基于 BHO(Browser Helper Object)的网址过滤实现①

魏景东

(中国石化 洛阳分公司信息中心, 洛阳 471012)

摘 要: 网址过滤(Filtering URL)是信息安全方的一个重要课题, 研究了浏览器(IE)辅助对象 BHO(Browser Helper Object)的工作机制, 并通过使用 BHO(Browser Helper Object)对象过滤可疑网址这个试例来全面介绍 BHO(Browser Helper Object)对象组件的工作机制及使用 Delphi7.0 开发实现过程, 同时也给出了使用 BHO 过滤网址的一种有效简便方法.

关键词: BHO(Browser Helper Object); ATL COM; ActiveX; Delphi7.0; IUnknown; Idispatch; IobjectWithSite

Implementation of Filtering URL Based on BHO(Browser Helper Object)

WEI Jing-Dong

(Sinopec Luoyang Branch Information Center, Luoyang 471012, China)

Abstract: Filtering URL is an important problem on information security. This article has researched working-machanism of BHO, and introduced technology implementation of BHO on filtering unlawful URL using the Delphi7.0, and gave an effective method of Filtering URL by BHO.

Key words: BHO(Browser Helper Object); ATL COM; ActiveX; Delphi7.0; IUnknown; Idispatch; IobjectWithSite

1 引言

BHO(Browser Helper Object)是微软早在1999 年推出的作为浏览器对第三方程序员开放交互接口的业界标准. 通过 BHO 接口, 第三方程序员可获取浏览器的行为和事件、获取浏览器的界面信息、实现浏览器的大部分功能. BHO 组件必须依靠浏览器才能运行. IE(4.0 以上版本)实例启动时会自动加载已注册的 BHO 组件, 并对其进行初始化. BHO 实例与 IE 实例运行在相同的内存上下文中, 有相同的生命周期[1].

作为一种 ATL COM 对象, BHO 对象由 IE 在启动时自动加载, 运行在 IE 的地址空间内, 能对 IE 中可访问的各类事件消息进行监听并作出相应处理, 即能监视浏览器(IE)的动作并作出相应处理, 当 IE 关闭时, BHO 对象停止运行[2]. 借助 BHO 对象, 可扩展 IE 的功能, 使 IE 功能更加人性化, 也可对用户上网进行必要的管理. IE 加载 BHO 的过程如图 1 所示.

BHO 通过简单的代码就可以进入浏览器领域的“交互接口”(INTERACTIVED Interface). 通过这个接口, 程序员可以编写代码获取浏览器的行为, 比如“后退”、“前进”、“当前页面”等, 利用 BHO 的交互特性, 程序员还可以用代码控制浏览器行为, 比如修改替换浏览器工具栏, 添加自己的程序按钮等. BHO 原来的目的是为了更好的帮助程序员打造个性化浏览器, 以及提供更简洁的交互功能, 现在很多 IE 个性化工具就是利用 BHO 来实现, 还能够安装钩子以监控一些消息和动作[3]. 国内一些大型商业银行网站快捷查询服务安全插件就是基于 BHO 技术实现的.

下面通过使用 Delphi7.0 开发一个过滤可疑网址的 BHO 插件来介绍浏览器(IE)辅助对象 BHO 的开发实现应用过程及主要技术细节.

2 BHO对象的开发实现应用

2.1 BHO 对象的开发实现

在 Delphi7 中, 新建 ActiveX Library 项目 COM_1.

① 收稿时间:2012-09-23;收到修改稿时间:2012-10-16

再在 COM_1 项目中新建 COM Object, 命名为 IEMonitor(如图 2 所示), 将图 2 所示 Options 项的 Include Type Library 前选钩去掉, 如图 2 所示.

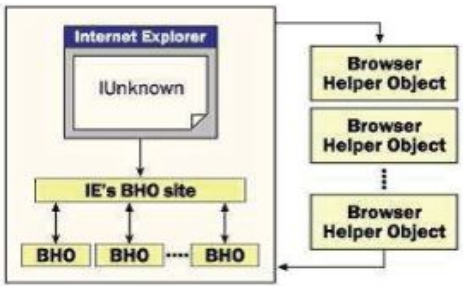

图 1 浏览器调用 BHO 工作原理

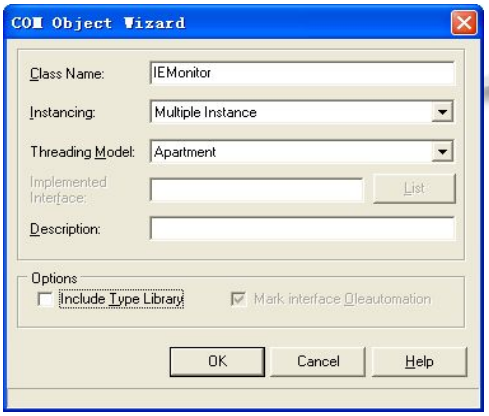

图 2 BHO 对象创建示意图

点击图 2 的 OK 按钮, 生成 BHO 对象插件的原始架构[4], 将 ActiveX Library 项目 COM_1 保存为项目名 COM_1.DPR, 将生成 BHO 对象插件的原始架构单元编码保存为 com.pas. 至此, 该 BHO 对象插件原始架构基本形成.

COM_1.DPR 的编码仅含有注册信息, 如下:

```
library Com_1;
uses
  ComServ,
  Com in 'Com.pas';
exports
  DllGetClassObject,
  DllCanUnloadNow,
  DllRegisterServer,
  DllUnregisterServer;
{$R *.RES}
Begin
end.
```

BHO 对象插件的主要功能在 com.pas 中实现, 要对 com.pas 的编码进行的扩充. 作为特殊的 COM 对象, IEMonitor 必须实现同浏览器 IE 通讯的两个接口 IObjectWithSite[6]和 Idispatch[7]. 下面分别介绍一下这两个接口:

(1) IObjectWithSite 接口

IObjectWithSite 接口用来挂钩和监控浏览器事件, IE 在加载 BHO 时, 会将自己的 IUnknown 接口[7]用 pUnkSite 参数传给 BHO 对象 IEMonitor. 通过对参数 pUnkSite 的解析即可获得浏览器 IE 接口 IWebBrowser2[5], 获得 IWebBrowser2 后, 又可得到浏览器事件连接点接口, 再使用该接口的 Advise 方法, 便可实现对浏览器事件的监听. IObjectWithSite 接口包含 GetSite 和 SitSite 方法, 其中由 SetSite 实现 IObjectWithSite 接口的主要功能.

(2) IDispatch 接口

IDispatch 接口主要用来对浏览器 IE 事件进行处理, 每当浏览器 IE 有事件发生时, IE 就会调用 IDispatch 接口的 Invoke 方法通知事件类型及参数, 并请求 BHO 对象 IEMonitor 对事件进行处理. 因此在 IDispatch 接口中 Invoke 方法是最重要的方法, BHO 对象 IEMonitor 的功能基本上都在 Invoke 方法中实现, 在 Invoke 方法中, Params 参数包含了被激发的事件所包含的参数数目及参数值, 过程 BuildPositionalDispIds 从 Params 参数中提取参数值, 并放到数组中. 至于 IDispatch 接口的其它方法 GetTypeInfoCount、GetTypeInfo 和 GetDsOfNames, 则只需返回 E_NOTIMPL 即可.

因此, BHO 对象 IEMonitor 在 com.pas 中的定义要扩充如下:

```
TIEMonitor=class(TComObject,IDispatch,IObjectWithSite)
  Public
    function GetTypeInfoCount(out Count:Integer):
HResult;stdcall;
    function GetTypeInfo(Index,LocaleID:Integer;
out TypeInfo):HResult;stdcall;
    function GetIDsOfNames(const IID:TGUID;
Names:Pointer;NameCount,LocaleID:Integer;DispIDs:Po-
inter):HResult;stdcall;
    function SetSite(const pUnkSite:IUnknown):
HResult;stdcall;
    function GetSite(const riid:TIID;out site:IUn-
```

Known):HResult;stdcall;

function Invoke(DispID:Integer;const IID:TGUID;LocaleID:Integer;Flags:Word;var Params;var Result,ExcepInfo,ArgErr:Pointer):HResult;stdcall;

private

IEThis:IWebBrowser2; //从 IE 传递过来的 IE 接口

Cookie:Integer;

Protected

end;

com.pas 的 Interface 部分要 uses CoMobj, SHDOCVW,Registry,ComServ,ComConst, 并声名注册表变量 var reg:Tregistry;用于 BHO 组件注册.

IE 加载 IEMonitor 后, IEMonitor 工作过程: 首先通过该 BHO 对象初始化编码向注册表项 HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\explorer\Browser Helper Objects\ 注册自己, 通过 IEMonitor 的 SetSite()方法将 IE 接口赋给 IEMonitor 的 IEthis, 通过 Invoke()方法对 IE 事件作出反应, IE 关闭时自动退出运行. 有多少个 IE 实例就有多少个 IEMonitor.

2.2 插件 **IEMonitor** 的主要方法过程实现关键技术如下:

(1) procedure DoBeforeNavigate2(const pDisp: IDispatch;var URL:OleVariant;

Var Flags: OleVariant; varTargetFrameName: OleVariant; var PostData:OleVariant;

var Headers:OleVariant;var Cancel:WordBool);过程

功能: IE 打开新网址事件处理过程.

处理过程: 打开可疑网址列表文件, 循环读取文件 URLFile 中的可疑网址, 同 IE 当前打开网址比较, 如相同, 则转向指定网址(假设 http://10.123.15.63).

该过程关键编码

```
var s:String;
URLFile:TextFile;
…
readln(URLFile,s);      //读取可疑网址列表文件
if(Trim(URL)=trim(s)) then   //发现要过滤的可疑网址
    begin
        cancel:=true;
        url:='http://10.123.15.63';
        (pDisp as IWebbrowser2).Navigate2(URL, Flags,TargetFrameName,PostData,Headers);   //转向指定网址
    end;
```

(2) procedure BuildPositionalDispIDs(pDispIDs: PDispIDList;const dps:TDispParams);过程

功能: 从 Params 参数中提取被激发事件所包含的参数数目及参数值, 放入数组中.

该过程关键编码

```
var i:Longint;
Assert(pDispIDs<>nil);
for i:=0 to dps.cArgs-1 do pDispIDs^[i]:=dps.cArgs-1-i;
if(dps.cNamedArgs<=0) then Exit;
for i:=0 to dps.cNamedArgs-1 do pDispIDs^[dps.rgdispidNamedArgs^[i]]:=i;
```

(3) function TIEMonitor.Invoke(DispID:integer;const IID:TGUID;LocaleID:Integer;Flags:Word;var Params; VarResult,ExcepInfo,ArgErr:Pointer):HResult; IDispatch 接口的 Invoke 方法

功能: 根据接收 IE 事件种类作相应处理. 其核心编码如下:

```
……
case DispID of
    250: //网页打开事件 BeforeNavigate2
        begin
DoBeforeNavigate2(IDispatch(dps.rgvarg^[pDispIDs^[0]].dispVal),
POleVariant(dps.rgvarg^[pDispIDs^[1]].pvarVal)^,   //IE 当前网址
POleVariant(dps.rgvarg^[pDispIDs^[2]].pvarVal)^,
POleVariant(dps.rgvarg^[pDispIDs^[3]].pvarVal)^,
POleVariant(dps.rgvarg^[pDispIDs^[4]].pvarVal)^,
POleVariant(dps.rgvarg^[pDispIDs^[5]].pvarVal)^,
dps.rgvarg^[pDispIDs^[6]].pbool^);
                Result:=s_OK;
        end;
    end;          //end of case DispID of
```

(4) function TIEMonitor.GetSite(const riid:Tiid;out site:IUnknown):HResult;方法.

功能: 检测 IE 接口.

实现编码:

```
if(Assigned(IEThis))then
Result:=IEThis.QueryInterface(riid,site)
else Result:=E_FAIL;
```

(5) function TIEMonitor.SetSite(const pUnkSite: IUnknown):HResult; 方法.

功能: 使 IEMonitor 与 IE 通过 Advise 方法检测 IE 事件. 该方法涉及几个接口及转换, 给出完整编码:

```
var
    cmdTarget:IOleCommandTarget; //[6]
    Sp:IServiceProvider;   //[6]
    CPC:IConnectionPointContainer; //[6]
    CP:IConnectionPoint;    //[6]
begin
  if(Assigned(pUnkSite)) then
    begin
        cmdTarget:=(pUnkSite as IOleCommandTarget);
        Sp:=(CmdTarget as IServiceProvider);
       if (Assigned(Sp))then //获得 IE WebBrowse 接口
    Sp.QueryService(IWebBrowserApp,IWebBrowser2,
IEThis);
        if(Assigned(IEThis))then
          begin
    IEThis.QueryInterface(IConnectionPointContainer,
CPC);//寻找连接点
    CPC.FindConnectionPoint(DWEBBrowserEvents2,
CP);
    CP.Advise(Self,Cookie);//通过 Advise 方法检测 IE
事件
          end;
    end;
    Result:=S_OK;
end;
```

该 BHO 对象初始化编码:

```
Initialization
  TComObjectFactory.Create(ComServer, TIEMonitor,
Class_IEMonitor,
    'IEMonitor', '', ciMultiInstance, tmApartment);
    reg:=TRegistry.Create;  //向注册表添加自己的 Guid
字符串关键字
    reg.RootKey:=HKEY_LOCAL_MACHINE;
    reg.OpenKey('Software\Microsoft\Windows\Current
Version\explorer\Browser Helper Objects\'+GuidToString
(Class_IEMonitor),true);
    reg.Free;
end.
```

最后, 利用 Delphi7.0 开发工具将上述项目 COM_1.DPR 制作成 BHO 对象插件 COM_1.DLL.

2.3 BHO 对象插件的应用

同所有 COM 对象一样, BHO 需使用 regsvr32 进行注册或卸载[8], 还需将自己的 Guid 字符串关键字添加到注册表

HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\explorer\Browser Helper Object\ 下, 该项工作可手工创建, 该试例在 BHO 中用注册表对象直接创建.

另外要创建 c:\MyIEBHO.txt 文件, 里面输入要过滤的可疑网址.

最后运行 regsvr32 COM_1.dll 进行注册或运行 regsvr32 COM_1.dll /u 以注销. 注册成功后, 重新运行 IE, 在地址栏中完整地输入待过滤的可疑网址, 即可发现 IE 直接转向 http://10.123.15.63. 在 Microsoft windows /XP 系统下测试, 该 BHO 注册成功后, 打开 IE 时, BHO 组件内主要函数的运行次序: (1) 启运 SetSite. (2) 启运 GetSite. (3) IE 中只要有激发事件就启运 Invoke. (4) 关闭 IE 时启运 SetSite.

3 结语

上述试例在 Microsoft windows /XP 下成功运行, BHO 插件 COM_1.dll 的功能还可进一步扩充以用于网吧青少年的上网管理及办公电脑的安全管理. 该项研究工作的理论创新点在于对 BHO 工作机制的简明形象刨析, 工程创新性在于将 BHO 技术非常具体的应用于解决信息安全方面的现实问题. 该试例的开发看上去复杂, 涉及多个接口的实现, 实际上 BHO COM 插件的开发基本上是模板化的, 真正体现创意和功力的主要集中于事件的处理编码上. 该试例全面含概了 BHO COM 插件开发应用的全过程及主要技术细节, 供从事 BHO COM 插件开发的技术人员参考.

(下转第 169 页)

表 2 首次订购用户验证结果

目标	实际用户数	预测准确用户数	准确率	合计
是否消费	3000（x>0）	2030	67%	77%
	3000（x=0）	2587	86%	
消费天数	3000（x>0）	1523	51%	69%
	3000（x=0）	2587	86%	

4.2 有订购历史行为用户数据验证

选取数据说明:

1) 样本用户: 考察期之前无订购行为、2011.11.1 至 2011.11.10 期间首次按章订购的用户.

2) 观察期: 首次订购时间至 2011.12.31.

3) 预测期: 2012.1.1 至 2012.1.31.

结果说明:

抽取 6000 个用户（3000 个 1 月实际有交易行为的用户以及 3000 个 1 月实际无交易行为的用户），按照上述组合 2 种情况分别进行验证, 发现-按章订购次数的情况, 模型预测效果不甚理想, 此种情况不适合用该模型, 具体结果不再展示.

准确性定义:

准确性定义同首次订购用户, 这里不再赘述.

数据验证结果如表 3 所示.

表 3 有订购历史行为用户验证结果

目标	实际用户数	预测准确用户数	准确率	合计
是否消费	3000（x>0）	2676	89%	77%
	3000（x=0）	1947	65%	
消费天数	3000（x>0）	1952	65%	65%
	3000（x=0）	1947	65%	

3 结果分析

根据验证结果显示, 依据不同的业务目标, 该模型能较好的抓住用户的付费意愿和付费次数. 通过客户价值预测, 可帮助业务关注者定位忠诚用户, 对该类客户进行进一步的细分和特征分析, 并能根据用户的其他业务特征制定有针对性的营销策略.

通过用户是否有消费意愿的预测, 对消费意愿减弱或意愿消失倾向的用户, 可提前制定措施有针对性的营销挽回, 同时结合付费次数预测, 在合理限值内进行营销以避免形成过打扰.

综上所述, 本文通过手机阅读领域用户数据验证了适当业务目标的定义下, 使用 BG/NBD 模型能够有效而且准确的进行客户价值预测, 对客户消费意愿进行预测, 定位忠诚用户群, 从而有针对性地进行营销能够起到很好的作用.

参考文献

1 张春莲. BG/NBD 模型对客户购买行为的预测分析.时代经贸,2008,6(97):51–52.

2 王永贵,董大海.客户关系管理的研究现状、不足和未来展望.中国流通经济,2004,(6):52–56.

3 周洁如.客户关系管理中的价值创造研究.上海管理科学,2003,(4):55–56.

4 Reichheld F. Learning from Customer Defections. Harvard Business Review,1999,(2):56–69.

5 陈明亮.客户重复购买意向决定因素的实证研究.科研管理,2003,24(1):110–115.

6 万里,廖建新,王纯.基于社会网络信息流模型的协同过滤算法.吉林大学学报,2011,41(1):270–275.

(上接第 139 页)

参考文献

1 王娟,郭永冲,王强.基于 BHO 的网络隐蔽通道研究.计算机工程,2009(5):159–161.

2 桑庆兵,吴小俊.基于 BHO 的网站过滤系统研究与实现.计算机工程与应用,2009,45(31):18–19.

3 http://baike.baidu.com/view/362533.htm.

4 东方人华,吕伟臣编.Delphi7.0 入门与提高.北京:清华大学出版社,2006,369–380.

5 ../Delphi7/Source/Internet/SHDocVw.pas[CP/OL]:478–580.

6 ../Delphi7/Source/rtl/win/ActiveX.pas[CP/OL]:3068–4813.

7 ../Delphi7/Source/rtl/sys/System.pas[CP/OL]:254–275.

8 魏志强,王忠华.程序设计 Delphi 5.0.北京:中国铁道出版社,2000.238–252.