卫宁健康单点登录接口规范

目录

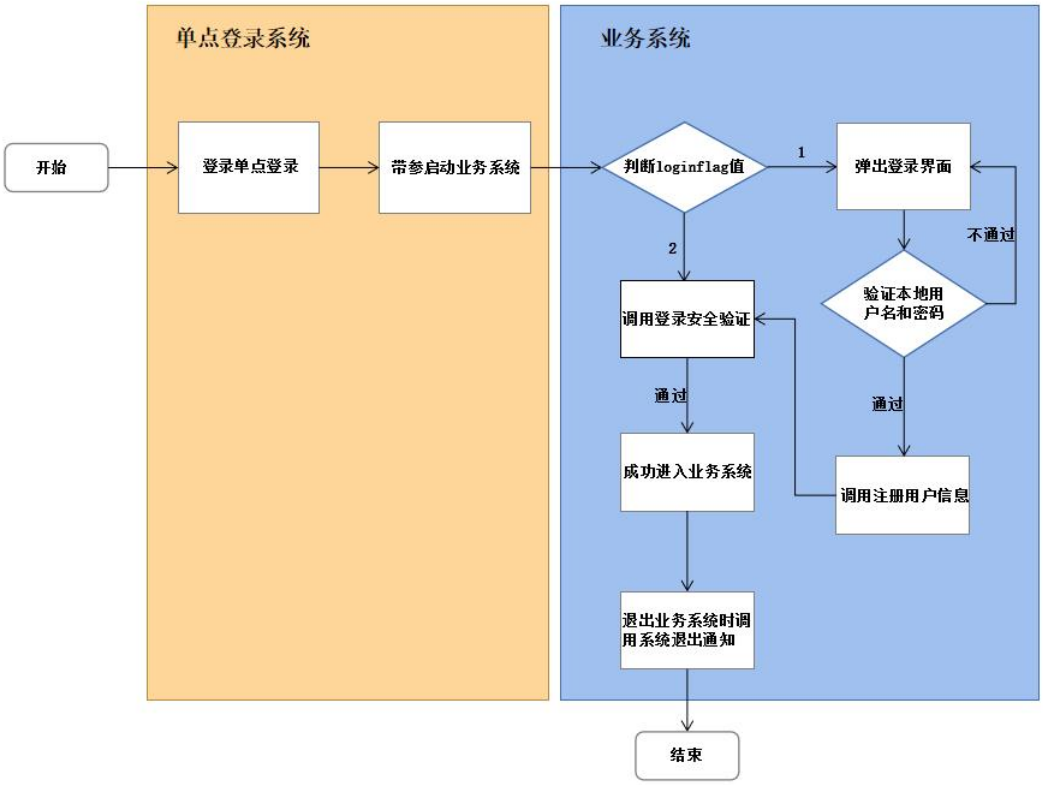
第一章 单点登录 3
一、概述 3
二、业务流程说明 3
三、参数传入方式 4
第二章 服务接口说明 5
一、注册服务（LoginInfoRegister） 5
二、验证服务（LoginVerify） 6
第三章 示例 7
一、 Soapui 7
二、 .net 8

第一章　单点登录

一、概述

用户只需登录一次，即可通过单点登录系统访问医院的多个应用系统，二次登陆时无需重新输入用户名和密码。

二、业务流程说明



业务系统信息注册：业务系统在单点登录系统中添加业务系统信息，包括业务系统编码、业务系统名称、业务系统程序结构（CS/BS）、业务系统程序路径或 url 地址等。

登录单点登录系统：用户首先启动单点登录系统，并输入平台用户名和密码进入单点登录系统。平台用户名及密码的维护可在平台管理界面中进行维护。

带参启动业务系统：单点登录系统启动业务系统的程序，并传入相应的参数。业务系统可以是 C/S 或者 B/S 结构的程序，分别采用启动 EXE 带参和调用 URL 地址的方式。具体的参数如下：

序号	参数符号	参数值	备注
1	ptflag	PTSSO	固定值
2	appid	业务系统的 ID	业务系统在单点登录系统中注册时分配的程序 ID,由单点登录系统管理员提供
3	userid	单点登录系统中用户 ID	
4	loginid	业务系统中用户登录 ID	首次采用单点登录方式登录时默认值为“-”。
5	captcha	验证码	
6	loginflag	登录标志	首次登录时为 1,非首次登录为 2
7	extendparam	可扩展参数	XML 格式，key、value 的形式实现,空的参数会传入‘-’符号。暂空

判断 loginflag 值：业务系统读取参数中的 loginflag 值，根据不同的值进行不同的业务流转。

验证本地用户名和密码:业务系统根据 loginflag 判断是否弹出登录界面，当 loginflag 为 1 时弹出登录界面。用户输入业务系统本地的用户名和密码，业务系统按照本地的验证逻辑验证成功后进入下面的步骤。不成功则再次弹出登录界面。

调用注册用户信息：首次登录时，业务系统验证本地系统的用户名和密码成功后，将登录的用户 ID 和用户名称通过服务注册到平台。如果业务系统采用的用户信息和单点登录系统是同一套则此过程可以省略。详见注册用户信息服务。

调用登录安全验证：业务系统被启动之后需调用认证服务的验证服务，以确保此次启动是合法的。详见登录安全验证服务。

三、参数传入方式

（1）BS 程序调用示例(URl 后面加七个参数，参数中间以&符号隔开)：

http://ip:port/+url?ptflag=PTSSO&appid=EMPI_test&userid=OO&loginid=-&captcha=08068671-a60b-4aef-8e59-1ae9868b1284&loginflag=1&extendparam=-

（2）CS 程序调用示例(exe 启动文件后面带七个参数，参数中间以空格分开)：

D/:xxxx.exe PTSSO EMPI_test 00 - 08068671-a60b-4aef-8e59-1ae9868b1284 1 -

第二章 服务接口说明

采用 WEBSERVICE 的方式提供服务。

注：使用 SoupUI 测试，删除 <!--Optional:-->，XML 参数前需要包装：<![CDATA[ ]]>

一、注册服务（LoginInfoRegister）

1、 业务系统用户注册。

函数	LoginInfoRegister(string inputdata)		
应用场景	首次使用单点登录系统或者统一认证的方式登录业务系统时调用。目的是映射业务系统用户与平台用户		
输入参数说明	appid	varchar(30)	业务系统注册后被分配的 ID
	userid	varchar(20)	平台中统一用户的 ID
	loginid	varchar(20)	业务系统中登录的 ID。
	loginname	Varchar(40)	业务系统中的用户名
	password	varchar(40)	暂不用，传空
格式	<?xml version="1.0" encoding="GB2312" standalone="yes"?> <data> <appid>系统 ID</appid> <userid>平台用户 ID</userid> <loginid>业务系统的用户 ID</loginid> <loginname>业务系统的用户名称</loginname> <password>密码</password>		

	</data>		
返回值参数	Retcode	String	状态码,AA:成功 AE:失败
	Msg	string	失败原因
格式	<?xml version="1.0" encoding="GB2312" standalone="yes"?> <output> <retcode>状态码</retcode> <msg>失败原因</msg> </ output>		

二、验证服务（LoginVerify）

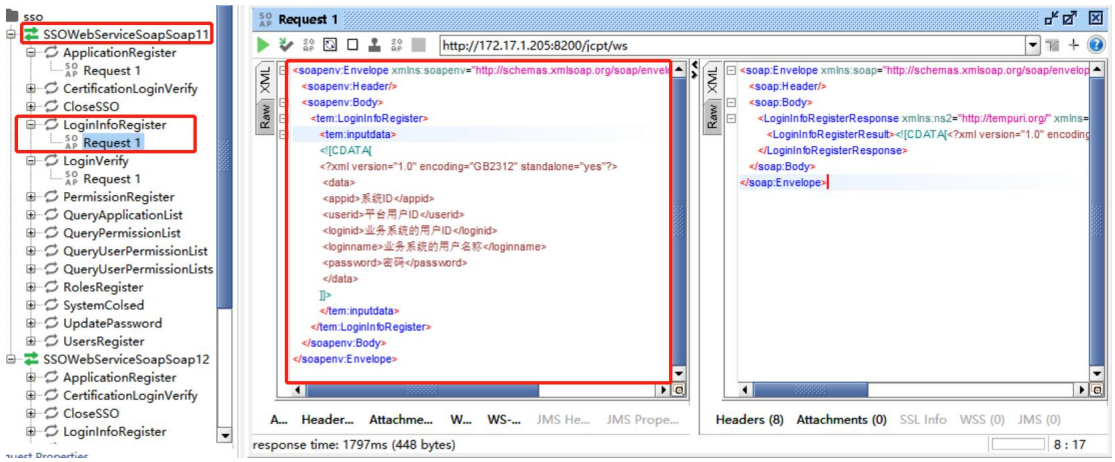
1、 登录安全验证。

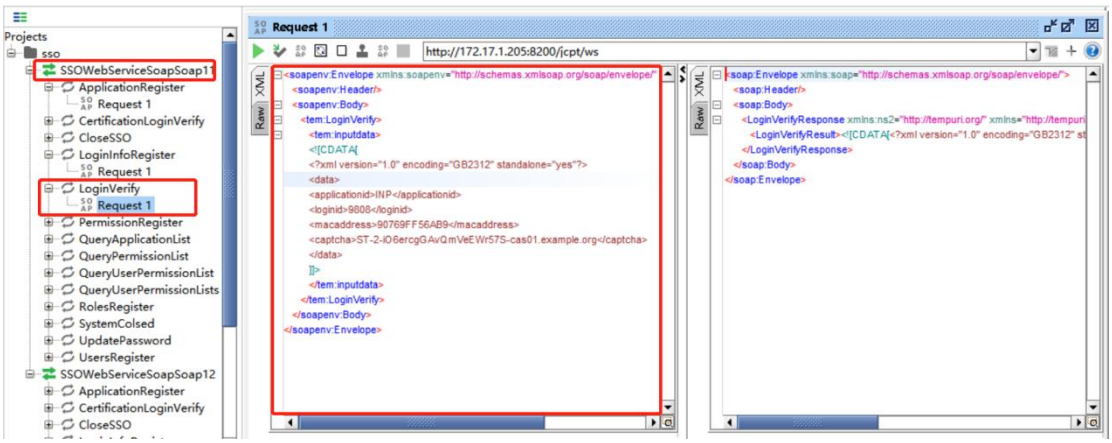
函数	LoginVerify(string inputdata)		
应用场景	从单点登录窗口登录业务系统时,会以参数形式传递给业务系统一个验证码，然后再调用此函数验证是否是安全登录，返回值为 AA 时应成功登录，否则应拒绝登录。		
传入参数说明	applicationid	varchar(20)	当前登录的业务系统 ID
	loginid	varchar(10)	当前登录的业务系统用户 ID
	macaddress	Varchar(40)	当前做登录操作的电脑 mac 地址
	captcha	Varchar(40)	登录时平台传入的验证码
格式	<?xml version="1.0" encoding="GB2312" standalone="yes"?> <data> <applicationid>系统 ID</applicationid> <loginid>当前登录的业务系统用户 ID</loginid> <macaddress>当前做登录操作的电脑 mac 地址</macaddress>		

	<captcha>登录时平台传入的验证码</captcha> </data>		
返回值参数	retcode	String	状态码,AA:成功 AE:失败
	msg	string	失败原因
格式	<?xml version="1.0" encoding="GB2312" standalone="yes"?> <output> <retcode>状态码</retcode> <msg>失败原因</msg> 〈/output〉		

第三章 示例

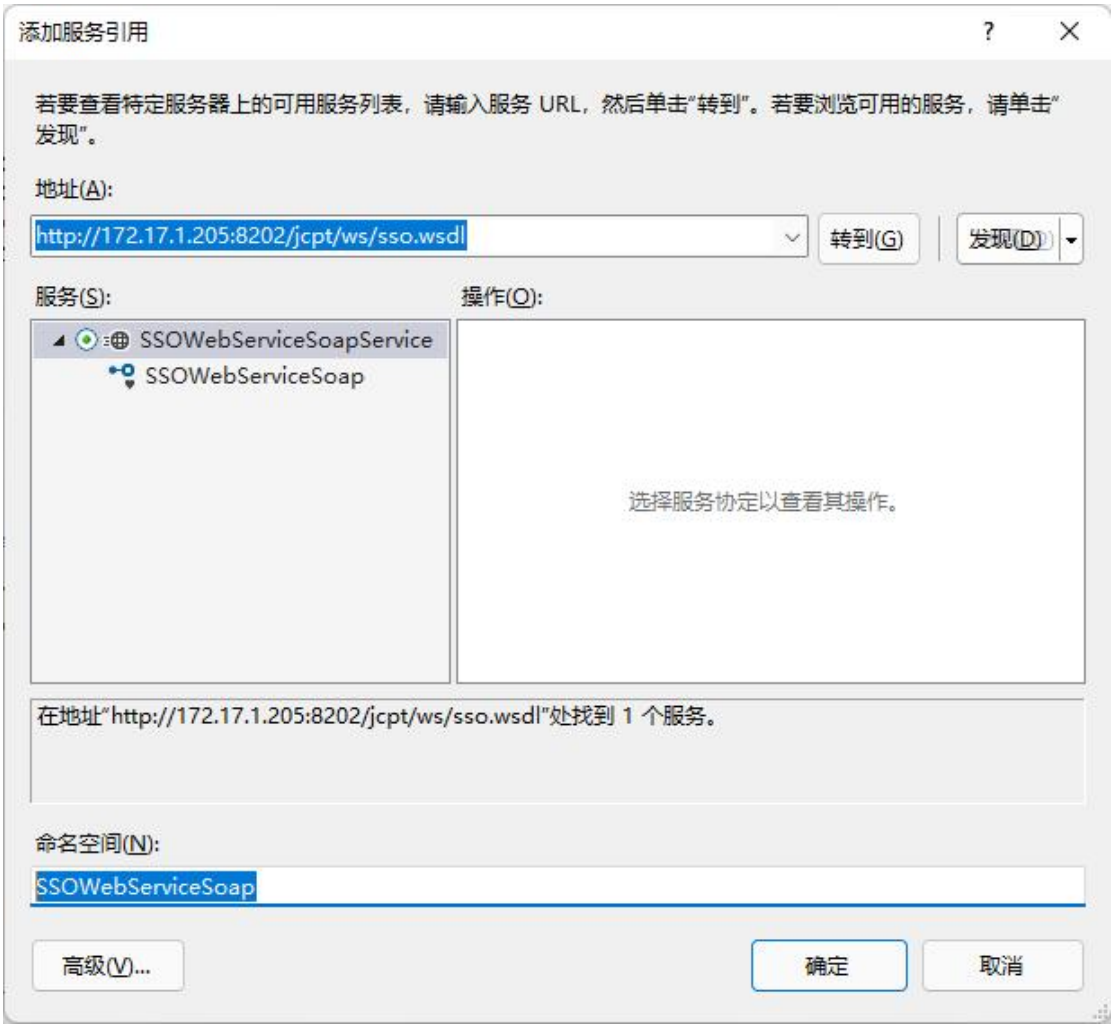
一、Soapui





二、.net

1. 添加服务引用（或者名为 web 引用）



2. 代码调用

```
SSOWebServiceSoap.SSOWebServiceSoapClient ds = new SSOWebServiceSoap.SSOWebServiceSoapClient("SSOWebServiceSoapSoap11", "http://172.17.1.205:8202/jcpt/ws/sso.wsdl");
```

```
SSOWebServiceSoap.LoginVerify invalue = new SSOWebServiceSoap.LoginVerify();
invalue.inputdata =
"<data><applicationid>EMPI</applicationid><loginid>00</loginid><macaddress></macaddress
><captcha>ST-16-pQ1wIcU62nOdPdrSsLNm-cas01.example.org</captcha></data>";
SSOWebServiceSoap.LoginVerifyResponse res = ds.LoginVerify(invalue);
string result = res.LoginVerifyResult;
```

三、Java

```
import org.apache.cxf.endpoint.Client;
import org.apache.cxf.jaxws.endpoint.dynamic.JaxWsDynamicClientFactory;

public class test {

    /**
     *  1. 导入 pom
     *          <dependency>
     *              <groupId>org.apache.cxf</groupId>
     *              <artifactId>cxf-rt-frontend-jaxws</artifactId>
     *              <version>3.5.3</version>
     *          </dependency>
     *          <dependency>
     *              <groupId>org.apache.cxf</groupId>
     *              <artifactId>cxf-rt-transports-http</artifactId>
     *              <version>3.5.3</version>
     *          </dependency>
     *  2. 修改 url
     * @param args
     */
    public static void main(String[] args) {
        String url = "http://172.17.1.205:8200/jcpt/ws/sso.wsdl";
        // 创建动态客户端
        JaxWsDynamicClientFactory dcf = JaxWsDynamicClientFactory.newInstance();
        //对方的 wsdl 地址
        Client client = dcf.createClient(url);
        try {
            String xml = "<?xml version=\"1.0\" encoding=\"GB2312\" standalone=\"yes\"?>"
+
                    "<data>" +
                    "<applicationid>INP</applicationid>" +
                    "<loginid>9808</loginid>" +
                    "<macaddress>90769FF56AB9</macaddress>" +
"<captcha>ST-2-iO6ercgGAvQmVeEWr57S-cas01.example.org</captcha>" +
                    "</data>";
```

```
                Object[] resp = client.invoke("LoginVerify",xml);
                String respXml = (String)resp[0];
System.out.println("============================respXml============================");
                System.out.println(respXml);
System.out.println("============================respXml============================");
            } catch (Exception e) {
                e.printStackTrace();
            }
        }

}
```