第 36 卷 第 7 期 计 算 机 工 程 2010 年 4 月
Vol.36 No.7 Computer Engineering April 2010
·软件技术与数据库· 文章编号：1000—3428(2010)07—0052—03 文献标识码：A 中图分类号：TP311.52

Ajax 技术的数据响应优化

谭 力，杨宗源，谢瑾奎
(华东师范大学计算机科学技术系，上海 200241)

摘 要：对 Ajax 工作原理和数据传输性能进行分析，从响应数据的返回格式角度入手，通过实验，对比 2 种数据格式 XML 和 JSON 的差异与优劣，基于实际应用场景，给出权衡取舍的建议，总结出效率和安全性更高的 Ajax 模型，改善了目前 Ajax 带来的数据响应冗余的缺陷。

关键词：Ajax 技术；数据响应；XML 传输格式；JSON 传输格式

Data Response Optimization of Ajax

TAN Li, YANG Zong-yuan, XIE Jin-kui
(Department of Computer Science and Technology, East China Normal University, Shanghai 200241)

【Abstract】This paper analyzes the work principle and data transmission performance of Ajax, and discusses the difference between XML and JSON by experiments with a focus on the format of response data returned. Some suggestions to choose either based on the current application scenarios are given, an optimized model of Ajax which is more efficient and secure is summarized, which improves the shortcomings such as data response redundancy caused by Ajax nowadays.

【Key words】Ajax; data response; XML transmission format; JSON transmission format

1 概述

作为多项传统技术的优势整合，Ajax 以其各技术分支所体现出的综合优势提供着较好的用户体验，因此，获得了广阔的 Web 应用市场。Ajax 综合使用了 JavaScript, XHTML, CSS, DOM, XML , XSTL 以及 XMLHttpRequest 等技术，集上述各技术的优缺点于一身。当运用 Ajax 技术来开发一个健壮的软件系统时，需要考虑 Ajax 的可移植性、效用性和可用性等非功能性因素。而在 Ajax 与服务器异步交互的过程中如何高效地传递响应数据则是很重要的一个环节。本文从 Ajax 的数据响应角度进行分析，探讨如何趋利避害，构建一个高效的 Ajax 应用程序，提出了一个整体优化方案。

2 Ajax性能优化分析及研究进展

在 Ajax 广泛应用的今天，提高 Ajax 应用程序的执行效率已成为首要问题。要对 Ajax 的性能实施优化，首先要考虑的是 Ajax 的基本工作原理。Ajax 采用异步交互方式，在用户与服务器之间引入了一个用 JavaScript 编写的 Ajax 引擎，来代替用户与服务器进行交互。这样用户可以无需等待响应，继续其他的 Web 交互。传统的 Web 应用模式(同步交互方式)如图 1 所示。

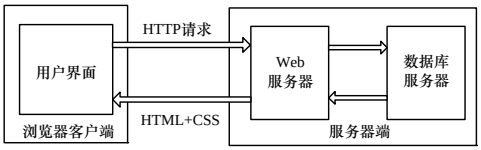

图 1 传统的 Web 应用模式(同步交互方式)

基于 Ajax 的 Web 应用模式(异步交互方式)如图 2 所示。从图 1 和图 2[1]的对比中可以看到，与传统 Web 应用相比，在基于 Ajax 的 Web 应用中，在向作为中间层的 Ajax 引擎返回响应数据并解析时，需要额外的 XML 或文本中间数据解析过程，存在着一定的效率低下问题，这会影响该 Web 应用程序的整体性能。

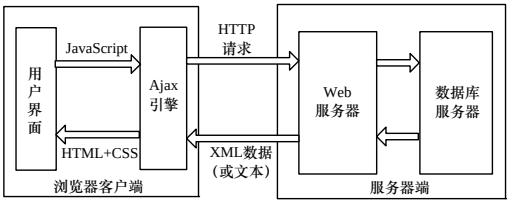

图 2 基于 Ajax 的 Web 应用模式(异步交互方式)

目前，国内关于 Ajax 性能优化已有的研究主要包括：对基于 Ajax 的 MVC 模式进行改造，借鉴 Ajax 和 DataWindow 技术创建和实现较高交互性能的 B/S 应用[2]，提出一种基于 JSON 的对象序列化算法，通过分析 JSON 文法并建立对象导航图，来解决解析 XML 所造成的缺陷，对 Ajax 的首页加载模式进行改进[3]等。但是，这些研究只是针对 XML 或 JSON 各自的缺陷进行改进，当程序员面对一个 Ajax 实际开发场景时，却仍无法作出关于选用何种响应数据载体的合理选择。

3 响应数据传输格式对比

Ajax 引擎可以用 2 种数据响应格式(纯文本和 XML 文档)来获取从服务器返回的信息。若使用纯文本格式，当前最常用的格式是 JSON。用 JSON 和 XML 来返回响应数据各有利

基金项目：高等学校博士点基金资助项目“构件系统中软件架构定义与构件实现一致性和通信方法的研究”(20060269002)
作者简介：谭 力(1986－)，男，硕士研究生，主研方向：软件工程，Web 开发技术，形式化方法；杨宗源，教授；谢瑾奎，讲师
收稿日期：2009-08-03 **E-mail：**darkwhite29@gmail.com

弊，而由于 XML 作为 Web 上数据表示事实标准的地位，在传统方式下较多采用 XML 作为数据载体。以下将从几个方面作进一步分析，通过对比两者的优劣，就可选用一个当前应用场景下最合适的响应数据载体，从而达到优化 Ajax 响应数据传输过程进而提高整个程序性能的目的。

3.1 数据量

下面通过 2 个示例：在 Google Suggest 中输入查询字符串“compiler”后返回搜索建议(示例 1)和电子邮件(示例 2)，来对 2 种数据响应格式作出对比。

示例 1 的 XML 格式代码如下：

```
<?xml version=“1.0” encoding=“utf-8”?>
<suggestions terms=“compiler”>
<suggestion term=“compiler compliance level” results=“451, 000 results” />
<suggestion term=“compiler design” results=“952, 000 results” />
<suggestion term=“compiler options” results=“253, 000 results” />
<suggestion term=“compiler optimization” results=“417, 000 results” />
<suggestion term=“compiler services” results=“51, 400 results” />
<suggestion term=“compiler course” results=“5, 400, 000 results” />
<suggestion term=“compiler conference” results=“5, 080, 000 results” />
<suggestion term=“compiler error” results=“1, 700, 000 results” />
<suggestion term=“compilers and compiler generators” results=“314, 000 results” />
</suggestions>
```

示例 1 的 JSON 格式代码如下：

```
{“suggestions”:{
“terms”:“compiler”,
“suggestion”:[{
“term”:“compiler compliance level”,“results”:“451, 000 results”}, {
“term”:“compiler design”,“results”:“952, 000 results”},{
“term”:“compiler options”,“results”:“253, 000 results”},{
“term”:“compiler optimization”,“results”:“417, 000 results”},{
“term”:“compiler services”,“results”:“51, 400 results”},{
“term”:“compiler course”,“results”:“5, 400, 000 results”},{
“term”:“compiler conference”,“results”:“5, 080, 000 results”},{
“term”:“compiler error”,“results”:“1, 700, 000 results”},{
“term”:“compilers and compiler generators”,“results”: “314, 000 results”}
] }}
```

由于篇幅关系，示例 2 的代码不具体给出。XML 和 JSON 格式响应数据量分析如图 3 所示。

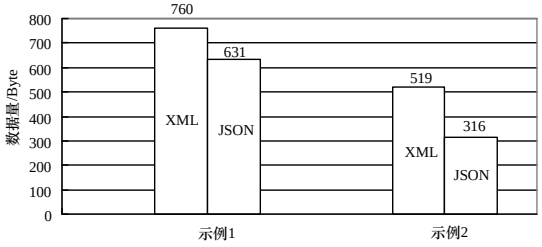

图 3 XML 和 JSON 格式响应数据量分析

可以看到，若服务器端返回的是 XML 格式的数据，除了冗余的开始结束标签之外，还必须确保该 XML 文档含有特定内容的首部信息，如 XML 版本号和编码格式；而若采用 JSON 返回响应数据，使用的结构化符号将短小精悍许多，如逗号、冒号等只占一个字符的简单符号，这将在很大程度上减少浏览器和服务器之间传输的数据量。随着应用程序中的数据交换量的增长，数据结构的复杂化，JSON 数据量小的优势将更明显。

3.2 客户端解析效率

JSON 与 XML 都是结构化的数据交换格式，两者的不同在于 XML 本身是 DOM 树结构的，需要 JavaScript 操作 DOM 元素来进行解析才能获取其中的数据。而 JSON 本身就是 JavaScript，因此，只要调用 JavaScript 的 eval()方法将 JSON 字符串序列化成为 JavaScript 对象之后，就可以直接读取其属性来获取数据。

下面通过一个简单的测试程序来计算代码的执行时间，从而比较 2 种格式读取数据时的解析效率。其中，解析 XML 中响应数据的代码如下：

```
var t1 = new Date().getTime();
for(var i=0; i<10000; i++){
   var data = request.responseXML;
   var name = data.getElementByTagName
            (“name”)[0].firstChild.nodeValue;
   var website = data.getElementsByTagName
            (“website”)[0].firstChild.nodeValue;
   var email = data.getElementsByTagName
            (“email”)[0].firstChild.nodeValue;
}
var t2 = new Date().getTime();
alert(t2-t1);
```

解析 JSON 中响应数据的代码如下：

```
var t1 = new Date().getTime();
for(var i=0; i<10000; i++){
   var data = eval(‘(’+request.responseText+‘)’);
   var name = data.person.name;
   var email = data.person.email;
   var website = data.person.website;
}
var t2 = new Date().getTime();
alert(t2-t1);
```

上面 2 段分别解析了 XML 和 JSON 中响应数据的代码中，for 循环中的代码是两者各自的解析操作，在解析操作代码段的开始和结束分别使用了 JavaScript 中的 getTime()函数来记录时间，最后计算差值，即代码的运算时间。为消除偶然误差，重复实验 10 次，数据解析效率分析如图 4 所示。

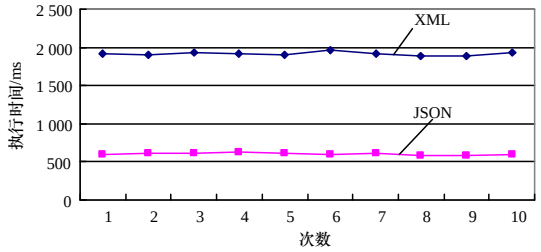

图 4 XML 和 JSON 格式的数据解析效率分析

从图 4 可以看出，JSON 的数据解析效率远比 XML 高，即使用 JSON 作为响应数据载体能会加快页面响应速度，使得运用 Ajax 而获得的即时页面更新效果更加显著，用户体验更加流畅。

3.3 服务器端开发效率

对于如何将一段数据序列化为一个 XML 文档，各种服务器端编程语言都有提供多种方式来实现，如在.NET 框架

下，C#中的 XmlSerializer 类，通过结合 TextWriter 类，它可以序列化一个对象成为 XML 格式。

而对于 JSON 的自动生成支持工具，目前来讲还比较少。因此，从服务器端开发效率上来讲，较早出现并且更加标准化、规范化的 XML 会比 JSON 有更好的表现。

3.4 安全性分析

JSON 是一种用于在 2 台机器之间的传输数据的数据交换格式。由于它承载的只是数据，不会含有赋值和调用，因此它是安全中立的。而当开发人员用 eval()函数把 JSON 数据作为 JavaScript 代码执行，从而转化为 JavaScript 对象时，攻击者可以在JSON数据中携带恶意的JavaScript代码发送给客户端，这样 eval()函数就会执行这些恶意代码，系统可能会因此而崩溃。

另外，JSON 本质上就是 JavaScript，由于 Web 应用程序的所有访问者都可以阅读到程序中的 JavaScript 源代码，因此对于一些敏感数据，除非使用了代码混淆器之类的工具来使得 JSON 文档的 URL 无法预测，使用 JSON 才是安全的。

XML 由于解析时不含有任何本地执行过程，因此相对 JSON 来讲更安全一些。

4 优化策略

通过以上分析，可以总结出 XML 与 JSON 的对比结果如表 1 所示。

表 1 XML 和 JSON 的对比结果

数据格式	数据量	解析效率	开发效率	安全性
XML	大	低	高	较好
JSON	小	高	低	差

可以看出，XML 表现较好的方面是用户可读性、服务器端处理效率和安全性，而 JSON 优于 XML 的方面是数据量和客户端解析效率。于是，针对具体的开发场景，可以给出如下响应数据载体选取方案用作基本优化策略：在开发一个基于 Ajax 技术的 Web 应用程序时，在安全性要求不高及服务器处理能力较强的场景下，选用 JSON 更好；在用户体验要求不高的安全敏感场景下，选用 XML 较合适；在用户体验和安全性要求都较高的场景下，应从大局着眼，选用较为安全的 XML 而牺牲一部分系统性能。

总之，开发人员需要认真评估在不同场景下 2 种响应数据表示方式的成本和效率，了解两者的差异后，再来根据实际需要进行合理选择，或直接采用传统的页面重载刷新的方式而不是采用 Ajax 技术。

5 结束语

本文的不足之处在于提出的部分观点尚无法给出量化的模型来衡量，有的只能通过主观经验来判断，如怎样根据安全隐患的严重程度来决定选用 XML 还是 JSON，这都是需要进一步研究的内容。

参考文献

[1] Garrett J J. Ajax: A New Approach to Web Applications[EB/OL]. (2005-02-18). http://www.adaptivepath.com/ideas/essays/archives/000385.php.

[2] 王 东, 孙 彬. 基于 Ajax 的 MVC 框架的改造分析[J]. 计算机应用, 2007, 27(S1): 293-295.

[3] 阳 锋, 徐建波. Ajax 技术的性能改进研究[J]. 计算机工程与科学, 2008, 30(6): 146-148.

编辑 顾姣健

（上接第 51 页）

(4)术语的非交性检验。给定术语集 S，如果对于任意 2 个原子术语或者术语公式 x, y 和任意一个本体商空间 $[O]_i=<[X]_i, [F]_i, [T]_i>, i\in\{0, 1,\cdots, n\}$，有$[x]\cap[y]=\varnothing$，则称 y 与 x 非交。

根据以上定义，对“本体商空间$[O]_i<[X]_i, [F]_i, [T]_i>, i\in\{0, 1,\cdots, n\}$中的术语 a 和 b 是等价的”进行验证，具体方法如下：

(1)给定本体 $O=<X, F, T>$和属性集等价关系集 R。

(2)在属性等价关系集 R 的作用下，对本体 O 进行商空间运算，形成不同层次的商空间：$[O]_0<[O]_1<\cdots<[O]_n$。

(3)如果存在$[O]_i=<[X]_i, [F]_i, [T]_i>, i\in\{0,1,\cdots, n\}$，使得$[a], [b]\in[X]_i$ 成立，且$[a]\subseteq[b]$和$[b]\subseteq[a]$成立，则根据商空间的保真原理[4]，一定有 $a\equiv b$ 成立。

其他方面的检验方法类似，在此不再重复。

3.2 实例检验

定义 9 实例声明的一致性：给定本体 $O=<X, F, T>$，若存在本体属性集等价关系集 R 是实例声明 β 的一个模型，则称 β 是一致的，否则称 β 是不一致的。若属性集等价关系集 R 既是 β 的一个模型，又是$[O]_n$的一个模型，则称 β 关于$[O]_n$是一致的。若属性集等价关系集 R 是$[O]_n$的一个模型，则称$[O]_n$是一致的。

下面对本体实例的一致性进行验证：

(1)给定本体 $O=<X, F, T>$和属性等价关系集 $R=\{R_0, R_1, \cdots, R_n\}$。

(2)在等价关系 $R_n\in R$ 的作用下，对本体 O 进行商空间运算，确定实例商空间$[O]_n=<[X]_n, [F]_n, [T]_n>$。

(3)如果存在个体 a，使得类的实例声明$[C]_i([a]_n), i\in 1, 2,\cdots, n$ 成立，则称实例声明 β 关于$[O]_n$是一致的。

(4)如果存在个体 a, b，使得属性的实例声明$[P]_i([a]_n, [b]_n), i\in 1, 2,\cdots, n$ 成立，则实例声明 β 关于$[O]_n$是一致的。

4 结束语

本文介绍了属性粒度商空间下本体的形式化和检验问题，实现了在不同层次的属性商空间之间进行自由跳转。下一步的工作是研究基于属性粒度商空间的本体推理，构造一个基于属性粒度商空间的完备推理体系，并且基于此进行应用开发。

参考文献

[1] 王洪伟, 吴家春, 蒋 馥. 本体的形式化模型及在语义查询中的应用[C]//全国搜索引擎和网上信息挖掘学术讨论会论文集. 北京: 高等教育出版社, 2003: 205-213.

[2] 邓 凯, 吴家春, 王洪伟. 本体论在知识图书馆的应用研究[J]. 情报科学, 2003, 21(1): 106-108.

[3] 李彦敏, 王晓东. 基于角色概念的合作学习 Ontology 的构建[J]. 河南师范大学学报: 自然科学版, 2007, 35(2): 47-49.

[4] 张 钹, 张 铃. 问题求解理论及应用[M]. 2 版. 北京: 清华大学出版社, 2007.

[5] Zhang Ling, Zhang Bo. The Quotient Space Theory of Problem Solving[J]. Fundamental Informatics, 2004, 59(2/3): 287-298.

[6] 邓志鸿, 唐世渭, 张 铭, 等. Ontology 研究综述[J]. 北京大学学报: 自然科学版, 2002, 38(5): 730-738.

编辑 张 帆