基于FLEX技术构建可离线Web应用程序的研究与实现

潘大四
(浙江警官职业学院信管系　杭州 310018)

摘　要　随着Web应用程序复杂性越来越高,传统的Web应用程序开始逐渐不能满足Web浏览者全方位的体验需求。RIA(Rich Internet Application,富网络应用系统)技术将客户层的设计从以页面为中心提升到以组件为中心,从而改进了客户端程序的设计。本文在研究RIA及Macromedia Flex技术基础上,提出如何构建可离线工作的Web应用程序。实际项目证明利用本地存储数据使浏览者可以在明确脱机、低带宽或连接时断时续的情况下高效地工作,同时可提高系统的响应速度,减少用户的等待时间。

关键词　Web应用程序,RIA,Flex,本地存储

Research and Implementation of Web Application Support Offline Processing Based on Flex

PAN Da-si
(Zhejiang Police Vocational Academy, Hangzhou 310018, China)

Abstract　As Web application is growing more and more complicated, traditional Web application will not satisfy Web user's all-sided experience requirement sooner or later. RIA (Rich Internet Applications) technology promotes the designing of client representation layer from page-oriented to component-oriented which enhances the designing of client programs. Based on the technology of RIA and Macromedia Flex, we propose methodologies of how to design Web application run in occasionally connected environments. Project experience justifies that using local storage allows the user to continue to work efficiently when explicitly offline, in low bandwidth, or when connectivity is intermittent, andcan improve system's response speed, reduce user's waiting time.

Keywords　Web applications, RIA, Flex, Local storage

1　引言

互联网已经日益成为应用程序开发的默认平台,提供了传统胖客户端模型(Client/Server)的替代模型,基于浏览器/服务器(Browser/Server)的瘦客户端模式应用程序开始占取主流位置,Web的广泛使用解决了C/S应用程序部署和更新的困难。基于浏览器的瘦客户端应用程序是在Web服务器上部署和更新的,因此消除了将应用程序的任何部分显式部署到客户计算机并加以管理的必要性。但瘦客户端应用程序也有一些缺点:

(1)由于采用了HTML页面形式的用户界面,客户端的数据处理能力较C/S应用程序有所回落,用户体验比较糟糕,无法像C/S那样使用丰富的效果来展示数据。

(2)浏览器必须总是具有网络连接,这意味着用户在断开连接时将无法访问应用程序,当再次连接后,必须重新输入数据,这就降低应用程序的可用性。由于应用程序的大部分逻辑和状态位于服务器上,因此瘦客户端会频繁地向服务器发回数据和处理请求。浏览器必须等待响应到达,然后用户才能继续使用该应用程序;因此,应用程序的响应速度通常要比胖客户端应用程序慢得多。这个问题在低带宽或高延迟的情况进一步恶化,并且产生的性能问题可能导致应用程序可用性和用户效率大幅度下降。

网络中断将使B/S程序无法运行。在一个网络连接依赖程度越来越高的世界里,不可避免会由于某些原因暂时失去连接,如存在延迟或带宽问题,或者可能需要拆卸网络的某些部分进行维护。即使用户的确具有良好的网络连接,应用程序也可能无法在所有时间都能访问网络资源。如所请求的服务可能繁忙、停止运行或者只是暂时不可用。

如今许多程序都移植到基于浏览器/服务器(B/S)模式的Web程序,对于服务器上某些内容,如用户每次都需要下载一些固定的容量偏大的内容到客户端本地缓冲中方可使用,这无疑在浪费有限的网络资源,同时页面等待时间会让用户难以接受。从C/S到B/S,这两者受限于技术本身分别发展成了重客户端和重服务器端的设计模式。

随着Web应用程序复杂性越来越高,传统的Web应用程序已经渐渐不能满足Web浏览者更高的、全方位的体验要求了,如何才能做到将C/S的交互用户体验与传统的Web应用程序的部属灵活结合起来的网络应用程序? RIA(Rich Internet Application)[1]技术出现使得在客户端和服务器端能够进行更好的平衡,RIA使用相对健壮的客户端描述引擎,能够提供响应速度快和图形丰富的用户界面,而快速的响应需要提升数据获得的速度作为支撑。本文在深入研究Flex[2]技术基础上,构建具有离线操作能力的"偶尔连接"客户端。通过在客户端直接存储数据,可加速Web应用程序,若服务器端数据无更新,用户可反复使用本地数据而不需要连接服务器重新获取数据,这样的方式还允许用户脱机使用数据,直至客户重新连接到服务器,极大提高客户端表现能力与性能。

2　Flex及客户端数据存储技术研究

2.1　Flex技术

Web开发人员一直想构建一种比传统HTML更丰富的客户端,这种用户接口比用HTML能实现的接口更加健壮、

潘大四　工程师,硕士研究生,主要研究方向为图形/图像处理、网络数据库。

反应更加灵敏和更具有令人感兴趣的可视化特性。RIA 技术的出现允许在因特网上以一种像使用 Web 一样简单的方式来部署富客户端程序。

RIA 是集桌面应用程序的最佳用户界面功能与 Web 应用程序的普遍采用和快速、低成本布署以及互动多媒体通信的实时快捷于一体的新一代网络应用程序。RIA 中的 Rich Client(丰富客户端)提供可承载已编译客户端应用程序(以文件形式,用 HTTP 传递)的运行环境,客户端应用程序使用异步客户/服务器架构连接现有的后端应用服务器,这是一种安全、可升级、具有良好适应性的新的面向服务模型,这种模型由采用的 Web 服务所驱动。目前几种比较成熟的 RIA 客户端开发技术[3]:包括 Macromedia Flash/Flex, Laszno, Dojo, Microsoft 的 Avalon。

Macromedia Flex 是为满足希望开发 RIA 的企业级程序员的需求而推出的表示服务器和应用程序框架,它可以运行于 J2EE 和. NET 平台。Flex 表示服务器在应用服务器内运行,提供基于标准的、声明性的编程方法和流程,并提供运行时服务、应用程序整合与管理能力。Flex 整合的能力可以轻松地通过 Web 服务、Java 对象访问或 XML 使用现有的代码及信息,用于开发和部署丰富客户端应用程序的表示层。Flex 还可以与一些现有的表示技术与框架结构如 JS PStruts 等进行集成。应用程序设计基于 Flex 开发使用扩展的 UI 组件库与基于 XML 的 MXML 来定义丰富的用户界面,利用面向对象的脚本语言(ActionScript)来处理程序逻辑,由 Flex 服务器翻译成 SWF 格式的客户端应用程序,在 Flash Player 中运行。结合了声音、视频和实时对话的综合通信技术使 RIA 具有前所未有的网上用户体验,具有高度互动性和丰富的用户体验。

2.2 客户端本地存储数据

浏览器通常将某些特殊数据,如 SessionID 或为了方便用户下次登录保存用户姓名等存储在本地 cookie 中,但 cookie 对单个域名仅仅可以存储不超过 4kB 的数据,因此 cookies 通常用来存放少量数据。近来有三种技术允许在本地存放超过 4kB 容量限制,Macromedia Flash、IE 的 User Data 和 Firefox 的 DOM Storage,如表 1 所示。Flash 从 6.0 开始就逐步具备建立窗体风格的应用程序的功能。据 Macromedia 称已经有 98% 以上的桌面系统的浏览器都安装了 Macromedia Flash Player,这使得以 Macromedia Flash Player 为客户端的 RIA 可以支持种类广泛的平台和设备。Flash Player 可以在无需用户干预下保存单个域名下 100kB 的数据,在客户进入 Flash 设置管理器中允许设置本地存储容量 100kB,1MB,10MB 至无限制。

表 1 浏览器本地存储

浏览器	存储形式	最大容量
任意	Cookies	4kB
Flash player 6	Flash Local Shared Object	100kB (可设置,最大到无限制)
IE5 及以上	UserData	1MB
Firefox2 及以上	DOM Storage	5M

数据存储在客户端可为应用程序提供内容丰富且响应迅速的用户界面,以及强大的客户端处理能力。例如,可以使用户能够执行复杂的数据操作、可视化、搜索或排序操作。通过最大限度地使用本地资源以及将本地资源集成到客户端应用程序,可以使应用程序更好、更有效地使用硬件环境。充分利用本地硬件的处理能力、内存和高级图形功能,使用客户计算机上的资源还可以减少服务器端硬件要求,最大限度地利用代码和数据部署在客户端上并且在本地执行和访问。除了存储容量大之外,Flash Object 比 HTTP cookies 另外一个好处是,SOL 文件存放在浏览器的缓存之外,当浏览器清空缓存时并不会移除文件,另外 Flash object 可以被后续的页面访问。

3 基于 Flex 技术的 PRNPAS 奖惩考核系统架构

3.1 体系结构

本文应用的实际系统是浙江警官职业学院承担的某监狱奖惩考核系统,采用 B/S 设计模式,技术上遵循 J2EE 架构,总体上分为三层结构(如图 1),分别为客户端表示层、应用服务层(业务逻辑层)、数据层。

表示层:在表示层引入 Flex 插件,客户端程序和表示层的逻辑处理程序能够同时运行在客户机由 Flash Player 提供的平台上。Flex 对现有的技术起到补充作用,并不取代现有的工具、应用服务器和数据库,为提高现有应用程序的效率增加一个强大的表示层,可充分发挥 Flex 在客户端的本地数据存储能力。Flex 数据集成提供了所有必需的工具和功能,可用于访问数据、处理数据并以多种不同的格式将数据重新发送到服务器。表示层具体采用灵活的基于组件的体系结构和对象模型,用于连接到外部数据源、管理数据和将数据绑定到用户界面组件。Flex 包括 WebService Connector 组件(连接到 Web 服务)和 XMLConnector 组件[4](通过 HTTP 返回 XML 的任何外部数据源,如 JSP、Servlet)。前端数据集成包含以下三个主要方面,如图 1 中客户端表示层部分:

- 数据连接:组件体系结构的连接器层,提供了发送和从各种数据源(如 Web 服务和 XML)接收数据的功能。数据连接提供了连接到外部数据源以检索并发送数据的功能。
- 数据管理:组件体系结构的数据储备库层,提供对数据操作(如编辑、排序、筛选、聚合和更改转换)的智能管理。
- 数据绑定:体系结构的数据管道层,提供了在组件属性之间共享数据的机制。管道集成了诸如格式器和编码器等对象,可以提供对组件间的数据传播方式的完全控制。

Flash 将数据由后台传递到 Flash,通过 Flash 界面做出修改后,传回到后台的过程分为三个方面:连接、管理和绑定,三个方面由不同组件负责。由数据连接开始,接收由服务器端编程语言从数据库中抽取数据,这一部分由各连接器(Connector 组件)负责,连接器组件用于从服务器获取数据,以及将更新数据包重新发送到服务器;之后交到数据管理,由数据集(Dataset 组件)负责,作为数据在前台的一个暂存地方,最后通过数据绑定显示到组件,数据流如图 1 所示。相反,由 Flash 将数据传回后台,所有修改通过 DataSet 组件收集,然后交给解析器(Resolver 组件),对数据的修改操作(增加、删除和修改)转为外部数据来源的格式,最后交给连接器,传回后台。

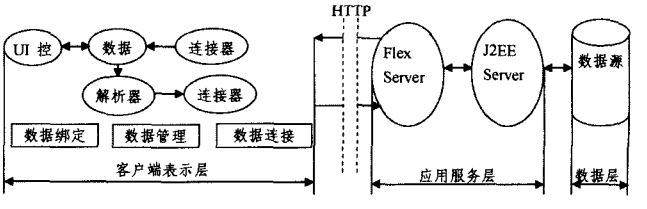

图 1 奖惩考核系统框架

应用服务层:Flex presentation server[5]在应用服务器内运行,并为 Flex 应用程序提供整合与管理能力。Flex 整合的

能力可以轻松地通过 Web 服务、Java 对象访问或 XML 传递信息。Flex 还可以与一些现有的表示技术与框架结构如 JSP Struts 等进行集成。可在服务器上建立和公开业务逻辑，因此，使用特别针对该任务设计的产品 J2EE Application Server 进行实施。

数据层：PRNPAS 奖惩考核系统采用轻量级的 O/R Mapping 框架 Hibernate 进行持久化，Hibernate[6] 是一个开源的对象关系映射工具，它减轻了对 JDBC 的直接操作，使得数据检索和更新、事务处理、数据库连接池更加简单，完全着眼于关系数据库的 OR 映射，对于最终用户几乎是透明的，使系统完全摆脱对具体数据库的依赖，提高系统的适应性与灵活性。

3.2 系统运行界面

系统运行界面如图 2 所示。

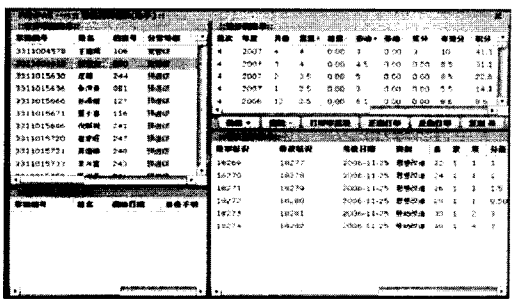

图 2 奖惩考核系统运行界面

4 Flex 技术在奖惩罚考核系统中的应用示例

下面以 PRNPAS 奖惩考核系统中频繁使用的考核依据树为例，介绍 Flex 技术在提高客户端性能方面的应用。在奖惩考核系统中，通常需要查看考核条例进行评定，考核依据是个短期内内容固定、容量较大的数据（大于 1M），为方便进行考核，系统将考核内容按照条、款、项形成一目录树，如图 3 所示，考核人员通过展开树点击相应条例进行考核。按照常规设计，直接在页面初始化时，将考核所有条、款、项内容下载到页面缓冲中构造目录树，请求页面时用户的等待时间会比较长。在整个考核系统中，这样的问题在很多模块中都会出现。如果每次都需从服务器中获得，不仅浪费带宽资源，而且页面等待时间会让用户难以接受。另外按照考核规则进行考核过程中或完成而未提交前，若遇掉线或网络故障，那么考核结果就会丢失，考核人员不得不连接服务器重新进行考核。

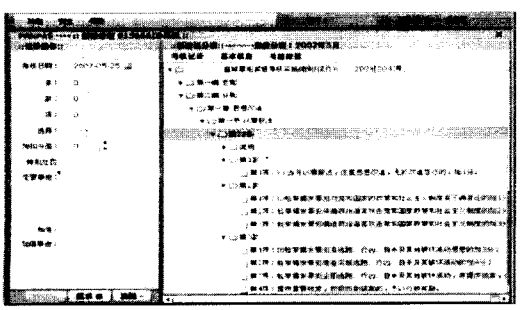

图 3 PRNPAS 奖惩考核条例目录树

根据考核管理的具体情况，为减少用户等待页面的时间，采用了基于 Flex 技术，将一些固定的容量偏大的内容或短时间内内容不会发生更改的数据存放在客户端本地，加快用户获取数据的能力，同时将考核结果在网络拥塞时暂存本地，避免数据丢失后需重新录入，重新连接后，即可更新后台数据。支持可偶尔连接操作，数据不一定要传回后台数据库存储。

DataSet 组件可以将数据存储在 LocalShared 对象里，LocalShared 对象使 Flash 在使用者的硬盘里存储数据。这样程序便可离线操作，所有修改后的数据在硬盘中会存储一份拷贝，下次在线时，可选择跟后台进行数据更新，当然可以选择不更新，离线和在线各有一份不同的数据。构建可离线操作的 UI 组件的步骤如下：

（1）初始化 UI 组件界面，相关数据未到达，此时为空；

（2）从用户本地机器上查找有无相应控件数据包，若有则加载数据到 UI 组件；

（3）若本地无数据包，则连接服务器加载数据，若本地有数据包，此时先连接服务器，判断是否有比本地新的数据，同时预先用本地数据包初始化控件，避免用户长时间等待数据失去耐心；

（4）XMLconnector 连接器侦听服务器的数据是否到达，数据到达时，若 DATASET 为空，则赋值给 DATASET。若 DATASET 非空，通过侦听时间戳，本地和服务器时间戳不同，则替换原来数据，同时按照程序设置规则，若数据容量大于一定数值，则保存在本地。

以下为设计可离线操作的目录树代码：

```
class com. epp. sheetson. PrnpasSuggestionsTree extends com. epp.
sheetson. WindowContent {
    private static var THIS:PrnpasSuggestionsTree;
    private var tree:Tree;
    var xml:XMLConnector;
    var xml4update:XMLConnector;
    var dataset:DataSet;
    private function onLoad():Void {
        //初始化树。
        iniTree();
        //如果本地有建立树的数据包，则在本地加载。
        dataset. loadFromSharedObj("$TermsTree");
        if(dataset. xmldata==undefined){
            xml. URL ="http://localhost:80/PRNPAS/TERMS_
            TREE/PRNPAS_Terms_Tree_xml. jsp";
            xml. trigger();//向服务器发送请求，这里用的是本机
            兼作服务器
        }else {
            xml4update. URL = "http://localhost:80/PRNPAS/
            TERMS_TREE/PRNPAS_Terms_Tree_Update. jsp";
            xml4update. trigger();  //发送时间戳比较请求
            tree. dataProvider = dataset. xmldata;  //为提高用
            户体验感觉先用本地数据初始化
        }
        //侦听服务器下载的数据包到达本地。
        xml. addEventListener("result",function(){
            if(THIS. dataset. isEmpty()){
                THIS. dataset. addItem();
            }else{
        //DATASET 数据集非空情况下，先移动到记录头，为保证
                新数据覆盖
                THIS. dataset. first();  }
                THIS. dataset["xmldata"] = THIS. xml. re-
                sults;//更新数据集
        //判断数据是否大于 1K，若大于 1K 则存入客户端本地
                    if(XML(THIS. dataset. xmldata). toString().
                length>1024) {
                THIS. dataset. saveToSharedObj ("$Term-
                sTree");}
        });
        //时间戳侦听，若本地时间戳与服务器不同，则重新下载数
        据
        xml4update. addEventListener("result",function(){
                var local:String = THIS. tree. dataProvider. first-
                Child. attributes. label. split(":")[1]. split(")")
                [0];
                var server:String = THIS. xml4update. results.
                firstChild. firstChild. nodeValue;
                if(server! =undefined && local! =server){
                    THIS. xml. URL =
                    "http://localhost:80/PRNPAS/ TERMS _
                    TREE/PRNPAS_Terms_Tree_xml. jsp";
                    THIS. xml. trigger();
    }
});
```

实际应用统计表明，用户首次打开包含树目录页面的速

度在3～5s之内，在数据本地存储后，再次登录服务器时，打开同样页面的速度在2～3s范围内，并且只要用户首次打开页面后，就可离线操作，可见运用本地数据存储可提高响应速度，减少用户等待时间。另外运用Flex技术在客户端表现能力方面明显得到提升。

5 并发控制

通过在客户端上存储数据，可以显著改善应用程序的性能和可用性，但必须确保适当地刷新数据并且不会使用陈旧的数据。因为许多用户可以访问和使用相同的数据，必须考虑数据并发的影响。相对于传统Web应用程序优点是：更新可能几乎立即发生，但有时可能发生在数天甚至数周以后。对于可离线操作的应用程序而言，陈旧数据的风险大于始终连接的应用程序。

PRNPAS奖惩考核系统根据实际并发访问的概率不高的情况下，采用了Hibernate的乐观锁[7]解决方案，即为数据增加一个版本标识，通过为数据库表增加一个"version"字段来实现。读取出数据时，将此版本号一同读出，之后更新时，对此版本号序列递增，将提交数据的版本数据与数据库表对应记录的当前版本信息进行比对，如果提交的数据版本号大于数据库表当前版本号，则予以更新，否则认为是过期数据。

结束语 本文在深入研究Flex技术的基础上，针对现有B/S模式应用程序响应速度慢、表现能力差的不足，结合现实用户的需求，提出构建可离线操作的Web应用程序。在此基础上，基于J2EE平台下实现了PRNPAS奖惩考核系统，在实践中得到了较好的应用，实践证明行之有效。在PRNPAS奖惩考核系统中，利用Flex技术中的客户端本地存储，好处在于：

1)可以继续使用现有的应用程序模型(包括J2EE和.NET)，构建更为直观、用户界面更加友好，易于使用、反应更迅速并且可以脱机使用的应用程序。

2)减少用户等待时间，提高系统响应速度，减少带宽成本以及增强客户关系等。

参 考 文 献

[1] Luke W. Web Application Solutions: A Designer's Guide. http://www.lukew.com/resources/WebApplicationSolutions.pdf. 2005,4

[2] Webster S, McLeod A. Developing Rich Clients with Macromedia Flex[M]. Peachpit Press, 2004

[3] Gadge V V. Rich Internet Applications 的技术选项. http://www.ibm.com/developerworks/cn/Web/. 2006,8

[4] 颜金沙，KCLY小土豆工作室. Flash MX 2004 ActionScript 2.0与RIA应用程序. 北京：电子工业出版社，2005(2)：417-425

[5] Bustelo L G, Ruano J. 使用Macromedia Flex开发Web服务客户端. http://www-128.ibm.com/developerworks/cn/webservices/ws-macroflex/. 2004,9

[6] 夏昕，曹晓刚，唐勇. 深入浅出Hibernate[M]. 北京：电子工业出版社，2005(6)：53-77

[7] 李琳骁，王海龙. POJOs IN ACTION中文版：用轻量级框架开发企业应用. 电子工业出版社，2007,4

(上接第287页)

然后，循环读管道里的数据直到无数据可读为止。

整个拦截程序编译为一个ActiveX DLL组件RSLChecker.dll。ActiveX DLL是微软提出的广泛应用于Windows系列的一种代码封装技术，提高了程序代码的可重用性，加快了程序项目的开发速度。

RSLChecker.dll在操作系统中注册后，Web服务器中的ASP程序即可调用这个组件：

```
Set objCheck = Server.CreateObject("RSLChecker.Checkobj")
```

其中，Checkobj是RSLChecker.dll中的类名。

4.2 RSL多模块支持

按照RAISE开发方法[5]，一个用RSL规范描述的系统可以由多个RSL模块组成，这些模块可以分布在多个不同的路径下，模块之间存在复杂的调用关系。这给RRTChecker系统的运行带来了困难，需要调用其它RSL模块时，Web服务器无法到客户端的硬盘上读取。

为了解决这些问题，使用了SessionID标识用户，并使用路径重写技术修改路径信息：

1)对客户端的每一次请求，检查SessionID。如果是新的SessionID，建立一个临时文件目录，作为该SessionID的临时根目录；

2)如果浏览器输入窗口的RSL模块是从文件读入的，检查文件信息，在该SessionID临时根目录中建立相应目录；

3)分析RSL模块中对其它模块的引用信息，将其路径部分影射为临时根目录下的对应目录。

4)将RSL模块存入相应目录。调用shell拦截程序开始工作。

同时，不得不对用户的工作方式作出一些限制，包括：用户RSL模块不能存放在多个硬盘上；必须由最底层模块开始检查，逐步检查到高层模块等。

结束语 基于Web的RAISE工具RRTChecker于2002年在联合国大学国际软件研究所(UNU/IIST)开发完成。经UNU/IIST和贵州科学院数年来的实际应用表明，该系统使用简单，升级容易(换成新版本的rsltc.exe)，特别适用于学习掌握RAISE和开发中小型的RAISE系统。RRTChecker可在Intranet和Internet上使用。该系统在互联网上位于http://www.gzas.org/rslchecker。

以原有的形式化方法工具为基础，通过Web技术，将工具的所有功能集成整合到统一的、用户友好的Web界面上。这种低成本的形式化方法工具开发方法和技术同样可以适用于其它形式化方法，具有广泛的应用前景。

参 考 文 献

[1] Formal Methods Education Resources Tool Pages, Department of Computer Science, Worcester Polytechnic Institute. http://www.cs.indiana.edu/formal-methods-education/Tools

[2] George C, et al. The RAISE Specification Language, CRI A/S, Denmark 1992

[3] George C. The development of the RAISE tools. Lecture Notes in Computer Science, Berlin: Heidelberg, Springer, 2003, 2757

[4] Li Dan, Bernhard K, Aichernig. Combining Algebraic and Model Based Test Case Generation. Lecture Notes in Computer Science, Berlin, Heidelberg, Springer, 2005, 3407

[5] George C, et al. The RAISE Development Method. Prentice Hall, 1995