHR(XML HTTP Request)对象支持对服务器进行异步HTTP操作，包括GET、POST、PUT和DELETE。客户端无需等待服务器的响应就可以进行下面的操作。Ajax应用是具有以下几个特征的Web应用[4]：

1) 使用级联样式表和 HTML 结合进行呈现。
2) 使用 XML 进行数据交互操作。
3) 使用 XHR 实现异步数据获取。

可以说，Ajax 是一种 Web 开发风格。Web 浏览器仅在必要时才通过编程语言获取数据，并动态修改用户界面，可见“动态”和“仅在必要时”是 Ajax 的本质所在。Ajax 的这些特征使它成为富客户端的主要动力，使无状态服务器和有状态客户端成为可能。无状态的服务器正好迎合了 REST 的要求，同时提供有状态的客户端也使应用程序资源和数据资源的绑定转换到了客户端，可以实现目前大多数 Web 的应用。可以说在 Ajax 没有出现之前，REST 的很多准则都无法实现。正是 Ajax 促使 REST 流行起来，逐渐使 Web 应用更加贴近其本来的面目。

浏览器可以缓存 Ajax 的控制代码，即引擎，还可以缓存 Ajax 的数据。比如 Dojo Toolkit 就是一个很好的例子。Dojo 提供了构建时的工具来创建一个包含所有应用程序逻辑、表示和风格的压缩 JavaScript 文件 dojo.js。由于它终究只是一个文件，因此 Web 浏览器可以对其进行缓存，这意味着我们第二次访问启用 Dojo 的 Web 应用程序时，很可能就会从浏览器缓存中加载 Ajax，而不是从服务器上加载它。由于 Ajax 引擎只是一个文件，因此它也是可以使用代理缓存的。在大型的企业内部网中，只要有一名员工曾经下载过某个特定版本的应用程序的 Ajax 引擎，其他任何人都可以从内部网网关上上获取一个缓存过的拷贝。这样 Ajax 引擎可以控制客户端的状态，是客户端成为有状态的客户端。同时，由于 Ajax 请求时，只是请求需要的业务数据，而原来的绝大部分没有变化的数据不需要重新下载，因此可以缓存大部分数据，而只改变或获得很少一部分关键数据即可。

另一方面，Ajax 还可以很轻松的处理服务器故障，服务器崩溃或者重新启动对于用户来说都是完全透明的，因为服务器崩溃不会影响会话状态，这些都保存在用户的浏览器中。可以说 Ajax 的出现，使 REST 的实现成为真正的可能，解决了 REST 客户端很多的效率问题，但是带来了一些安全方面的问题。

4.2 交互方式

REST 下的 Web2.0 应用客户端和服务器进行交互时，大部分数据采用 JSON 编码的格式。尽管 Ajax 这个缩写词暗示以 XML 作为数据表示格式，但是 XML 并不是 Web 应用程序可用的惟一数据交换机制。JSON 是 JavaScript Object Name 与 XML 类似，JSON 也是基于文本的，是人机可读的。从机器可读性的角度来看，JSON 似乎更容易解析和使用，特别是在浏览器应用程序中。因为它采用 JavaScript 对象和数组数据结构的语法。在浏览器应用

程序中使用 JSON，就可以将整个数据结构作为第一类 JavaScript 对象对待。这样，对于 Web 开发者来说，完全避免了额外的解析工作。JSON 格式还可以在其他流行的编程语言中使用，比如 Java 语言、PHP、Ruby 和 C++等等。

JSON由两种结构组成：名称/值对的集合和有序的值列表[5]。名称/值对中名称是字符串格式的，而值主要是指对象、记录、结构、散列表、关联数组等大多数语言支持的数据格式。值序列在大多数语言中，表示为数组。这些都是通用的数据结构，几乎所有现代编程语言都支持它们。因此由这些结构组成的数据格式可以在不同的编程语言之间方便地交换。另一个方面，JSON省去了XML中繁琐的重复标记的数据形式，大大缩小数据的大小，提高了客户端和服务器的通信效率。比如一本书的表述分别如下：

JSON 形式：{“book”:{“number”: 1, “content”: “this is a book”}}

XML 形式：

```
<book>
    <number>1</number>
    <content>this is a book</content>
</book>
```

明显 JSON 形式的数据更加简洁。

4.3 REST 服务

基于 REST 的 Web2.0 在服务器端需要提供基于 REST 的 Web 服务(REST 服务)，即使用 REST 体系结构风格创建的 Web 服务。要创建基于 REST 的 Web 服务，首先需要确定希望作为 Web 服务进行公开的所有资源。这些资源，需要使用唯一的 URI 来标识它们。例如，为了便于图书资源易于检索和操作，图书的可以使用如下 URI 来标识：/book/{bookNumber}，那么/book 这个 URI 就表示所有图书的集合，即图书列表。

然后需要确定操作这些资源的 API，由于采用 HTTP 标准的操作方法，不需要另外定义 API，只需要在 HTTP 包的头部标明 HTTP 的具体操作，数据格式以及操作的 URI 就行。由上表可知，通过向服务器发送不同的 URI 和 HTTP 包对服务器公开的资源进行响应的 CRUD 操作。如下表标明的 REST API：

表 2：图书资源的 REST API

方法	URI	数据格式	目的
GET	/book	test/json	取得所有图书的列表
GET	/book/1	test/json	取得图书 1 的详细信息
POST	/book/2	test/json	将图书 2 的信息加入到数据库中
PUT	/book/2	test/json	将数据库中图书 2 的信息更新到最新信息
DELETE	/book/2	test/json	从数据库中删除图书 2 的所有信息及图书 2 条目

最后就是在服务器端实现这些具体的 Web 服务。一般采用 Web 服务的标准模型，下图所示为 Web 服务模型中涉及的重要角色和交互：

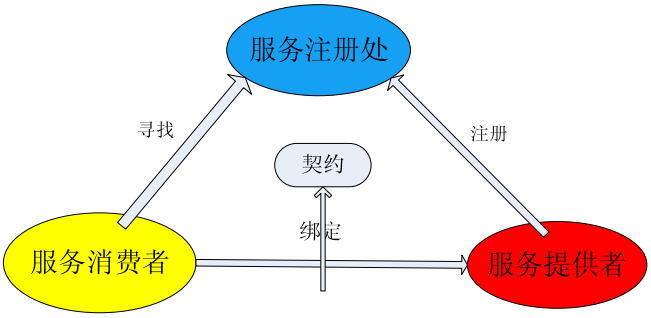

图 1: Web 服务模型

服务提供者是服务的所有者或者平台，它提供了服务的运行平台和具体的服务实现。服务消费者是寻找并调用服务或者发起与服务的交互的应用程序。服务消费者可以是由人控制的浏览器应用程序，也可以是没有用户界面的程序。服务注册处是一个可搜索的服务描述目录，服务提供者将他们的服务描述注册到这里。服务消费者通过注册处寻找服务并获得服务的相关信息。注册让服务提供者能够注册服务描述，从而让服务消费者可以找到服务。发现可以让服务消费者能够直接获得服务描述。绑定让服务消费者能够使用服务描述中指定的绑定细节，在运行时调用服务的实现或者发起与服务的交互。因此，通过这个 Web 服务模型可以根据 REST 的 API 提供不同的满足用户需求的服务。

5 结论

本文主要分析了 Web2.0 的模型特点和 REST 架构的主要特色，并进一步论述了在 REST 架构下，开发 Web2.0 应用主要的技术模型。在 Web2.0 的需要下，REST 的及时出现，为 Web2.0 的发展奠定了架构基础，Web2.0 同时促进了 REST 架构的改进，两者互相影响，互相依赖，必将指引着 Web 应用向着更加快捷简单的方向发展，必将为人们提供更加贴近大众，更能满足用户需求的 Web 服务。

参考文献

[1] Tim O'Reilly. What Is Web 2.0.http://www.oreillynet.com/pub/a/oreilly/tim/news/2005/09/30/what-is-web-20.html?page=1, 2005.

[2] 英格兰德 著，黄刚 译. Java 与 SOAP. 中国电力出版社, 2002 年 12 月.

[3] Paul Prescod. Roots of the REST/SOAP Debate. http://www.prescod.net/rest/rest_vs_soap_overview/, 2003.

[4] Christian Gross. Ajax and REST Recipes. Apress, 2006.

[5] JSON.org. 介绍 JSON. http://www.json.org/json-zh.html, 2005.

The Research on Web2.0 based on REST Architecture

Abstract

Web2.0 has been a kind of web application model instead of Web1.0 in recent years. The appearance of REST architecture is in need of the development of Web2.0. At the same time, Web2.0 promotes REST to use more in Web application contrarily. This paper analyzed Web2.0 application model and REST architecture traits respectively, and researched into how to develop the Web2.0 application based on REST architecture. In the end, the main technique model used in this application is drawn.

Keywords: REST，Web2.0，Ajax，SOAP

作者介绍：李亚强，北京邮电大学计算机科学与技术学院硕士研究生，研究方向：嵌入式处理和宽带通信。