在线分销产品
Q 信息处理服务用户手册

目录

1.概述

Q 信息服务是在线机票分销中不可或缺的环节，在线分销 Q 信息处理服务可提供实时的 Q 信息查询、删除、转移等操作服务，为代理人在线销售系统提供了快捷、简便、高效的途径。

2.版本

主要内容	Q 信息处理
参考文档	
创建部门	

批准单		
版本号：		
批准人	批准人签字	批准日期

文档控制

版本	修改日期	修改内容描述	作者	批准人	批准日期
V1.0.0	2013.05.22	创建	王文亚		
V1.0.1	2013.06.14	修改(添加多个 Agent 号)	王宏文		

3.业务范围

Q 信息处理对应于主机的信箱处理

4. 功能范围

本服务支持的功能如下：

1、Q 信息转移（对应于主机的 QC 指令）

2、向指定信箱发送信息（对应于主机的 QE 指令）

3、Q 信息综合查询（QS）

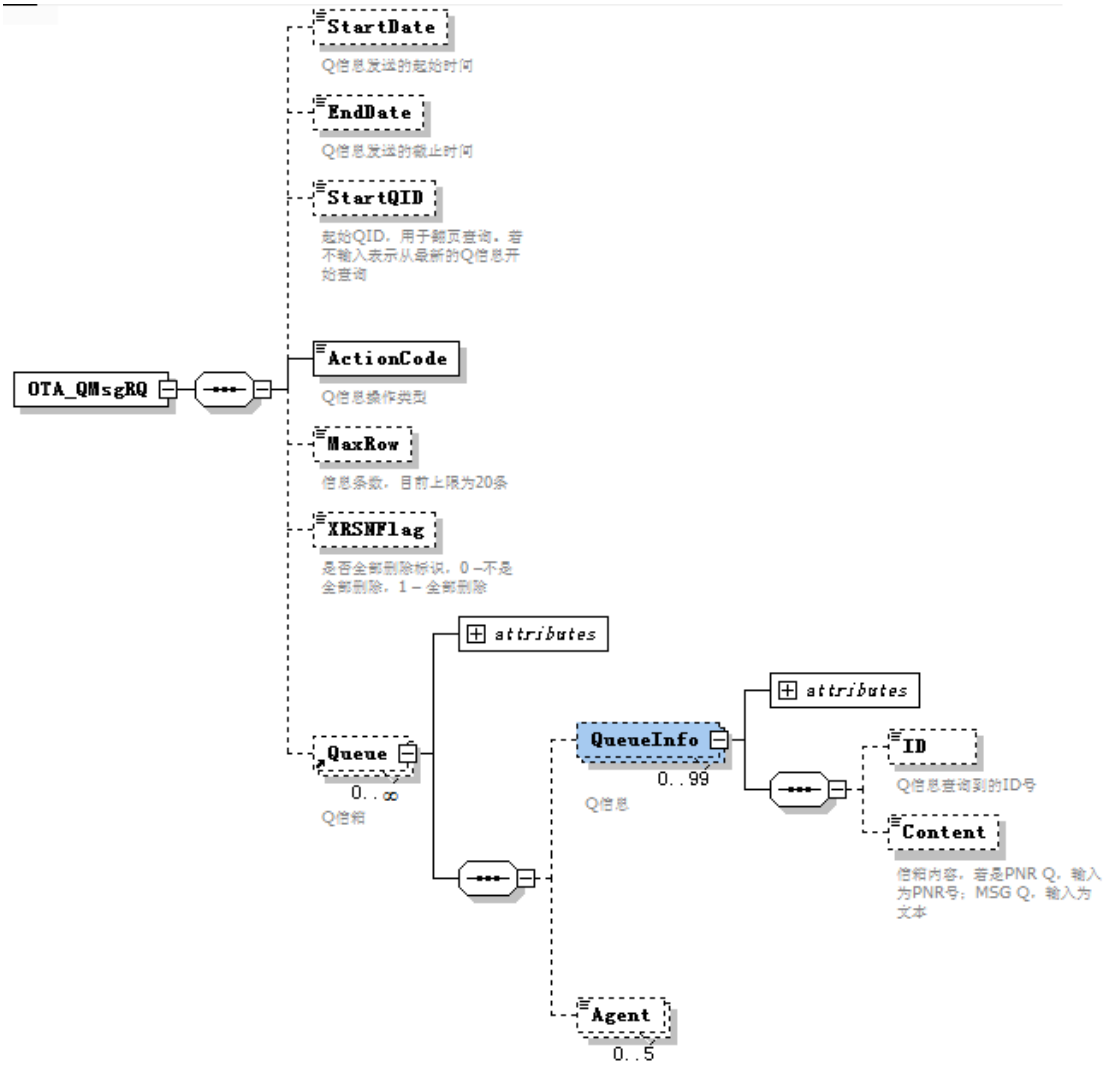
4、Q 信箱详细信息显示（对应于主机的 QT 指令）

5、清除一个 Queue 信息（对应于主机的 QN 指令）

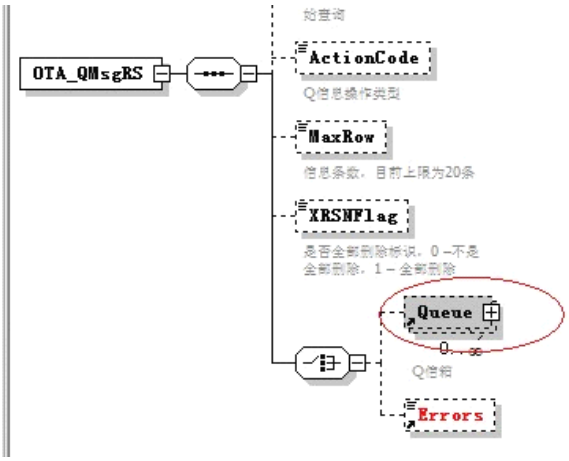
5.接口说明

5.1 接口结构

5.1.1 请求结构

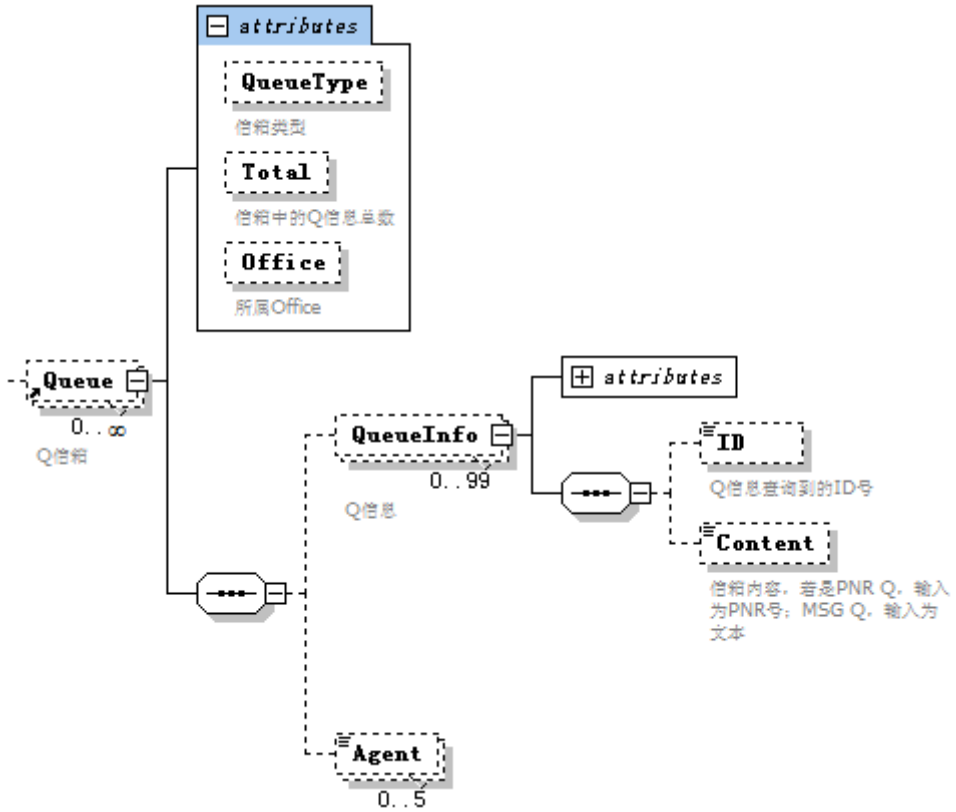

OTA_QMsgRQ
StartDate
Q信息发送的起始时间
EndDate
Q信息发送的截止时间
StartQID
起始QID，用于翻页查询，若不输入表示从最新的Q信息开始查询
ActionCode
Q信息操作类型
MaxRow
信息条数，目前上限为20条
XRSNFlag
是否全部删除标识，0－不是全部删除，1－全部删除
attributes
Queue
0..∞
Q信箱
attributes
QueueInfo
0..99
Q信息
ID
Q信息查询到的ID号
Content
信箱内容，若是PNR Q，输入为PNR号；MSG Q，输入为文本
Agent
0..5

5.1.2 响应结构



其中标识红色椭圆线的节点结构如下：

Queue 节点



5.2 请求参数说明

元素名称	说明	Rep	类型	默认值	备注
OTA_QMsgRQ		1			

@ActionCode	Q 信箱操作类型	1	String	
@StartQID	起始 QID	0..1	String	
@StartDate	Q 信息发送的起始时间	0..1	String	
@EndDate	Q 信息发送的截止时间	0..1	String	
@MaxRow	信息条数	0..1	String	目前上限为 20 条
@XRSNFlag	是否全部删除标识	0..1	String	0 不是全部删除，1 全部删除
Queue	Q 信箱	1..n		
@QueueType	信箱类型	1	String	
@Office	所属的 Office 号	1	String	
@Agent	所属的 Agent	0..5	String	目前支持最多 5 个 Agent 号
QueueInfo	Q 信息	0..n		
@Content	信箱内容	0.1	String	
@ID	ID	0..1	String	
@QType	信息类型	0..1	String	0 是 PNR Queue，8 是 MSG Queue

5.3 响应参数说明

元素名称	说明	Rep	类型	默认值	备注
OTA_QMsgRS		1			
@ActionCode	Q 信箱操作类型	1	String		
OTA_QMsgRSChoice	信箱的处理结果	1			
Queue		1..n			
@QueueType	信箱类型	1	String		
@Office	所属的 Office 号	1	String		
@Agent	所属的 Agent	0..1	String		
@Total	信箱中 Q 信息总数	0..1	long		
Agent	Office 下的 Agent 号	0..5	String		目前可支持最多输入 5 个 Agent 号
QueueInfo	Q 信息	0..n			
@Content	信箱内容	0..1	String		
@QPid	进行 queue 操作的	0..1	String		
@QType	信息类型	0..1	String		0 是 PNR Queue，8 是 MSG Queue
@QIType	Queue item 类型	0..1	String		仅取值 1 或 2。1 是 normal，2 是

				priority
@PNRDate	生成 PNR 的日期	0..1	Date	
@QDate	进行 queue 操作的日期和时间	0..1	Date	
Errors	错误信息集合	0..1		
Error	错误信息	0..1		
@ShortText	错误详细信息	0..1	String	
@Code	错误代码	0..1	String	
@StartQID	起始 QID	0..1	String	
@StartDate	Q 信息发送的起始时间	0..1	String	
@EndDate	Q 信息发送的截止时间	0..1	String	
@MaxRow	信息条数	0..1	String	目前上限为 20 条

6. 错误说明

6.1http 协议错误

由于接口为 HTTP 接口，HTTP 协议会返回 403 错误：

http 错误代码	说明
403	用户名、密码、IP 错误或无权限

6.2 接口错误

<Errors>节点记录了错误信息。错误信息分为两类：业务错误、系统错误。详细见下表：

系统错误

错误代码	**错误描述**	**说明**
-303	Inner service error!	内部服务错误

业务错误

错误代码	**错误描述**	**说明**
-1	Biz Error	业务异常
-301	office prermission denied	没有 office 权限
-400	format of xml string is not valid	xml 内容格式错误
-402	input error	输入错误

7. 参考样例

输入请求中的参数说明如下：

```
<OTA_QMsgRQ>
    <!--ActionCodeQ 信息操作类型 -->
    <ActionCode>OPQC</ActionCode>
    <!--XRSNFlag 信息操作类型 -->
    <XRSNFlag>1</XRSNFlag>
    <!--Queue,Q 信箱,可以有多个该节点 ， 里面的子节点请参照输入请求节点说明-->
    <Queue QueueType="DO" Office="BJS191">
    <!--QueueInfo,Q 信息节点  -->
        <QueueInfo>
            <ID>1830710085</ID>
        </QueueInfo>
    </Queue>
    <Queue QueueType="BE" Office="BJS191" />
</OTA_QMsgRQ>
```

具体的样例如下：

7.1 转移信息（QC）

RQ:

```
<OTA_QMsgRQ>
    <ActionCode>OPQC</ActionCode>
    <Queue QueueType="DO" Office="BJS191">
        <QueueInfo>
            <ID>1830710085</ID>
        </QueueInfo>
    </Queue>
    <Queue QueueType="BE" Office="BJS191" />
</OTA_QMsgRQ>
```

RS:

```
<OTA_QMsgRS>
    <ActionCode>OPQC</ActionCode>
    <Queue QueueType="DO" Office="BJS191">
        <QueueInfo>
            <ID>1830710085</ID>
        </QueueInfo>
    </Queue>
    <Queue QueueType="BE" Office="BJS191" />
</OTA_QMsgRS>
```

7.2 删除全部信箱信息（QX）

RQ:

```
<OTA_QMsgRQ>
    <ActionCode>OPQX</ActionCode>
    <XRSNFlag>1</XRSNFlag>
    <Queue QueueType="DO" Office="BJS191" >
        <Agent>17856</Agent>
        <Agent>88888</Agent>
    </Queue>
</OTA_QMsgRQ>
```

RS:

```
<OTA_QMsgRS>
    <ActionCode>OPQX</ActionCode>
    <XRSNFlag>1</XRSNFlag>
    <Queue QueueType="DO" Office="BJS191" />
</OTA_QMsgRS>
```

7.3 删除部分信箱信息（QX）

RQ:

```
<OTA_QMsgRQ>
    <ActionCode>OPQX</ActionCode>
    <XRSNFlag>0</XRSNFlag>
    <Queue QueueType="DO" Office="BJS191" >
        <QueueInfo>
            <QID>187645</QID>
        </QueueInfo>
    </Queue>

</OTA_QMsgRQ>
```

RS:

```
<OTA_QMsgRS>
    <ActionCode>OPQX</ActionCode>
    <XRSNFlag>0</XRSNFlag>
    <Queue QueueType="DO" Office="BJS191" />
</OTA_QMsgRS>
```

7.4 查询信箱的详情（QR）

RQ:

```
<OTA_QMsgRQ>
    <ActionCode>OPQR</ActionCode>
    <MaxRow>2</MaxRow>
    <Queue QueueType="DO" Office="BJS191" >
        <Agent>17856</Agent>
        <Agent>88888</Agent>
    </Queue>
</OTA_QMsgRQ>
```

RS:

```
<OTA_QMsgRS>
    <StartDate>1970-01-01T00:00:00</StartDate>
    <EndDate>2111-01-01T00:00:00</EndDate>
    <StartQID>9999999999999999</StartQID>
    <ActionCode>OPQR</ActionCode>
    <MaxRow>2</MaxRow>
    <Queue>
        <QueueInfo QType="1" QIType="1" QAgent="68816" QPid="104"
            QDate="2011-11-12T23:05:08"                    PNRAgent="68816"
PNRDate="2005-12-11T00:00:00">
            <ID>1829641086</ID>
            <Content>HX3XPK</Content>
        </QueueInfo>
        <QueueInfo QType="1" QIType="1" QAgent="68816" QPid="104"
            QDate="2011-11-12T23:05:08"                    PNRAgent="68816"
PNRDate="2005-12-11T00:00:00">
            <ID>1829641085</ID>
            <Content>HX3XNE</Content>
        </QueueInfo>
    </Queue>
</OTA_QMsgRS>
```

7.5 查询信箱中的消息数量（QT）

RQ:

```
<OTA_QMsgRQ>
```

```
    <ActionCode>OPQT</ActionCode>
    <Queue Office="BJS191" >
        <Agent>17856</Agent>
        <Agent>88888</Agent>
    </Queue>

</OTA_QMsgRQ>
```

RS:

```
<OTA_QMsgRS>
    <ActionCode>OPQT</ActionCode>
    <Queue QueueType="GQ" Total="2" Office="BJS191" Agent="" />
    <Queue QueueType="RP" Total="47" Office="BJS191" Agent="" />
    <Queue QueueType="HT" Total="0" Office="BJS191" Agent="" />
    <Queue QueueType="KK" Total="410" Office="BJS191" Agent="" />
    <Queue QueueType="MT" Total="0" Office="BJS191" Agent="" />
    <Queue QueueType="RE" Total="0" Office="BJS191" Agent="" />
    <Queue QueueType="SR" Total="2048" Office="BJS191" Agent="" />
    <Queue QueueType="TC" Total="1895" Office="BJS191" Agent="" />
    <Queue QueueType="TL" Total="5106" Office="BJS191" Agent="" />
    <Queue QueueType="SC" Total="581" Office="BJS191" Agent="" />
    <Queue QueueType="IB" Total="0" Office="BJS191" Agent="" />
    <Queue QueueType="DS" Total="0" Office="BJS191" Agent="" />
    <Queue QueueType="QX" Total="0" Office="BJS191" Agent="" />
    <Queue QueueType="QY" Total="0" Office="BJS191" Agent="" />
    <Queue QueueType="QZ" Total="0" Office="BJS191" Agent="" />
    <Queue QueueType="QV" Total="0" Office="BJS191" Agent="" />
    <Queue QueueType="QW" Total="0" Office="BJS191" Agent="" />
    <Queue QueueType="LU" Total="0" Office="BJS191" Agent="" />
    <Queue QueueType="EP" Total="0" Office="BJS191" Agent="" />
    <Queue QueueType="JA" Total="0" Office="BJS191" Agent="" />
    <Queue QueueType="BE" Total="8" Office="BJS191" Agent="" />
    <Queue QueueType="IN" Total="0" Office="BJS191" Agent="" />
    <Queue QueueType="PH" Total="0" Office="BJS191" Agent="" />
    <Queue QueueType="DO" Total="0" Office="BJS191" Agent="" />
    <Queue QueueType="IS" Total="29" Office="BJS191" Agent="" />
    <Queue QueueType="NO" Total="0" Office="BJS191" Agent="" />
    <Queue QueueType="ON" Total="0" Office="BJS191" Agent="" />
</OTA_QMsgRS>
```

黑屏查询如：

```
                QT BJS191
GQ 0004 5000    RP 0046 0453    HT 0002 0999    KK 0423 3375
MT 0000 0200    RE 0000 0452    SR 2048 3078    TC 1896 2928
TL 5106 7840    SC 0606 0793    IB 0000 1890    DS 0000 1000
QX 0000 0702    QY 0000 3457    QZ 0000 0282    QV 0000 0942
QW 0000 2727    LU 0000 0970    EP 0000 2167    JA 0000 0290
BE 0011 0200    IN 0000 0200    PH 0000 0200    DO 0003 0200
IS 0029 0805    NO 0000 0200    ON 0000 0852
```

7.6 查询指定信箱（KK）消息数量（QT）

RQ：

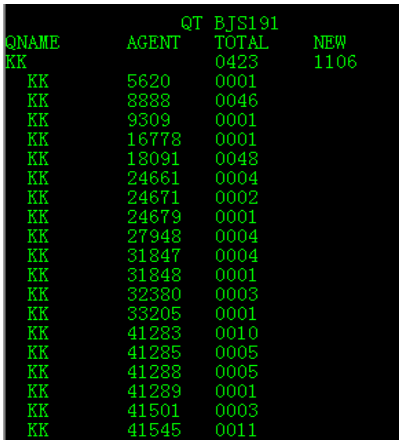
```
<OTA_QMsgRQ>
    <ActionCode>OPQT</ActionCode>
        <Queue QueueType="KK" Office="BJS191"/>

</OTA_QMsgRQ>
```

RS:

```
<OTA_QMsgRS>
    <ActionCode>OPQT</ActionCode>
    <Queue QueueType="KK" Total="1" Office="BJS191" Agent="5620" />
    <Queue QueueType="KK" Total="35" Office="BJS191" Agent="8888" />
    <Queue QueueType="KK" Total="1" Office="BJS191" Agent="9309" />
    <Queue QueueType="KK" Total="1" Office="BJS191" Agent="16778" />
    <Queue QueueType="KK" Total="48" Office="BJS191" Agent="18091" />
    <Queue QueueType="KK" Total="4" Office="BJS191" Agent="24661" />
    <Queue QueueType="KK" Total="2" Office="BJS191" Agent="24671" />
    <Queue QueueType="KK" Total="1" Office="BJS191" Agent="24679" />
    <Queue QueueType="KK" Total="4" Office="BJS191" Agent="27948" />
    <Queue QueueType="KK" Total="4" Office="BJS191" Agent="31847" />
    <Queue QueueType="KK" Total="1" Office="BJS191" Agent="31848" />
    <Queue QueueType="KK" Total="3" Office="BJS191" Agent="32380" />
    <Queue QueueType="KK" Total="10" Office="BJS191" Agent="41283" />
    <Queue QueueType="KK" Total="5" Office="BJS191" Agent="41285" />
    <Queue QueueType="KK" Total="5" Office="BJS191" Agent="41288" />
    <Queue QueueType="KK" Total="1" Office="BJS191" Agent="41289" />
    <Queue QueueType="KK" Total="3" Office="BJS191" Agent="41501" />
    <Queue QueueType="KK" Total="10" Office="BJS191" Agent="41545" />
    <Queue QueueType="KK" Total="232" Office="BJS191" Agent="46692" />
    <Queue QueueType="KK" Total="14" Office="BJS191" Agent="46861" />
    <Queue QueueType="KK" Total="5" Office="BJS191" Agent="68917" />
    <Queue QueueType="KK" Total="5" Office="BJS191" Agent="71936" />
    <Queue QueueType="KK" Total="5" Office="BJS191" Agent="75699" />
    <Queue QueueType="KK" Total="5" Office="BJS191" Agent="78845" />
```

```
    <Queue QueueType="KK" Total="5" Office="BJS191" Agent="78850" />
</OTA_QMsgRS>
```

结果中包含该 OFFICE 下所有 Agent 的 KK 信箱中的消息数。黑屏查询如：



7.7 向信箱中添加消息（QE）

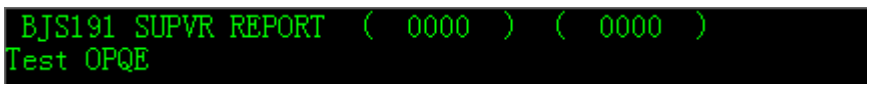
RQ：

```
<OTA_QMsgRQ>
<ActionCode>OPQE</ActionCode>
<Queue QueueType="RP" Office="BJS191"/>
<Queue QueueType="RP" Office="BJS191">
    <QueueInfo QType="8">
        <Content>Test OPQE</Content>
    </QueueInfo>
</Queue>
</OTA_QMsgRQ>
```

RS：

```
<OTA_QMsgRS>
    <ActionCode>OPQE</ActionCode>
    <Queue QueueType="RP" Office="BJS191" />
<QueueInfo>
            <ConTent>Test OPQE<</ConTent>
    </QueueInfo>
</OTA_QMsgRS>
```

通过黑屏查询信息：



8. 接口调用

Q 信息处理服务提供基于 XML 文档规范的标准 HTTP 协议接口，通过在标准 HTTP 通道上传输 XML 格式的请求和数据来完成服务调用。服务的输入输出定义在标准的 XSD 文档中，用户根据自有系统的实现环境结合 XSD 文档进行数据到对象间的转换。在线分销提供服务接口 URL,用户使用 HTTP 协议按照服务定义正确构造请求输入，正确解析数据输出实现服务的调用。

8.1 接口申请

请与您所在城市的中国航信分支机构联系，资质审查通过后签署试用协议，即可获得试用接口。

8.2 接口认证

授权用户调用试用接口时需要使用已分配的账号和密码、并使用申请时 的 IP。如果已变化 IP 请联系业务人员申请变更 IP。

8.3 接口地址

接口描述位置：

http://espeed.travelsky.com/develop/xml/QMsg?xsd

接口试用地址：

http://espeed.travelsky.com/develop/xml/QMsg

8.4 JAVA 调用示例

Java 调用 http 接口示例如下：

其中该样例程序只是为了展示如何接入使用

```
import java.io.BufferedReader;
import java.io.ByteArrayOutputStream;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.util.zip.GZIPInputStream;
```

```
import java.util.zip.GZIPOutputStream;

import org.apache.commons.httpclient.Credentials;
import org.apache.commons.httpclient.DefaultHttpMethodRetryHandler;
import org.apache.commons.httpclient.HttpClient;
import org.apache.commons.httpclient.HttpState;
import org.apache.commons.httpclient.HttpStatus;
import org.apache.commons.httpclient.UsernamePasswordCredentials;
import org.apache.commons.httpclient.auth.AuthScope;
import org.apache.commons.httpclient.methods.ByteArrayRequestEntity;
import org.apache.commons.httpclient.methods.PostMethod;
import org.apache.commons.httpclient.methods.RequestEntity;
import org.apache.commons.httpclient.params.HttpMethodParams;

/**
 * http 客户端调用示例(数据压缩版)
 */
public class HttpClientDemo {

    public static void main(String[] args) {

        //实际使用时，请设置正确的变量值
        //用户名
        String username = "用户名";
        //密码
        String pwd = "密码";
        //服务地址
        String serviceUrl = "服务地址";
        //请求 XML
        String reqStr = "请求 xml";

        // 构造 HttpClient 的实例
        HttpClient httpClient = new HttpClient();
        //调用验证信息
        HttpState state = new HttpState();
        Credentials credentials = new UsernamePasswordCredentials(username, pwd);
        state.setCredentials(AuthScope.ANY, credentials);
        httpClient.setState(state);

        // 创建 POST 方法的实例
        PostMethod postMethod = new PostMethod(serviceUrl);
        // 使用系统提供的默认的恢复策略
        postMethod.getParams().setParameter(HttpMethodParams.RETRY_HANDLER,
                new DefaultHttpMethodRetryHandler());
```

```
try {
    // 请求参数的数据压缩
    ByteArrayOutputStream out = new ByteArrayOutputStream();
    GZIPOutputStream gzip = new GZIPOutputStream(out);
    if (reqStr != null && !"".equals(reqStr)) {
        gzip.write(reqStr.getBytes());
    }
    gzip.close();
    RequestEntity requestEntity = new ByteArrayRequestEntity(out
            .toByteArray());
    postMethod.setRequestEntity(requestEntity);
    postMethod.addRequestHeader("Content-Type",
            "text/html;charset=UTF-8");
    postMethod.addRequestHeader("accept-encoding", "gzip");
    postMethod.addRequestHeader("content-encoding", "gzip");

    // 执行 getMethod
    int statusCode = httpClient.executeMethod(postMethod);
    if (statusCode != HttpStatus.SC_OK) {
        throw new Exception("Invoke Get Method Failed, HttpStatus = "
                + statusCode);
    }

    // 返回结果的数据解压
    InputStream is = postMethod.getResponseBodyAsStream();
    BufferedReader br = new BufferedReader(new InputStreamReader(
            new GZIPInputStream(is)));

    StringBuffer respStr = new StringBuffer();
    String line = null;
    while ((line = br.readLine()) != null) {
        respStr.append(line);
    }

    // 打印结果
    System.out.println(respStr.toString());

    out.flush();
    out.close();
    is.close();
} catch (Exception e) {
    // 发生致命的异常，可能是协议不对或者返回的内容有问题
    e.printStackTrace();
} finally {
```

```
                // 释放连接
                postMethod.releaseConnection();
            }

        }
}
```

8.5.NET 调用示例

.Net 调用 http 接口示例如下：

其中该调用示例程序只是为了展示如何接入使用

```
(1)建立一个 cs 文件,与服务器通信的类：
// 定义 xml 与服务器的通信
class OTA_XMLHTTP
  {
        string user;        //用户名
        string pwd;    //密码
         string url;   //服务地址
        public OTA_XMLHTTP(string _user,string _pwd,string _url)
        {
            user = _user;
            pwd = _pwd;
            url = _url;
        }
        // 获取服务器返回的字符串
        public string GetResponse(string requestXml)
        {
            string xmlString = "";     //要返回的 xml 字符串
            bool a= requestXml.Contains("<?xml version="1.0" encoding="UTF-8"?>");
            requestXml = requestXml.Replace("<?xml version="1.0" encoding="UTF-8"?>\n", "");
            try
            {
                //制定服务器地址
                HttpWebRequest request = (HttpWebRequest)HttpWebRequest.Create(url);
                request.Method = WebRequestMethods.Http.Post; //设定 http 的传递方式
                request.ContentType = "application/x-www-form-urlencoded";
               //设定 http 的 header
               if (user != null && pwd != null)
               {
                    string user_pwd = user + ":" + pwd;
                    byte[] authBytes = Encoding.UTF8.GetBytes(user_pwd.ToCharArray());
```

```
                    request.Headers.Add("Authorization", "Basic " + Convert.ToBase64String(authBytes));
                    request.Headers.Add("Content-Encoding", "gzip");
                    request.Headers.Add("Accept-Encoding", "gzip");
                }
                //将数据写入流中
                string para = "request";
                ASCIIEncoding encoding = new ASCIIEncoding();
                byte[] data = encoding.GetBytes(para + " = " + requestXml);
                request.ContentLength = data.Length;
                Stream stream = request.GetRequestStream();
                stream.Write(data, 0, data.Length);
                stream.Close();
                //建立获取 http 返回的 response
                HttpWebResponse response = (HttpWebResponse)request.GetResponse();
                //读取 http 返回的字符串
            Stream responseStream = response.GetResponseStream();
            if (response.ContentEncoding.ToLower().Contains("gzip"))
                responseStream = new GZipStream(responseStream, CompressionMode.Decompress);
            else if (response.ContentEncoding.ToLower().Contains("deflate"))
                responseStream = new DeflateStream(responseStream, CompressionMode.Decompress);
            StreamReader streamReader = new StreamReader(responseStream, Encoding.UTF8);
                xmlString = streamReader.ReadToEnd();
                streamReader.Close();
            }
            catch (WebException e)
            {
                xmlString = e.Message;
            }
            return xmlString;
        }
    }
```

（2）调用上面的文件

调用方法：

```
        string user="用户名";
        string pwd="密码";
        string url="服务地址";
        OTA_XMLHTTP xmlHttp = new OTA_XMLHTTP(user, pwd, url);
        PonseXmlBox.Text = xmlHttp.GetResponse(requestXml);
```