会话跟踪技术及其应用

朱泽民

(黄冈师范学院 计算机系,湖北 黄冈 438000)

摘　要:会话跟踪是 WEB 应用系统中一个很重要的技术,用于向用户提供个性化服务等用途。本文在分析几种传统会话跟踪技术的基础上,介绍了主流开发平台 JAVA 和 NET 中引入的新的实现技术。

关键字:会话跟踪;cookie;URL 重写;HttpSessionState;HttpSession

中图分类号:TP393　　文献标识码:B　　文章编码:1672-6251(2006)07-0056-04

Session tracking technology and its application

ZHU Ze-ming

(Computer Department of Huanggang Normal University, Huanggang 438000, China)

Abstract: Session tracking is a very important technology in the WEB application system, for example it can provide personalization service to the user. This article analyzes several traditional session tracking technology, and introduces the new realization technology on the mainstream developing platform as JAVA and .NET.

Key words: Session tracking; Cookie; URL rewriting; HttpSessionState; HttpSession

1　引言

在建立基于 WEB 的电子商务等系统中,常常需要通过提供个性化服务与客户更好的沟通,并使客户更容易找到令自己满意的产品和服务,如在 MSN.com 等网站,一些高级技术允许访问者定制主页来满足个人需要和偏好。会话跟踪技术就是其中的主流技术。

从技术上来说,Internet 通信协议可以分为两大类:有连接协议和无连接协议,两者的最大差别在于客户端和服务器端之间维持联机上的不同。用于 WEB 通信的 HTTP 是一种无连接的协议,对于简单的 WEB 来说,是一个很不错的协议。如果一个客户端只是单纯地请求一个文件(HTML 或 GIF),服务器端可以响应给客户端,并不需要知道一连串的请求是来自于相同的客户端或是不同的客户端,而且服务器端也不需要担心客户端现在是处在连接状态还是已经断线了。但是这样的通信协议使得服务器端难以判断所连接的客户端是否是同一个人。

然而,在写 WEB 应用时,有时必须考虑一些问题:如何把相关的请求联系起来,即怎样维护用户在服务器中的状态?这种维护对 Web 应用所提请求之间的状态的概念就称为"会话跟踪"。会话跟踪(session tracking)是指一类用来在客户端与服务器之间保持状态的解决方案,简单地说,当一个客户在多个页面间切换时,服务器会保存该用户的信息。

实现跟踪主要有两种方法:一是让客户在每次发起请求时标识自己,然后从服务器上存储并获取与本客户相关的数据;二是向客户发送数据,并让客户在发起请求时把数据送回。在具体实践中,也有多种实现方法,传统的有使用 Cookies 技术、重写包含额外参数的 URL(URL Rewriting)、建立含有数据的隐藏表单字段(Hidden Form Field)等。随着 WEB 技术的发展与进步,各种高级语言开发工具在对传统的会话跟踪技术提供较好的支持的同时,提供了新的会话跟踪技术,其中以 SUN 公司的 JAVA 语言与 MICROSOFT 公司的 .NET 平台为代表。本文在传统会话跟踪技术的基础上,介绍了会话跟踪的新技术。

2　传统会话跟踪技术

2.1　Cookie 技术

Cookie 是由 WEB 服务器存储到个人计算机上的许多"关键字=值"对的小文本文件,它允许网站跟踪访问者的活动。首次访问网站时,用户的计算机可能接收一个 Cookie;以后每次重新访问那个网站时,都会重新激活该 Cookie。收集到的信息是一条匿名记录,其中包

收稿日期:2006-03-20

作者简介:朱泽民(1978-),男,讲师,硕士研究生,研究方向:计算机应用。

含用于个性化来访问的数据。服务器 Cookie 设置了有效期,时间一过浏览器就会停止发送 Cookie。根据保留时间的长短,Cookie 可以分为会话 Cookie 和持续性 Cookie。

例如,购物应用程序的 Cookie 可能存储用户的标识符。一旦商品添加到购物车,或者执行要 WEB 服务器发出请求的其他任务,服务器就会接收一个包含了用户标识符的 Cookie。然后,服务器利用这个标识符来定位该用户的购物车,并执行任何必要的处理。

现在主流的程序设计工具都提供了 Cookie 类。如在 JAVA 语言的 Servlet API、.NET Framework 都提供了一个 Cookie 类。通过 Cookie 类可以非常方便的创建一个新的 Cookie,设置它的名字和值以及有效期等等,然后把 Cookie 放在 HttpServletResponse 对象中并发送给浏览器;还可以从 HttpservletRequest 对象中获得 cookie,并读得它们的值,然后传递给响应对象。

使用 Cookie 的好处在于它比在 URL 或表单中存储数据更直观,我们不用欺骗浏览器将数据发回,浏览器有意识地参与了这个过程。但是使用 Cookie 仍会有问题,主要有:

(1)Cookie 可以用于在比一次短会话更长时间的内跟踪用户——实际上,它们可用来跟踪某个用户向站点发起的每一个请求。一些人担心这意味着 WEB 站点的管理员能够收集到用户所浏览网页的足够信息,因此他们关闭了浏览器的 Cookie 功能。

(2)并不是每个浏览器都支持 Cookie,有些用户为了防止泄露隐私以及从安全性上考虑,可能会禁用浏览器的 Cookie。

(3)由于功能限制或者设置有误,代理服务器不能够代理 Cookie,导致通过代理服务器上网的用户不能登录进入以 Cookie 进行会话跟踪。

为了防止 Cookie 会话攻击,用户可以采取许多措施来加强安全,如结合多种会话跟踪技术,来增加攻击者难度;跟客户端 IP 地址相结合,即把当前会话与客户端 IP 地址结合在一起来加强安全的等。

2.2　URL 重写技术

HTTP 的 GET 请求由服务器的统一资源定位符(URL)及紧随其后的包含参数/值对的查询串组成。例如,一个 HTTP GET 请求的例子可能是:

http://bjweb.163.net/cgi/ldapapp?funcid=main&sid=HAPGfUDusCLAQSIm

在这个例子中,服务器地址是 bjweb.163.net;服务器的资源是 cgi/ldapapp,查询串是 funcid=main&sid=HAPGfUDusCLAQSIm。一般情况下,这些参数都由用户在 HTML 表单中输入,但是也不总是那样。

URL 会话跟踪是把标识会话的字符串(如以例中的 funcid=main&sid=HAPGfUDusCLAQSIm)加在 URL 里面,对于客户端的每一个 HTTP 连接请求,服务端都会把 URL 里的会话标识和它所保存的会话数据关联起来,从而能够区分不同的客户端,以及进行用户会话跟踪。一般在关闭浏览器后,保存在服务器里的会话关联数据并不会立即失效,一段时间内用户请求过的 URL 仍然有效,他人只要从浏览器的历史记录里找到该 URL,就可以点击进入用户访问过的网页,并不需要任何密码验证。

使用 URL 重写方式有几个缺点:

(1)对于大量的会话数据,URL 会变得很长而失去控制,并且在某些环境下,URL 字符串的长度有一定的限制。

(2)保密性不强,自己访问的数据很容易泄露给他人。别有用心的人可以通过拦截请求或者查看浏览器的历史信息来获得查询串数据,在浏览器地址栏里输入相同的 URL 就能轻易地进入用户的网页,如 Web-Mail。

2.3　隐藏表单字段

隐藏表单字段的方法,是利用 HTML 内 Hidden 的属性,把客户端的信息,在用户不察觉的情形下,偷偷地随着请求一起传送给到服务器处理,这样一来,就可以进行会话跟踪的任务了。可以下列的方法来做隐藏表单字段的会话追踪。

<input type="Hidden" name="userID" value="15">

然后将重要的用户信息,如 ID 之类独一无二的数据,以隐藏字段的方式传送给服务器。隐藏字段的优点在于 session 数据传送到服务器端时,因为值作为隐藏表单字段存储起来,HTTP Post请求就可以避免一些保密问题和将字段追加到 HTTP GET 请求时的环境限制。不过这种做法还是有它的缺点:一旦 session 数据储存在隐藏字段中,就仍然有暴露数据的危机,因为只要用户直接观看 HTML 的源文件,session 数据将会暴露无疑。这将造成安全上的漏洞,特别当用户数据是依赖于用户 ID、密码来取得的时候,将会有被盗用的危险。

3　Java ServIet APl的会话跟踪技术

Java Servlet API 定义的 HttpSession 是在服务器端保存会话信息,在客户端的 Cookie 中保存会话 ID 或者将会话 ID 添加到 URL 后面的一种状态'记忆'技术。当用户第一次访问站点时,分配给用户一个会话对象和一个单独的会话 ID,这个 ID 是惟一的。在以后的

请求中，会话 ID 标识了这个用户，会话对象作为请求的一部分发送给 Servlet，Servlet 能从会话对象中读取信息，或者为其添加信息。

Servlet 采用“超时限制”的办法来判断用户是否还在访问：如果某个用户在一定的时间之内没有发出后继请求，则该用户的会话被作废，他的 HttpSession 对象被释放。会话的默认超时间隔由 Servlet 容器定义（默认 30 分钟）。这个值可以通过 getMaxInactiveInterval 方法获得，通过 setMaxInactiveInterval 方法修改，这些方法中的超时时间以秒计。如果会话的超时时间值设置成 -1，则会话永不超时。Servlet 可以通过 getLastAccessedTime 方法获得当前请求之前的最后一次访问时间。

典型地，使用 HTTP 会话的范型包括四个步骤[2]：

（1）创建会话

在 Servlet 的请求处理程序中，调用 HttpServletRequest 的 getSession（）方法，如果有会话则返回当前会话，没有则创建一个新会话，也可以调用 getSession（true）来执行同样的功能，或者调用 getSession（false）去访问一个现有的会话，如果没有合法的会话则返回 null。

```
HttpSession session=request.getSession(true);
```

（2）在会话中存储对象

调用 HttpSession 的 setAttribute（name，value）方法存储对象。如存储包含用户 ID 的 java.lang.String 类型的对象：

```
session.setAttribute("userID",userID);
```

（3）从会话中获取对象

调用 HttpSession 的 getAttribute（name）方法来获取被存储的对象。由于 HttpSession 不知道该对象的具体类型，所以要把值转换为所希望的类型。如获取用户 ID 的值：

```
String userID =(String) session. getAttribute ("userID");
```

（4）关闭会话

当使用完会话时，可以调用 HttpSession 的 invalidate（）方法来关闭它：

```
session. invalidate( );
```

文献［6］通过典型的购物车实例说明了如何运用 HttpSession 对象来实现会话跟踪技术。

4　利用 HttpSessionState 类实现会话跟踪

HttpSessionState 类是微软公司 .NET Framework 类库所提供的一个类，提供对会话状态值以及会话级别设置和生存期管理方法的访问。命名空间 System.Web.SessionState，程序集 System.Web（在 System.Web.dll 中），其 Visual Basic 中的原型为：

```
NotInheritable Public Class HttpSessionState Implements ICollection, IEnumerable
```

每个 WEB 窗体都包括一个 HttpSessionState 对象，可通过 Page 类的 Session 属性来访问它。网页被请求时，会创建一个 HttpSessionState 对象，并被指派给 Page 的 Session 属性。和 Cookie 相似，HttpSessionState 对象也能存储“名-值”对。这些会话项目通过调用 Add 方法来添加到 HttpSessionState 对象。如：

```
Session.Add (language,ISBN)' 添加 “名-值” 对到 Session
```

使用 HttpSessionState 对象（而不是 Cookie）的一个主要优点在于，HttpSessionState 对象可将任何类型的对象（而非仅仅是字符串）存储为属性值。

客户端首次连接 Web 服务器时，会为该客户创建一个唯一会话 ID。客户发出更多的请求时，它的会话 ID 会和存储在 Web 服务器内存中的会话 ID 进行比较。我们可能通过 SessionID 属性来访问会话的“唯一会话 ID”。如：

```
idLabel.Text = "Your unique session ID is: " & Session.SessionID
```

“将 HttpSessionState 对象 SessionID 属性值赋给标签 idLabel。

Timeout 属性指定在 HttpSessionState 对象被丢弃之前，最多能有多长的时间处于不行动状态。如：

```
timeOutLabel.Text="TimeOut: " & Session.Timeout & "minutes."
```

“将 HttpSessionState 对象 TimeOut 属性值赋给标签 timeOutLabel。

HttpSessionState 对象具有丰富的属性和方法，程序员可以很方便的对其进行操作。

5　小结

从上面的介绍来看，传统的会话追踪方式使用比较麻烦，使用 HttpSession 对象和 HttpSessionState 类可以非常方便地实现会话追踪。由于新的会话机制是基于 Cookie 或 URL 重写技术，融合了这两种技术的优点，当客户端允许使用 Cookie 时，可以使用 Cookie 进行会话追踪，如果客户端禁用 Cookie，则选择使用 URL 重写。

在实际的 WEB 编程中，使用会话跟踪技术，尤其是灵活地同时使用多种跟踪技术，会给网站带来意想不到的好处。与此同时，每种技术都有其缺限，所以有

使用时，技术员必须综合考虑利弊，采用一种合理的方案。

参考文献

[1] Danny Ayers，等著（英），曾国平，等译. Java 服务器高级编程[M]. 北京：机械工业出版社，2001.

[2] 杨浩，等. Java Servlet 会话跟踪技术的应用研究[J]. 计算机应用，2002，(9)：45~47.

[3] H M Deitel，等著（美），周靖译. Visual Basic .NET 高级程序员指南[M]. 北京：清华大学出版社，2003.

[4] Francesco Balena 著（美），李珂，等译. Visual Basic.NET 技术内幕[M]. 北京：清华大学出版社，2003.

[5] 刘淼. J2EE 平台会话管理机制分析与改进[J]. 航空计算技术，2004，(9)：116~118.

[6] 邢小永，等. Servlet API 中 HttpSession 对象的运用[J]. 天水师范学院学报，2005，(4)：65~77.

（上接第 33 页）

施管理，企业员工面临着裁员恐慌而不积极配合，管理部门不认真，大量的协调问题使进度一拖再拖。

⑹资金短缺问题

企业收款难、资金拖欠问题严重，自有资金周转较慢；企业融资困难，企业与银行之间的信息不对称造成企业融资的交易成本高；企业自身经营风险高，自有资金相对缺少。

⑺人才缺乏与人才流动大的问题

中小企业信息化常常缺乏高水平的系统管理与网络管理人才，中小企业也常常面临人才外流问题的困扰。这主要是因为：缺乏高效的人才培养与保持机制；人才激励机制扭曲；绩效评估不合理、不健全。

⑻经营管理的问题

对信息化计划实施费用与进度预算失误；不可预见的财务风险；实施中的控制偏差与技术的吸收消化能力的不足；对信息系统维护的漠视。

⑼信息交流不畅

企业从事信息化人员缺乏对技术的理解能力；很多中小企业地处中小城市或乡镇，缺乏获取相关信息的有效途径，难以有针对性地获得对完整解决方案的支持。

⑽缺乏信息化评估体系，投资不平衡，机构不合理

3　结语

中小企业的信息化问题是牵涉到多方的一个系统工程。因而中小企业信息化应从系统论的观点看待这些问题，这些问题不可能在同一个中小企业上同时出现，但就某一方面的问题一旦出现将可能导致中小企业信息化建设的失败，甚至导致中小企业的破产。因此，中小企业要慎重考虑自身信息化的问题，做好前期分析工作，找出问题并有效解决它们，把其视为一个长期发展的战略规划。

参考文献

[1] 李晓东. 方案与抉择—中外信息化发展研究[M]. 北京：中国友谊出版公司，2002.

[2] 胡启立. 中国信息化探索与实践[M]. 北京：电子工业出版社，2001.

[3] 游文丽. 对我国企业信息化管理现状的思考[J]. 商业研究，2003，(2).

[4] 周立群，谢思全. 中小企业改革与发展研究[M]. 北京：人民出版社，2002.