全国公民身份信息系统
（NCIIS）

Web服务接口调用规范

全国公民身份证号码查询服务中心

2014年4月

目录

第一章　　概述

全国公民身份信息系统是以全国公安人口信息管理系统提供的数据为基础，通过互联网和行业专网向政府部门、社会各界、人民群众依法提供公民身份信息服务。公民身份信息核查是全国公民身份信息系统(NCIIS)的主要功能，是系统对外服务的核心业务。Web 服务接口为典型的行业客户提供了使用全国公民身份信息系统的访问接口，该接口透明的代理了 NCIIS 现有功能，为客户基于二次开发并将 NCIIS 现有功能集成到自身业务系统提供了技术基础。本 Web 服务接口的设计与实现遵循目前 Web Services 主流技术架构，满足开放系统的标准化规范要求。

第二章　　文档描述

本文档用于说明客户端调用全国公民身份信息系统 Web 服务接口规范。用于进行相关人口信息资源的核查访问。

如客户端使用 Java 语言实现，建议使用 xFire 实现，且版本(xFire1.2.6)需与服务端相同，调用方式参见第八章客户端调用说明示例。

如客户端使用非 Java 语言实现，具体实现方式请参见第三方开发语言技术文档。

1．接口描述文档 WSDL

如需获取客户接入服务平台 Web Services 的接口描述，可以访问以下网址：

https:// ipAddress/nciic_ws/services/NciicServices?wsdl

其中，ipAddress 为全国公民身份信息系统开通的客户接入服务器地址**(域名为：api.nciic.com.cn)**。

此 WSDL 文档描述了客户接入服务 WebServices 的全部细节和调用的方法。包括消息的格式、传输协议和服务地址等。

2．WSDL 文档的解析

作为服务请求者的客户为顺利与客户接入服务 WebServices 交互，必须首先获取该 WSDL 文档，并正确解析得到输入输出的各对象类。

WSLD 文档的获取，在 IE 地址中输入：

https:// ipAddress/nciic_ws/services/NciicServices?wsdl

将页面保存成一个文件（即*.wsdl）。因为采用的是 https 协议，所以需要对此文件内容进行微小的调整。通过文本编辑工具打开此文件，在文件内容的倒数第四行，即：

<wsdlsoap:address location="http://api.nciic.com.cn/nciic_ws/services/NciicServices" />，将 location 的值“http”修改成“https”，保存此文件即可。

不同的开发平台提供了不同的 WSDL 自动解析工具。如 VS.NET 中的 WSDL.exe 工具、Borland 公司的 Jbuilder、Bea 公司的 Weblogic Workshop 等。使用哪一种工具要视集团客户外部应用的系统平台和开发平台具体情况以及客户的使用习惯而定。

第三章　　术语和定义

3.1　缩略语

SOAP　　简单对象访问协议（Simple　Object　Access　protocol）

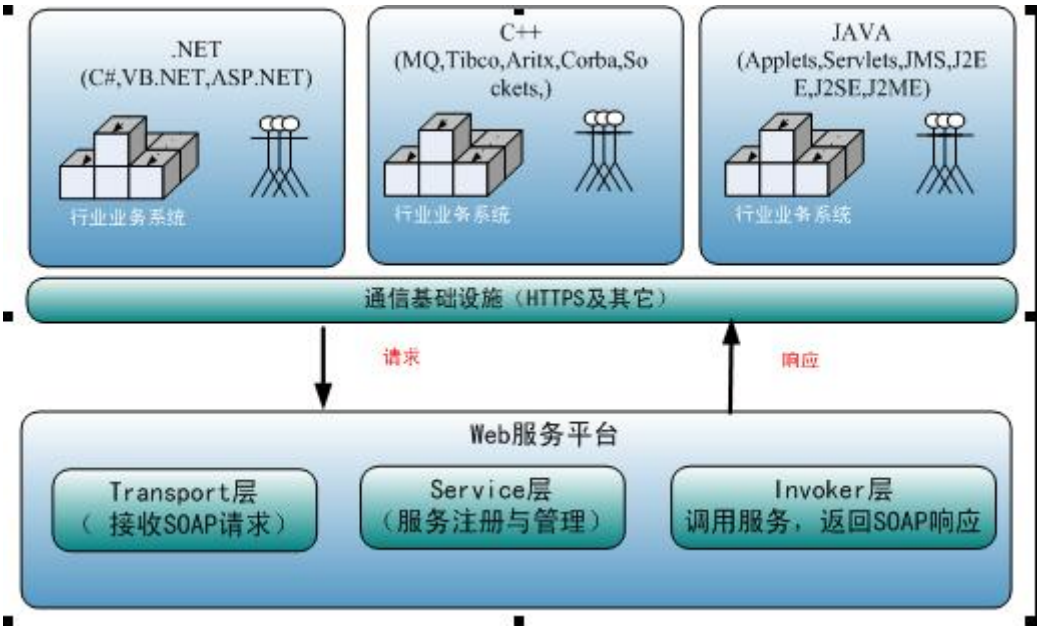
WSDL　　Web 服务描述语言（Web Service　Description　Language）

XFire　　Java SOAP 框架

第四章　　基本框架

4.1　概念模型

通过 Web 服务平台为各行业业务系统提供业务协作接口和数据支撑。如图所示，客户端向 Web 服务平台发送规范的数据访问请求（SOAP 请求），Web 服务平台接收数据访问请求后调用服务层完成请求处理，并将处理结果返回给客户端（SOAP 响应）。



4.2　应用场景

客户端根据服务请求的接口定义规范，开发服务请求应用程序访问所提供的服务。

说明：

1. 客户信息的验证通过授权文件参数来完成。授权文件在合同签署后会使用邮件等方式发送到客户手中（为加密格式）。客户在接口调用时将授权文件中读取的字符串作为服务调用时的一个接口参数传递给服务器端，服务端接收后通过授权文件内容进行身份验证。授权文件的内容包括：客户、用户、帐号、密码、IP 地址、具体调用的服务。
2. 采用 SOAP1.2 作为消息的封装格式进行服务请求，服务端接收到请求后进行验证，验证通过后以 SOAP1.2 作为消息的封装返回服务响应内容。

第五章　　接口定义

5.1　服务条件模板获取

接口名称：nciicGetCondition

传入参数：String inLicense

参数说明：

```
<?xml version="1.0" encoding=" UTF-8" ?>
<LICENSE>
  <USERINFO>
     <KHID>…</KHID>                          <!--客户 ID-->
     <YHID>…</YHID>                          <!--用户 ID-->
     <ZHID>…</ZHID>                          <!--帐号 ID-->
     <PASSWORD>…</ PASSWORD >                <!--登录口令-->
     <IP>…</IP>                              <!--登陆 IP-->
  </USERINFO>
  <SERVICEINFO>
     <FWID> …</FWID>                          <!--服务 ID-->
  </SERVICEINFO>
</LICENSE>
```

注意：

License 文件会在签署合同后通过邮件等其他形式得到，License 文件本身为加密格式，请参考示例代码编写客户端。如果授权文件中，参数：客户 ID、用户 ID、帐号 ID、密码（登录口令）、服务 ID 发生变化，则需要重新生成一个新的授权文件。

A）、正常返回结果：

```
<?xml version=\"1.0\" encoding=\"UTF-8\" ?>
- <ROWS>
<INFO>
    <SBM>******</SBM>
</INFO>
- <ROW>
  <GMSFHM>公民身份号码</GMSFHM>
  <XM>姓名</XM>
  </ROW>
- <ROW FSD="**" YWLX="**">
  <GMSFHM>XXXXXX</GMSFHM>
  <XM>XXXXXX</XM>
```

```
  </ROW>
- <ROW FSD="**" YWLX="**">
  <GMSFHM>XXXXXX</GMSFHM>
  <XM>XXXXXX</XM>
  </ROW>
  </ROWS>
```

注意：

1、通过授权文件验证，调用 nciicGetCondition 方法获取到的条件模版 xml 文件。填写数据时，依据此模版 xml 填写（详细说明请参看：**inConditions 参数说明**）。

B)、异常返回结果：

```
<?xml version="1.0" encoding="UTF-8" ?>
- <RESPONSE errorcode="xxx" code="0" countrows="1">
- <ROWS>
- <ROW>
    <ErrorCode>xxx</ErrorCode>                    <!--错误代码-->
    <ErrorMsg>xxxxxxxxx</ErrorMsg>                <!--错误描述-->
  </ROW>
  </ROWS>
</RESPONSE>
```

5.2 WS 号码姓名核查（支持单条和批量）

5.2.1. 功能描述

为获得 WEBSERVICE 服务许可证的客户提供人口信息快速身份核查服务。根据身份证号等信息核查系统内有无相匹配的人的基本信息，若存在则返回存在记录数，否则返回零条。分为单条核查和批量核查两种方式进行核查，单条核查只能核查一条记录的信息，批量核查则可以核查多条记录的信息，批量核查最多可核查 200 条记录。

5.2.2. 接口与参数

接口名称：nciicCheck

返回值： 1：正常返回；2：错误返回（包括返回空结果）

传入参数：String inLicense

String inConditions

参数说明：

字段	类型	描述
inLicense	String	客户身份校验文件（中心提供）
inConditions	String	条件 XML 字符串（由服务定义决定）

A）、inLicense 参数说明：

```
<?xml version="1.0" encoding=" UTF-8" ?>
<LICENSE>
  <USERINFO>
     <KHID>…</KHID>                          <!--客户 ID-->
     <YHID>…</YHID>                          <!--用户 ID-->
     <ZHID>…</ZHID>                          <!--帐号 ID-->
     <PASSWORD>…</ PASSWORD >                <!--登录口令-->
     <IP>…</IP>                              <!--登陆 IP-->
  </USERINFO>
  <SERVICEINFO>
     <FWID> …</FWID>                          <!--服务 ID-->
  </SERVICEINFO>
</LICENSE>
```

注意：

License 文件会在签署合同后通过邮件等其他形式得到，License 文件本身为加密格式，请参考示例代码编写客户端。如果授权文件中，参数：客户 ID、用户 ID、帐号 ID、密码（登录口令）、服务 ID 发生变化，则需要重新生成一个新的授权文件。

B）、inConditions 参数说明：

```
<?xml version=\"1.0\" encoding=\"UTF-8\" ?>
- <ROWS>
<INFO>
    <SBM>******</SBM>                  <!--用户唯一识别码-->
</INFO>
- <ROW>
  <GMSFHM>公民身份号码</GMSFHM>        <!--条件名称，不可修改-->
  <XM>姓名</XM>                        <!--条件名称，不可修改--->
  </ROW>
- <ROW FSD="**" YWLX="**">
<!-- 从第二个<ROW></ROW>开始填写数据；FSD 表示“业务发生地(6 位的行政区划编码)”；YWLX 表示“业务类型” -->
  <GMSFHM>********</GMSFHM>
  <XM>********</XM>
```

```
  </ROW>
- <ROW FSD="**" YWLX="**">
  <GMSFHM>********</GMSFHM>
  <XM>********</XM>
  </ROW>
  </ROWS>
```

注意：

1、用户唯一识别码：客户填写各自的业务帐号，集团用户可填写各自的小帐号，最大长度 40（20 个汉字）。

业务发生地：客户给自己的客户办理业务时的所在地，是 6 位的行政区划编码，最大长度 6（可以填写 3 个汉字）。例如：某某银行北京市朝阳区青年路支行，此支行所在地即为业务发生地。

业务类型：客户给自己的客户办理业务时，属于哪种业务类型，最大长度 40（可以填写 20 个汉字）。例如：某某银行北京市朝阳区青年路支行给市民办理开户、贷款、缴费等业务，此时办理的业务名称即为业务类型。

2、**inConditions** 条件 xml 串中，必须输入数据项：公民身份号码、姓名、用户唯一识别码、业务发生地、业务类型。

C)、正常返回结果：

```
  <?xml version="1.0" encoding="UTF-8" ?>
- <ROWS>
- <ROW no="1">                          <!-- no 属性为记录编号 --->
- <INPUT>                               <!--输入项节点-->
  <gmsfhm>61052119****</gmsfhm>         <!--公民身份号码-->
  <xm>王路</xm>                         <!--姓名-->
</INPUT>
- <OUTPUT>                              <!--输出项节点-->
- <ITEM>
  <gmsfhm />
  <result_gmsfhm>一致</result_gmsfhm>   <!--公民身份号码的核查结果-->
  </ITEM>
- <ITEM>
  <xm />
  <result_xm>一致</result_xm>           <!--姓名的核查结果-->
  </ITEM>
  </OUTPUT>
  </ROW>
- <ROW no="2">
- <INPUT>
```

```
  <gmsfhm>**********</gmsfhm>
  <xm>***</xm>
  </INPUT>
- <OUTPUT>
- <ITEM>
  <errormesage>服务结果:库中无此号，请到户籍所在地进行核实</errormesage>
                                        <!--库中无此号描述-->
  </ITEM>
- <ITEM>
  <errormesagecol />
  </ITEM>
  </OUTPUT>
  </ROW>
- <ROW no="3">
- <INPUT>
  <gmsfhm>61052119******</gmsfhm>
  <xm />
  </INPUT>
- <OUTPUT>
- <ITEM>
  <errormesage>必录项缺失</errormesage> <!--缺失必录项描述-->
  </ITEM>
- <ITEM>
  <errormesagecol>姓名</errormesagecol> <!--缺失必录项的字段名称-->
  </ITEM>
  </OUTPUT>
  </ROW>
- <ROW no="**">
- <INPUT>
  <gmsfhm>230602**********</gmsfhm>
  <xm>***</xm>
  </INPUT>
- <OUTPUT>
- <ITEM>
  <errormesage>业务发生地属性为空</errormesage>
  </ITEM>
- <ITEM>
  <errormesagecol />
  </ITEM>
  </OUTPUT>
  </ROW>
</ROWS>
```

说明：

公民身份号码与姓名是输入项，所以在结果节点中姓名所对应的核查结果节点存在一致、不一致描述；公民身份号码所对应的核查结果节点存在一致描述。

D)、异常返回结果：

```
<?xml version="1.0" encoding="UTF-8" ?>
- <RESPONSE errorcode="xxx" code="0" countrows="1">
- <ROWS>
- <ROW>
    <ErrorCode>xxx</ErrorCode>                    <!--错误代码-->
    <ErrorMsg>xxxxxxxxx</ErrorMsg>                <!--错误描述-->
  </ROW>
  </ROWS>
</RESPONSE>
```

说明：

1、 根节点<RESPONSE errorcode="**xxx**" code="**0**" countrows="**1**">中的“errorcode="**xxx**"”与“<ErrorCode>**xxx**</ErrorCode>”相同；“code="**0**" countrows="**1**""中的“code="**0**""暂时无用；“countrows="**1**""表示一行错误提示。

错误分类	错误代码	错误描述
数据库错误	-10	数据库连接失败
日志错误	-20	写入日志失败
预付费错误	-30	预付费扣费失败，请检查帐户余额
	-31	预付费用不足
XML 格式错误	-40	不是有效的 XML 文件
	-41	不是有效的 XML 文件格式
	-42	条件 XML 文件解析异常
服务相关错误	-50	服务不存在
	-51	服务不是接口方式
	-52	条件中节点与服务定义不一致
	-53	您没有权限使用此服务
	-54	提交的记录超过限制
	-55	构造服务条件失败

	-56	服务类别与调用的接口函数不符
服务条件错误	-60	服务条件项节点缺失
	-64	服务识别码信息不能为空（用户唯一识别码）
用户错误	-70	用户不存在
	-71	用户密码错误
	-72	IP 地址受限
授权文件错误	-80	授权文件格式错误
系统错误	-90	系统故障，请与管理员联系

第六章　　HTTPS 设置

HTTPS 是一种安全传输协议，已经广泛用于银行、证券、电信等对信息安全等级要求比较高的行业。

负载均衡设备目前提供了对 HTTPS 的支持，在 Webservice 中可以使用 HTTPS 实现安全数据传输，从而无需额外购置加密软件和设备。

基于上述原因，接口主要考虑采用 HTTPS 方式作为主要的安全解决实施方案。因此客户端需要按以下步骤进行配置(以 JAVA 为例)：

1、客户端在 IE 地址栏中输入：https://ipAddress:port, 其中，ipAddress 为人口库系统开通的客户接入服务器地址，port 为客户接入服务器的端口地址（默认端口 80）然后会弹出“安全警告”窗口。确定后，会弹出有关证书的提示窗口如下图：

选择“查看证书”后，在弹出窗口中，选择“详细信息”标签页，见下图：

选择“复制到文件”，

下一步，然后选择“base64 编码 X.509 (.CER) (S)”如下图：

再下一步，选择文件保存的路径和文件名。然后再“下一步”，然后“完成”。这样自定义的证书就完成了。

2、导入信任证书

假设你的证书别名是 tingting 证书文件名为 tingting.cer

并且你要导入到 cacerts 证书库(默认保护密码为 changeit)中，

命令：keytool -import -trustcacerts -alias tingting -file tingting.cer

-keystore cacerts -storepass changeit

其中-keystore C:\j2sdk1.5.0_17\jre\lib\security\cacerts 是绝对地址

-file：也是绝对地址；

以上是将自己创建的证书，倒入到客户端的 JRE 中。

第七章　　客户端调用说明示例

7.1　XFIRE 介绍

XFire 技术应用介绍:

Xfire 是 JAVA SOAP 框架，通过提供简单的 API 和支持标准协议，可以简洁快速开发面向服务的程序（将 Java 类方法转换成 Web 服务 ），内建在 STAX 基于流的 XML 解析引擎的基础上,Xfire 拥有很高的性能。Xfire 是新一代 WebService 框架，之所以称为新一代，是因为：

1、支持一系列 Web Service 的新标准--JSR181、WSDL2.0、SAOP1.2、JAXB2、WS-Security 等

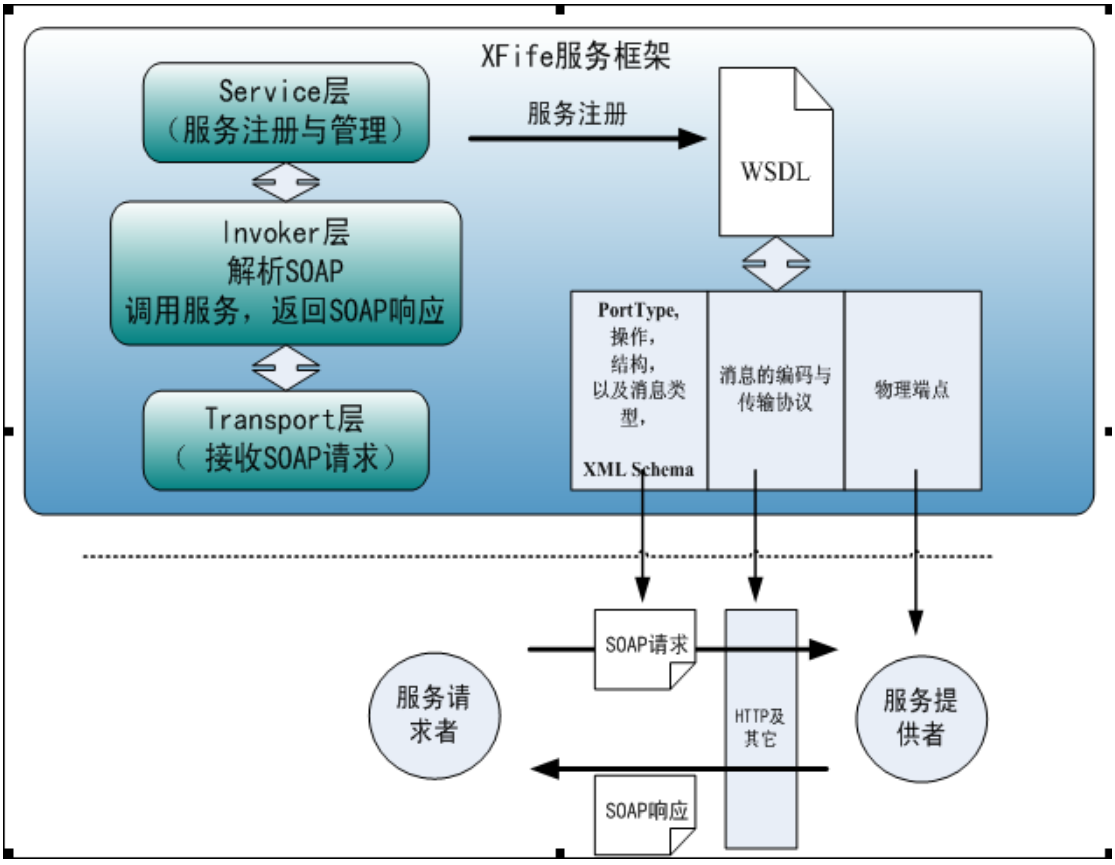
2、使用 Stax 解释 XML,性能有了质的提高。

3、与 Spring 的结合

4、灵活的 Binding 机制，包括默认 Aegis,xmlbeans,jaxb2,castor

5、客户端描述

基于 XFire 的服务框架：



客户端程序实现可以有多种方式，只要其开发语言支持对 webservice 的访问就没有问题。

这里以开发语言是 java、客户端 soap 框架是 xfire 为例。

7.2 客户端访问代码示例

客户端程序实现可以有多种方式，只要其开发语言支持对 WebService 的访问，访问主要是通过解析 WSDL 文件获得客户端调用程序的源代码，包括调用接口和各个输入输出的实体类，解析工具不同则获得的代码和调用接口类名会略有不同，但主要内容相同，这里只举例说明调用接口主要的相同部分的代码和一些客户端需要编写的类，为客户作为参考。

这里以开发语言是 java、客户端 soap 框架是 xfire 为例，其他客户端情况参考相应的技术文档，这里不一一列举。

1、 WebService 接口

Webservice 服务接口，该类可通过解析 WSDL 文件获得，类名与内容会略有不同，但其中包含的方法相同，以下例子中加入注释说明每个方法的功能，请客户参考。ServiceInf 接口类样例如下：

注：本类为服务接口类，其中的各种接口方法与 WebService 服务器提供的对外服务方法名称及功能相同。

```
public interface ServiceInf{
    // 核查方法
    public String nciicCheck(String inLicense, String inConditions);
    // 取得条件文件模板
    public String nciicGetCondition(String inLicense) throws
Exception;

}
```

2、调用类

该类不是通过解析 WSDL 文档得到的类，是示范怎样得到本地代理调用对外服务的类，请客户参考。

调用方法样例如下：

```
package com.nciic.webservice.client;

import java.io.BufferedReader;
import java.io.FileNotFoundException;
import java.io.FileReader;
import java.io.IOException;
import java.lang.reflect.Proxy;
import java.net.MalformedURLException;

import org.apache.commons.httpclient.protocol.Protocol;
import
org.apache.commons.httpclient.protocol.ProtocolSocketFactory;
import org.codehaus.xfire.client.Client;
import org.codehaus.xfire.client.XFireProxy;
import org.codehaus.xfire.client.XFireProxyFactory;
import org.codehaus.xfire.service.Service;
import org.codehaus.xfire.service.binding.ObjectServiceFactory;
import
org.codehaus.xfire.transport.http.EasySSLProtocolSocketFactory;
import org.codehaus.xfire.util.dom.DOMOutHandler;

import com.nciic.serv.webservice.ServiceInf;

public class NciicClient {
    public   static   final   String   SERVICE_URL   =   "https://
api.nciic.com.cn/nciic_ws/services/";
public NciicClient() {
}
```

```
/**
     * @param args
     * @throws MalformedURLException
     */
    public static void main(String[] args) throws
MalformedURLException {
      /**
       * 授权文件名称，如果该文件不在客户端工程根目录下，请
       * 将文件路径添加。如 C:\\1.txt
       */
       String license = "授权文件";
       String con="<?xml version=\"1.0\" encoding=\"UTF-8\" ?>
<ROWS><INFO><SBM>******</SBM></INFO><ROW><GMSFHM>公民身份号码
</GMSFHM><XM>姓名</XM></ROW><ROW FSD=" **" YWLX=" **" >
<GMSFHM>XXXXXX</GMSFHM><XM>XXXXXX</XM></ROW><ROW FSD=" **" YWLX="
**"><GMSFHM>XXXXXX</GMSFHM><XM>XXXXXX</XM></ROW></ROWS>";
       new NciicClient().executeClient("NciicServices", license ,
con);
    }

    public String executeClient(String serviceName, String license,
String condition)
          throws MalformedURLException {

       ProtocolSocketFactory easy = new
EasySSLProtocolSocketFactory();
       Protocol protocol = new Protocol("https", easy, 443);
       Protocol.registerProtocol("https", protocol);

       Service serviceModel = new ObjectServiceFactory().create(
             ServiceInf.class, "NciicServices", null, null);

       ServiceInf service = (ServiceInf) new
XFireProxyFactory().create(
             serviceModel, SERVICE_URL + serviceName);

       Client client = ((XFireProxy)
Proxy.getInvocationHandler(service))
             .getClient();

       client.addOutHandler(new DOMOutHandler());
```

```
        //压缩传输
        client.setProperty(CommonsHttpMessageSender.GZIP_ENABLED,
Boolean.TRUE);
        //忽略超时
        client.setProperty(CommonsHttpMessageSender.DISABLE_EXPEC
    T_CONTINUE, "1");
        client.setProperty(CommonsHttpMessageSender.HTTP_TIMEOUT,
"0");
        /**
        * 读取授权文件内容，因为授权文件为加密格式请不要对内容做任何修改
        */

String licensecode = null;
String result = null;
        BufferedReader in;
        try {
            in = new BufferedReader(new FileReader(license+".txt"));
            licensecode = in.readLine();
            //调用核查方法
             result = service.nciicCheck(licensecode, condition);
            //服务条件模板获取
             //result = service.nciicGetCondition(licensecode);
        } catch (FileNotFoundException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        } catch (IOException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }
        System.out.println(result);
        return result;
    }
}
```

7.3 常见问题描述

1、发生冲突如何解决

由于 xFire 官方网站上面 xFire1.2.6 版本提供的 lib 包中部分资源非最新版本（如：spring-1.2.6.jar）。故在进行 xFire 资源引入时，需根据客户系统实际情况来确定是否引入 xFire1.2.6 提供的资源包。在测试过程中发现官方网站上可能引起冲突的 jar 包：spring-1.2.6.jar、stax-api-1.0.1.jar(提示 javax.xml.namespace.QName 方法错误)、jdom-1.0.jar。开发时建议使用本规范提供的样例工程代码。

2、xFire 版本问题

如客户端使用 Java 语言实现，建议使用 xFire 实现，xFire 版本为 1.2.6。

第八章　　其他

1．在 IE 地址栏中输入 WSDL 地址：

https://api.nciic.com.cn/nciic_ws/services/NciicServices?wsdl，可以看到我单位对外发布的接口函数和参数。

2．客户端使用的 jdk 版本：jdk1.42 或者高于这个版本