AJAX 框架在 web 服务中的应用

Application of web services with AJAX Framework

骆建佳　尧飘海　朱伟　（浙江理工大学 杭州 310018）

摘要：AJAX 模型在 Web2.0 中的角色越来越重要，通过和传统 web 开发技术的对比，结合 AJAX 在网页开发中优势，研究了采用 AJAX 测试或调用 web 服务，并给出详细设计和实现；最后总结此方法与传统的 web 服务测试或调用的优势和弱势。

关键词：AJAX web 服务 JAVASCRIPT 简单对象访问协议

1 引言

Web2.0 技术正在如火如荼的进行，传统的 web 开发体系已不再能满足用户对交互性和体验性的需求。随着 AJAX 框架模型的提出和应用，逐渐实现了用户对 web 系统的交互的需求，它已成为世界所有用户关注的重点，也是人们谈论的中心，如何优秀而无缝的模拟 B/S 成为 C/S 技术已成为 web 开发人员的追求。

SOA（Service – Oriented Architectures）面向服务的架构技术在系统集成中具有良好的松散耦合性而得了各大厂商的技术支持和运用，如 oracle，sun，bea 等已经将 SOA 技术摆放在最重要的位置，成为人们普通关注的对象。特别是用户对各分散系统的集成和交互性的要求越来越高时，它的地位就不言而喻了。SOA 的关键技术主要集中在借助 web 服务的调用来完成，可以说如果没有 web 服务，它将不再存在。因此对 web 服务的测试和调用是也是和 web 开发人员息息相关的事情，但是目前的大部分用户对 web 服务的调用和测试都是借助于专门的工具来实现，如：SOAP Toolkit，XMLSPY 等。由于这些工具在测试应用时，常常因为不熟悉、烦琐或版权问题而得不到高效和广泛的使用；有时对开发者也不是一个很好的选择，如在某系统版本升级及对 XML 数据解包操作方面，性能存在明显的下降，使得用户无法忍受；对一些简单请求，却需要对数据的重复打包和解包的过程，用户有时根本无法忍受等待系统的响应时间而放弃使用了。于是，我们对此采用基于 AJAX 的框架模型来实现用户 web 服务的测试和调用，速度性能方面得到很大的提高，同时也给用户良好交互性和体验性。

2 AJAX 模型和 Web 服务

AJAX（Asynchronous JavaScript and XML）中的各种技术随着 web 系统的形成就已存在，它是各种技术的综合运用。目前由 Jesse James Garrett 综合并提出来，总得来说，它包含以下几个部分：：它使用 XHTML 和 CSS 标准化呈现；使用 DOM 实现动态显示和交互；使用 XML 和 XSTL 进行数据交换与处理；使用 XMLHttpRequest 对象进行异步数据操作；使用 Javascript 绑定和处理所有数据。它打破了使用页面重载的惯例技术组合，随着 AJAX 在用户体验上的表现，各大公司如：Microsoft，GOOGLE，BEA，SUN，IBM 也都纷纷加入和提供了对此技术的支持和开发，可以说 AJAX 已成为 Web 开发的重要武器。

传统 WEB 应用模型工作方式：用户通过接口向服务端发送一个 HTTP 服务请求，然后服务端接收和处理数据，并和各种传统系统交互，最后返回 HTML 页面到客户端。它通常是以超文本的形式返回给用户，从用户体验性上来看，采用超文本进行传输，对软件应用来说并不是必须的。请看图 1。

从图中我们通过与传统 WEB 模型对比，可以看出 AJAX 模型采用 JavaScript 和 AJAX 引擎层的方式来实现通过 XMLHTTPREQUEST 向服务器端发送异步请求或 XML 数据。与传统的 B/S 交互模式相比较，使用 AJAX 技术的不同点在于如图 2。

AJAX 应用程序通过在客户端和服务端之间引入

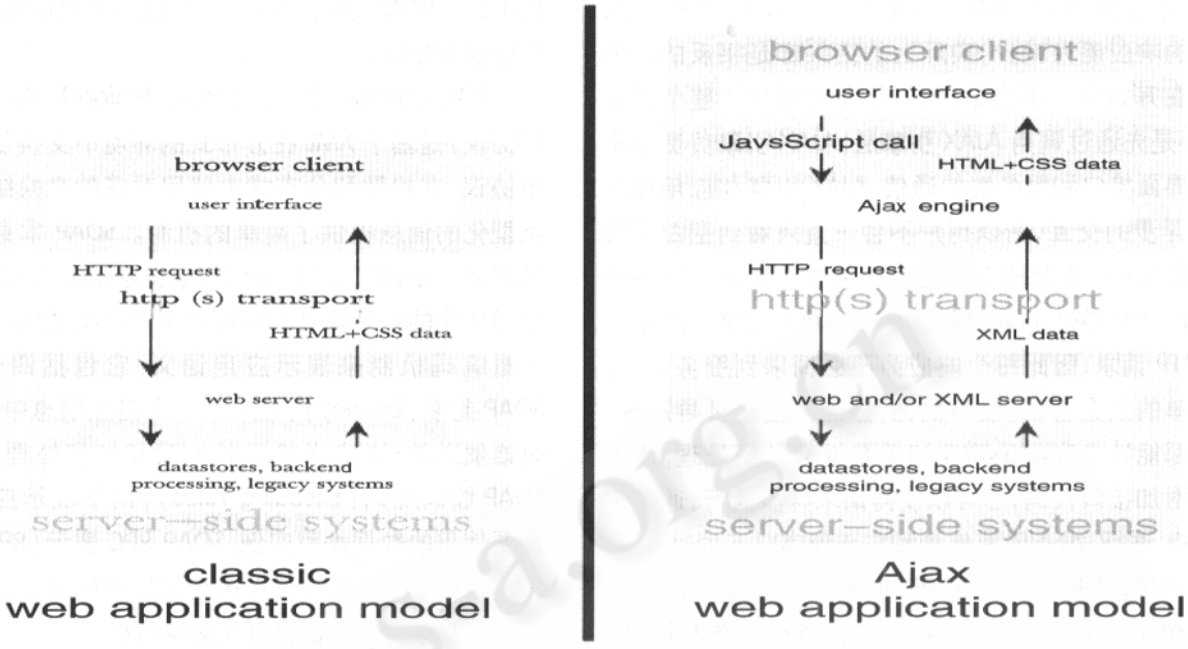

图 1 传统 WEB 模型和 AJAX 模型比较

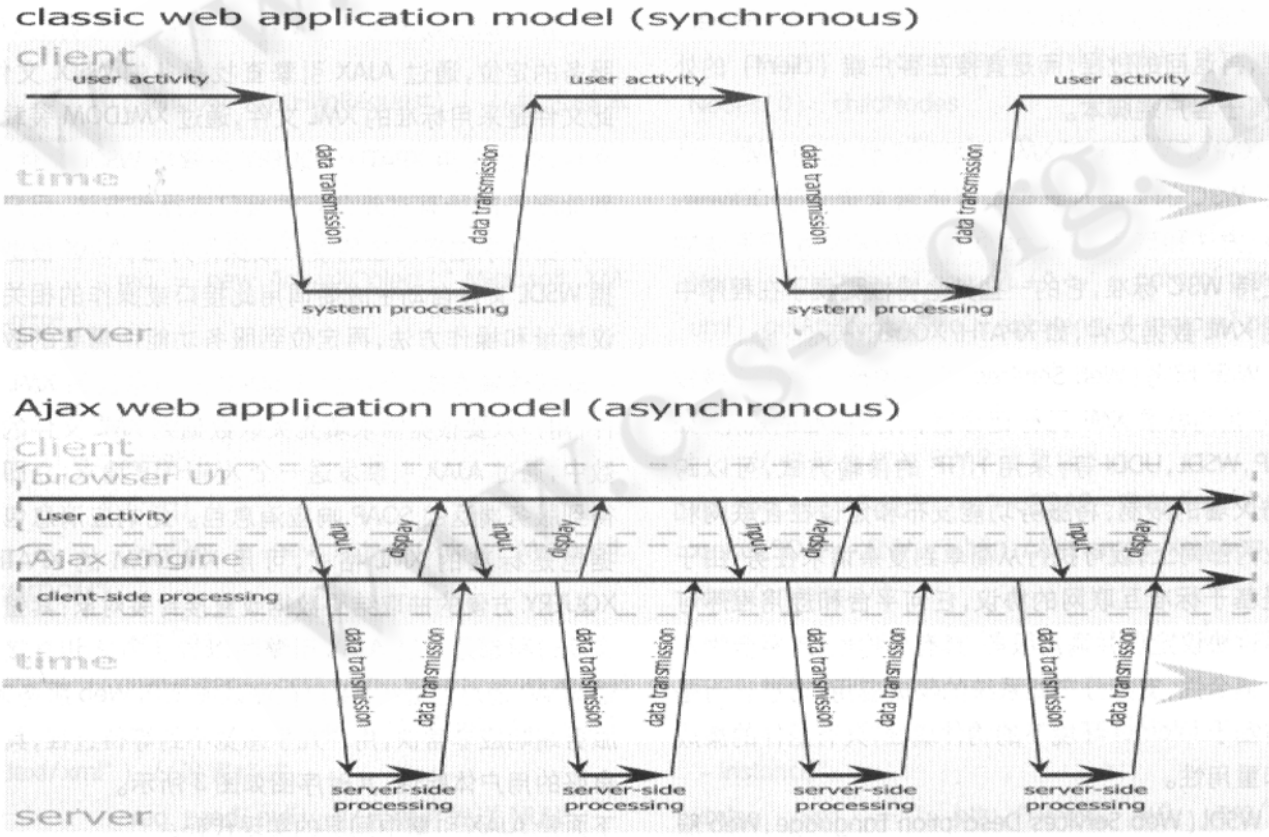

图 2 传统 WEB 模型同步工作方式和 AJAX 模型异步工作方式比较

一个 AJAX 引擎中间层，消除了 WEB 之间不断重复的 开始、停止交互动作，因此用户的整体性体验是连续

的，而不是割离的。AJAX 引擎表面上看来，使得应用程序的响应能力减少，实际上的结果却是相反的。浏览器在开始一个新会话时，并不是全部导入整个页面层，而是先通过调用 AJAX 引擎层；AJAX 引擎负责渲染用户界面接口和服务端的通信；AJAX 引擎和应用程序进行异步的交互，所以用户不会一直只看到空白页面或等待图标，直到服务器处理完成整个事务。用户的每个动作都通过 JAVASCRIPT 调用 AJAX 引擎而产生一个 HTTP 请求，因此每个响应为不必请求到服务端，比如简单的数据验证，缓冲数据，甚至一些网页导航等，引擎都能独立完成处理。如果是提交数据处理，通过装载附加接口代码，它可以使得请求异步执行，通常采用 XML 进行，因此没有停止和应用程序交互的过程。

JAVASCRIPT 是一种新的描述语言，最初由 Netscape 开发，后来得到各大公司支持；它嵌入到 HTML 的页面中，在用户浏览器上运行，通过 JavaScript 做到响应使用者的需求事件（如：表单的输入）而不用任何的网路来回传输，它不用先传给服务端（Server）处理，再返回的过程，而是直接在客户端（client）的处理，属于客户端脚本。

XMLDOM 是基于 XML 的对象模型，呈现 XML 文档内容；W3C 的文档对象模型规格通常定义了 DOM 的属性、方法和事件。Microsoft 的 XMLDOM 执行方法完全支持 W3C 标准，它的一些其他特性更便于在程序中使用 XML 数据文件，如 XPATH/XQUERY 等。

Web 服务（Web Services）是一种热门的企业级技术，它借用标准 XML 元标记语言，采用标准的规范，如 SOAP，WSDL，UDDI 等；采用 HTTP 的传输方式，可以跨越防火墙的限制；将服务功能发布和定位在互联网和企业内部网上，就可执行从简单到复杂请求任务；由于它是基于标准互联网的协议，任何平台和应用程序可以通过协议定位并调用服务，具有异构性、跨平台性及松散耦合性，支持分布式系统的集成应用；它被认为是当前基于 Internet 环境下的构件编程，具有组件的集成性和重用性。

WSDL（Web Services Description Language，Web 服务器描述语言）是用 XML 来描述 Web 服务接口定义语言标准，它描述了 Web 服务的三个基本属性：（1）服务功能——所提供的操作（方法）；（2）服务标准——交互的数据格式以及协议；（3）服务定位——协议相关的地址。WSDL 包含对操作方法和消息的抽象定义、具体协议和规范。

SOAP（Simple Object Access Protocol，简单对象访问协议）是基于 XML 在分布式的环境中交换信息的简单协议，在松散和分布式中使用对等的交换结构化和类型化的信息提供了简单的机制。SOAP 本身并不定义任何应用语义，如编程模型或特定语义实现，通过一个模块化的包装模型和对模块中特定格式编码的数据的重编码机制来表示应用语义；它包括四个部分：SOAP 封装（Envelop），定义了一个描述消息中的内容、是谁发送的、谁应当接受并处理和如何处理的框架；SOAP 编码规则（Encoding rules），用于表示应用程序需要使用的数据类型实例；SOAP RPC 表示（RPC representation），表示远程过程调用和应答的协定；SOAP 绑定（Binding），使用底层协议交换信息。

3　实现原理和设计

用户测试或调用一个 Web 服务时，首先确定 Web 服务的定位，通过 AJAX 引擎查找服务的 WSDL 文件，此文件是采用标准的 XML 文件，通过 XMLDOM 装载此文件，然后采用 XPATH/XQUERY 来解析 WSDL 文件，得到此服务的所有对外公布的接口或操作方法；用户可以根据需求选择对应的接口或操作方法，AJAX 模型根据 WSDL 文件自动生成要调用此接口或操作的相关协议地址和操作方法，再定位到服务功能所需要的数据格式和传输方式，自动生成 SOAP 的消息包的 XML 文件；用户只要根据需求填充关联数据到 XML 文件的参数中，通过 AJAX 引擎发送一个 XMLHTTP 请求，立即可得到服务端返回 SOAP 响应消息包。此响应消息包数据也是标准的 XML 格式，可用 XMLDOM 的 XPATH/XQUERY 方便的抽取结果数据或直接封装对象；其整个实现过程都是通过 AJAX 引擎的以异步方式和直接发送 SOAP 消息包进行，用户在测试或调用 Web 服务时，服务端响应非常快，用户几乎感觉不到等待过程，具有良好的用户体验性，其时序图如图 3 所示。

下面是 AJAX 引擎中间层的实现代码：

```
//定义 SOAP 消息包对象，用来保存服务 WSDL 文件
    var soapXmlClient;
    function soapAll(soapXml){
      this.soapXml = soapXml
```

```
}
//创建 XMLHttpRequest 对象
var req = null;
```

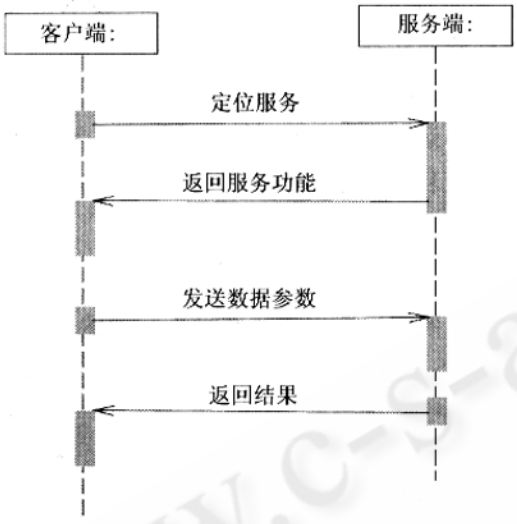

图 3　实现过程时序图

```
if (window.XMLHttpRequest) { //能否从本地创建
    req = new XMLHttpRequest();
} else if (window.ActiveXObject) {
    req = new ActiveXObject("Microsoft.XMLHTTP");
}
//创建 XMLDOM 对象
var tempStr
if (req) {
    xmlDoc = new ActiveXObject('Microsoft.XMLDOM');
    xmlDoc.async = true; //是否异步
    req.open("get", url, false); //传输方式
    req.setRequestHeader("Content-Type", "text/xml");//设置格式
    req.send(null); //传输并发送 NULL
    if (req.status==200){ // 返回正确结果标志,请参考[2]
        xmlDoc.load(req.responseXML); //载入返回结果 XML
    }
    soapXmlClient = new soapAll(xmlDoc);
```

下面是获得 Web 服务接口或操作的实现代码:

```
//通过 XPATH/XQUERY 得到操作的名称,此步骤可首先测试中使用;
operationList = xmlDoc.documentElement.selectNodes("wsdl:binding/wsdl:operation")
for(var i=0; i<operationList.length; i++){
    var tempOperation = operationList[i].getAttribute('name');
    var tempOption = new Option();
    tempOption.text = tempOperation;
    tempOption.value = tempOperation;
    document.all.selOper.options.add(tempOption)}
```

下面是生成消息包数据格式的代码:

```
// parameterStr 是采用类似获得 Web 服务接口方式可得到相应数据格式集的节点
param = parameterStr.childNodes(0).childNodes(0).childNodes
for(var paramNuml = 0; paramNuml < param.length; paramNuml++){
paramList = paramList + "<m:" + param[paramNuml].getAttribute('name') + ">" + param[paramNuml].getAttribute('type').substring(4, param[paramNuml].getAttribute('type').length) + "</m:" + param[paramNuml].getAttribute('name') + ">\n" }
// 自动生成的 SOAP 消息包
var soapBeginBodyStr = "<SOAP-ENV:Envelope
xmlns:SOAP-ENV = "http://schemas.xmlsoap.org/soap/envelope/"
xmlns:SOAP-ENC = "http://schemas.xmlsoap.org/soap/encoding/"
xmlns:xsi = "http://www.w3.org/2001/XMLSchema-instance"
xmlns: xsd = "http://www.w3.org/2001/XMLSchema">\n<SOAP-ENV:Body>\n"
var soapEndBodyStr = "</m:" + getoperation + ">" + "\n</SOAP-ENV:Body>\n</SOAP-ENV:Envelope>"
```

```
var soapStr = soapBeginBodyStr + tempStr + " \n" +
paramList + soapEndBodyStr; //SOAP 文件
//利用上面创建的 req 对象重新发送数据,就可以得到服务端返回的标准 XML 结果
document. all. soapText. value = soapStr // 用户端填充参数
req. send( document. all. soapText. value) ;
var rtnDocXml = req. responseXML
```

4 优势和弱势

从以上设计原理和代码实现可以得出,采用基于 AJAX 框架模型的 web 服务调用有如下优势:

(1) 采用 AJAX 框架模型主要采用 JAVASCRIPT, XMLHttpRequest, XMLDOM, HTML 等传统技术来实现,而不是某项新技术来完成开发,web 开发人员不需要重新学习就可以直接运用到工程中,大大提高了开发效率和减少学习成本。

(2) 整个功能的测试或调用都是在客户端完成和处理,可以减轻服务端的负担和冗余请求,实现按需分批载入数据,同时数据是利用 XML 格式直接传输和发送的,而不用经过对象系列化和反系列化来实现,减少了数据对象解包和压包的过程,提高了服务端的响应量和吞吐量。

(3) 通过 AJAX 模型,所有的处理可以在同一个页面完成,用户无须刷新页面或跳转页面,减少了用户等待时间,具有更好的用户体验性和交互性。

(4) web 开发人员把以上代码进行组件封装,开发人员无须重复编写开发,通过对组件的对象或函数调用来完成 web 服务调用,实现了软件组件复用技术,减少软件开发量。

(5) AJAX 模型得到各大软件商的技术支持和广泛应用,已成为技术潮流无法阻挡的趋势,用户对它的依赖性也越来越强,具有极大的市场。

当然,在研究和测试中,我们也发现 AJAX 模型对 web 服务调用存在着某些不足,如:

(1) 由于 AJAX 引擎采用了传统的 JAVASCRIPT 来实现,如果客户端禁用它,那么此特性用户将无法体验,或客户端的浏览器版本对 JAVASCRIPT 支持程序的不同,也可使得其在程序上的执行效率降低,随着 AJAX 的标准实行和统一,此问题将得以解决。

(2) 如果服务端返回的数据非常重,由于 JAVASCRIPT 本身运行速度限制,这可能会使得其在批量处理中受限,且目前对字符集的支持程度也有待加强。

5 结束语

AJAX 架构是采用传统技术来实现的一种模型,由于其在 B/S 应用中的优越表现,对传统的 web 开发来说,无疑是一种强而有力的冲击,将会得到越来越多的青睐。基于 AJAX 的 web 服务的测试和调用,给用户和开发人员带来了极大的交互性和体验性,而且也减轻了服务端的负担,提高了服务端的响应量和吞吐量,但是其在大规模数据调用中还存在性能缺陷有待进一步的研究。

参考文献

1 Jesse James Garrett. A New Approach to Web Applications[EB/OL]. http://www.adaptivepath.com/. 2005.2.

2 Nicholas C. Zakas, Jeremy McPeakandJoe Fawcett. Professional Ajax[M]. Wrox Press . 2006.

3 Eric Newcomer, Greg Lomow. Understanding SOA with Web Services[M]. Addison – Wesley. 2006.6.

4 柴晓路等, Web Services 技术、架构和应用[M],电子工业出版社, 2003.1.

5 James Snell. Call SOAP Web services with Ajax[EB/OL]. http://www.ibm.com/. 2006.1.