北京博思致新互联网科技有限责任公司

政付通行业缴费平台

接口规范说明书

版本信息：V**2.4.1**

编写日期：2020-07-06

文档修订记录

序号	版本号	修订日期	修订人	备注
1	V2.0.0	20200706	廖伟	第一版
2	V2.0.1	20200728	廖伟	修改部分描述
3	V2.0.2	20200831	廖伟	代缴费数据推送接口，新增额外缴款人（单位）字段
4	V2.0.3	20200908	廖伟	待缴费数据推送接口新增是否申请虚拟子账号字段
5	V2.0.4	20201208	廖伟	获取支付 url 接口更新
6	V2.0.5	20211210	廖伟	推送数据接口新增是否申请缴款码字段
7	V2.1.0	20220704	廖伟	新增多条件查询缴费状态接口，更新查询某天已缴费数据接口
8	V2.2.1	20230209	李安鹏	新增国密签名以及加密解密相关描述
9	V2.2.2	20230905	廖伟	退费接口新增收费项目
10	V2.2.3	20230915	廖伟	查询某天已缴费数据接口新增备注字段
11	V2.3.0	20230918	廖伟	新增分页查询某天已缴费接口
12	V2.4.0	20231016	邹鸣浩	待缴费数据推送接口新增向业务系统回调电子票据信息字段，新增回调票据信息通知接口
13	V2.4.1	20231109	廖伟	待缴费数据推送接口新增处罚决定书号

1 接口规范说明

1.1 前言

文档中的【业务系统】指接口调用方，【缴费平台】指服务提供方。缴费平台提供 JAVA 语言 SDK,其他语言请自行参考对应规则开发。

1.2 访问协议

统一技术标准，采用 HTTP 协议与服务进行交互，请求方式使用 POST。

传输方式	支持 HTTP/HTTPS 传输
提交方式	POST
数据传输格式	标准 JSON 格式
字符编码	统一采用 UTF-8 字符编码
签名算法	RSA2、SM2
加密算法	AES/CBC/PKCS5Padding、SM4/CBC/PKCS5Padding
加密要求	请求和接收数据均需要加密，详细方法请参考《加密与签名规则》
签名要求	请求和接收数据均需要校验签名，详细方法请参考《加密与签名规则》
API 服务地址	API 服务地址规则： https://域名:端口/api/v2/standard

1.3 访问授权

业务系统调用接口服务之前，必须获取接口缴费平台的访问授权，即 appid、业务系统私钥（签名用）、缴费平台公钥（验签用）和加密密钥。相关参数获取联系缴费平台。

1.4 公共参数规范

所有请求、返回参数一律小写，多个单词之间使用下划线连接，参数对应的值不做限制。

1.4.1 公共请求参数

其中参数类型后面的数字表示该参数最大长度,如果没有则表示该参数最大长度不限。

参数	类型	是否必填	描述	示例值
app_id	String(32)	是	缴费平台分配给业务系统的应用 ID	fcaa6dcb12e148dd9929bf787fd8ef84
method	String(128)	是	接口名称	bus.unpay.data.sync
sign	String(350)	是	业务系统请求参数的签名串，详见 1.5.3	详见示例
timestamp	String(19)	是	请求发送时的时间，格式"yyyy-MM-dd HH:mm:ss"	2019-07-18 10:44:33
version	String(3)	是	接口版本，默认为：1.0	1.0
data	String	是	接口请求参数集合，最大长度不限，除公共参数外所有请求参数都必须放在这个参数中传递。使用 AES 加密后传递	
sign_type	String	是	签名方式	RSA2/SM2
encrypt_type	String	是	加密方式	AES/SM4

1、 请求方参数主要包括 app_id、method 、sign、timestamp、version、data、sign_type、encrypt_type。

2、 参数 data 为接口具体参数且采用 AES/CBC/PKCS5Padding、SM4/CBC/PKCS5Padding 加密后的内容。

注：加密、签名所用的 key 的值必须与 app_id 对应

1.4.2 公共响应参数

其中参数后面的数字表示该参数最大长度,如果没有则表示该参数最大长度不限。

参数	类型	是否必填	描述	示例值
response	String	是	详见 response 参数	
sign	String(350)	是	验签详见 1.5.5	

response 解密后明细：

参数	类型	是否必填	描述	示例值
code	String	是	网关返回码，详见 1.4.2.1	10000
msg	String	是	网关返回码描述，详见 1.4.2.1	success
bus_code	String	否	业务返回码，出错的时候返回	
bus_msg	String	否	业务返回码描述，出错的时候返回	

1、接口返回内容主要包括 code、msg、签名 sign 等。

2、业务系统在接收到响应参数后先验签，验签成功后再对 response 参数的值进行解密，解密后进行业务处理。

1.4.2.1 返回结果参数中网关、业务返回码和对应描述说明

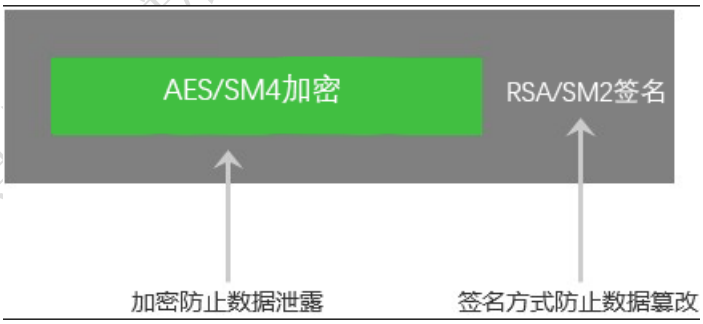
code	Msg	bus_code	bus_msg	解决方案
10000	接口调用成功，调用结果请参考具体 API 接口所对应的返回参数			
20000	服务不可用	20001	接口不可用	稍后重试或联系服务提供方
		20002	网关发生未知错误	稍后重试或联系服务提供方

		20003	网关未接受到请求数据	检查请求参数
		20004	公共请求参数不是标准 JSON 格式	检查请求参数
30000	授权权限不足	30001	app_id 无效	检查入参 app_id，app_id 不存在或者未上线
		30002	app_id 未上线	
		30003	app_id 未授权当前接口	检查接口是否错误或联系服务提供方
40000	缺少必传参数	40001	缺少 appId 参数	检查请求参数
		40002	缺少 method 参数	检查请求参数
		40003	缺少 sign 参数	检查请求参数
		40004	缺少 timestamp 参数	检查请求参数
		40005	缺少 version 参数	检查请求参数
		40006	缺少 data 参数	检查请求参数
		40007	缺少 signType 参数	检查请求参数
		40008	缺少 encryptType 参数	检查请求参数
50000	非法参数	50001	参数无效	检查参数，格式不对、非法值、越界等
		50002	不存在的接口名	检查入参 method 是否正确

		50003	服务校验签名不匹配	签名算法是否无误，是否按签名规则生成
		50004	非法的时间戳参数	时间戳参数 timestamp 非法，请检查格式是否为"yyyy-MM-dd HH:mm:ss"
		50005	非法接口版本	检查入参 version 是否正确
60000	验签成功后业务接口参数以及业务请求返回错误，详见具体描述。			

1.5 加密与签名规则

为了防止传递的信息被第三方抓包，泄露信息或对传递的数据进行信息篡改，需要对传输的信息加密和签名，加密和签名方式可以选择 RSA 和国密



◆ 使用 AES/SM4 算法对接口请求和响应内容进行加密，密文无法被第三方识别，从而防止接口传输数据泄露。

◆ 使用 RSA/SM2 算法对接口请求和响应内容进行签名，业务系统和缴费平台别加签验签，以确认接口传输的内容没有被篡改。

◆ 业务系统请求时应对请求参数中 data 参数的值先做加密，然后对整个公共请求参数按规则进行签名。选择 AES 加密使用 RSA 进行签名，选择 SM4 加密使用 SM2 签名

◆ 业务系统收到响应时应对响应参数中 response 参数的值，如果选择国密的方式进行处理，先做 SM2 验签，验签通过后对 response 参数的值进行 SM4 解密，否则先做 RSA 验签，验签通过后对 response 参数的值进行 AES 解密。

1.5.1 加解密

1.5.1.1 对公共请求参数 data 的值加密

以下是加密 java 示例处理逻辑（注：若您使用 PHP，可能需要自行安装用于加密的 php 扩展）。

- 密钥：缴费平台提供
- 原文：公共请求参数 data 的值
- 字符集：UTF-8
- 加密算法：AES/CBC/PKCS5Padding 或者 SM4

其它语言基本的加密逻辑一样，需要注意的是对加密后得到的字节数组先做 base64 编码，然后再新建字符串（由于 base64 后的字节一定是 ASCII 范围内，所以最后一步 new String 的时候无需指定字符集）。

AES 加密：

```
1.  public static String encrypt() throws Exception {
2.        String key = "缴费平台提供的 AES 秘钥";
3.        String content = "需要加密的参数";
4.        String charset = "UTF-8";
5.        String fullAlg = "AES/CBC/PKCS5Padding";
6.        Cipher cipher = Cipher.getInstance(fullAlg);
7.        IvParameterSpec iv = new IvParameterSpec(initIv(fullAlg));
8.        cipher.init(Cipher.ENCRYPT_MODE,
9.                new SecretKeySpec(Base64.decodeBase64(key.getBytes()), "AES"),
    iv);
10.       byte[] encryptBytes = cipher.doFinal(content.getBytes(charset));
11.       return new String(Base64.encodeBase64(encryptBytes));
12.   }
13.
14.   /**
```

```
15.     * 初始向量的方法, 全部为0. 这里的写法适合于其它算法,针对 AES 算法的话,IV 值一定是 128 位的(16 字节).
16.     *
17.     * @param fullAlg
18.     * @return
19.     * @throws GeneralSecurityException
20.     */
21. private static byte[] initIv(String fullAlg) {
22.     try {
23.         Cipher cipher = Cipher.getInstance(fullAlg);
24.         int blockSize = cipher.getBlockSize();
25.         byte[] iv = new byte[blockSize];
26.         for (int i = 0; i < blockSize; ++i) {
27.             iv[i] = 0;
28.         }
29.         return iv;
30.     } catch (Exception e) {
31.         int blockSize = 16;
32.         byte[] iv = new byte[blockSize];
33.         for (int i = 0; i < blockSize; ++i) {
34.             iv[i] = 0;
35.         }
36.         return iv;
37.     }
38. }
```

SM4 加密：

```
1.  /**
2.       * SM4 加密
3.       *
4.       * @param content 待加密数据
5.       * @param sm4Key  密钥
6.       * @return
7.       */
8.      public static String encrypt(String content, String sm4Key) throws CapitalException {
9.          try {
10.             Cipher cipher = Cipher.getInstance("SM4/CBC/PKCS5Padding", BouncyCastleProvider.PROVIDER_NAME);
11.             // 初始化 iv 偏移量
12.             byte[] iv = new byte[16];
13.             SecureRandom secureRandom = new SecureRandom();
14.             secureRandom.nextBytes(iv);
```

```
            IvParameterSpec ivParameterSpec = new IvParameterSpec(iv);

            cipher.init(Cipher.ENCRYPT_MODE, new SecretKeySpec(ByteUtils.fro
mHexString(sm4Key), "SM4"),
                    ivParameterSpec);
            byte[] bytes = cipher.doFinal(content.getBytes(CapitalConstants.
DEFAULT_CHARSET_UTF8));
            byte[] encryptedBytes = new byte[IV_SIZE + bytes.length];
            // 数组复制替换
            System.arraycopy(iv, 0, encryptedBytes, 0, 16);
            System.arraycopy(bytes, 0, encryptedBytes, 16, bytes.length);
            return ByteUtils.toHexString(encryptedBytes);
        } catch (Exception e) {
            throw new CapitalException(e);
        }
    }
```

1.5.1.2 对公共响应参数 response 的值解密

针对缴费平台返回的报文，业务系统需要先验签，再解密。

- 密钥：缴费平台提供
- 原文：公共响应参数 response 的值
- 字符集：UTF-8
- 解密算法：AES/CBC/PKCS5Padding 或者 SM4

AES 解密：

```
/**
 * AES 解密
 * @return
 * @throws IppApiException
 */
private static String aesDecrypt() throws IppApiException {
        String key = "缴费平台提供的 AES 秘钥";
        String content = "需要加密的参数";
        String charset = "UTF-8";
        String fullAlg = "AES/CBC/PKCS5Padding";
        Cipher cipher = Cipher.getInstance(fullAlg);

    try {
        //反序列化 AES 密钥
        SecretKeySpec secretKeySpec = new SecretKeySpec(Base64.decodeBase64(k
ey.getBytes()), "AES");
```

```
16.         //128bit 全零的 IV 向量
17.         IvParameterSpec iv = new IvParameterSpec(initIv(fullAlg ));
18.         Cipher cipher = Cipher.getInstance(fullAlg );
19.         //初始化加密器并解密
20.         cipher.init(Cipher.DECRYPT_MODE, secretKeySpec, iv);
21.         byte[] cleanBytes = cipher.doFinal(Base64.decodeBase64(content.getByt
    es()));
22.         return new String(cleanBytes, charset);
23.     } catch (Exception e) {
24.         throw new IppApiException("AES 解密失败：
    Aescontent = " + content + "; charset = " + charset, e);
25.     }
26. }
```

SM4 解密：

```
1.  /**
2.       * SM4 解密
3.       *
4.       * @param content 待解密数据
5.       * @param sm4Key  密钥
6.       * @return
7.       * @throws CapitalException
8.       */
9.     public static String decrypt(String content, String sm4Key) throws Capi
    talException {
10.         try {
11.             byte[] encryptedBytes = ByteUtils.fromHexString(content);
12.
13.             Cipher cipher = Cipher.getInstance(TRANSFORMATION_NAME, BouncyC
    astleProvider.PROVIDER_NAME);
14.
15.             IvParameterSpec ivParameterSpec = new IvParameterSpec(encrypted
    Bytes, 0, IV_SIZE);
16.
17.             cipher.init(Cipher.DECRYPT_MODE, new SecretKeySpec(ByteUtils.fr
    omHexString(sm4Key), ALGORITHM),
18.                     ivParameterSpec);
19.
20.             byte[] contentBytes = cipher.doFinal(encryptedBytes, IV_SIZE, e
    ncryptedBytes.length - IV_SIZE);
21.             return new String(contentBytes, CapitalConstants.DEFAULT_CHARSE
    T_UTF8);
22.         } catch (Exception e) {
```

```
            throw new CapitalException(e);
        }
    }
```

1.5.2 接口签名和验签

1.5.2.1 接口请求数据签名

以 JAVA 语言为例：业务系统首先将参数和值放入一个对象或 map 中，使用 JSONObject 把这个对象或 map 转化成 json 对象。然后构建签名原文，在构建签名原文时，需要把参数按 ASCII 码递增排序（具体排序方法直接调用 JAVA 的 Arrays.sort 方法），最后使用业务系统私钥的私钥对签名原文进行签名。

特别注意以下重要规则：

◆筛选并排序

获取所有请求参数（data 的值先加密），剔除 sign 字段，剔除值为空的参数，并按照第一个字符的键值 ASCII 码递增排序（字母升序排序），如果遇到相同字符则按照第二个字符的键值 ASCII 码递增排序，以此类推。

◆拼接

将排序后的参数与其对应值，组合成“key=value”的格式，并且把这些参数用&字符连接起来，此时生成的字符串为待签名字符串。

◆调用签名函数

使用各自语言对应的 RSA2（即 SHA256WithRSA)签名函数和缴费平台分配给业务系统的业务系统私钥对待签名字符串进行签名，得到签名字符串。

◆把生成的签名字符串赋值给 sign 参数，拼接到请求参数中。

RSA 签名：

◆代码实例

```
/**
     * sha256WithRsa 加签
     *
     * @param content    待签名字符串
     * @param privateKey 私钥
     * @param charset    字符编码
     * @return
     *
     * @throws IppApiException
     */
```

```
    private static String rsa256Sign(String content, String privateKey, String charset) throws IppApiException {

        try {
            PrivateKey priKey = getPrivateKeyFromPkcs8(IppConstants.SIGN_TYPE_RSA,
                    new ByteArrayInputStream(privateKey.getBytes()));
            Signature signature = Signature.getInstance(IppConstants.SIGN_SHA256RSA_ALGORITHMS);
            signature.initSign(priKey);
            if (BaseStringUtils.isEmpty(charset)) {
                signature.update(content.getBytes());
            } else {
                signature.update(content.getBytes(charset));
            }
            byte[] signed = signature.sign();
            return new String(Base64.encodeBase64(signed));
        } catch (Exception e) {
            throw new IppApiException("RSAcontent = " + content + "; charset = " + charset, e);
        }

    }
```

◆请求示例

例如下面的请求示例，参数值都是示例值，业务系统参考格式即可：

```
app_id=   33bfc65fe8e843eaaad0bb6a0eee9a41
method= bank.query.pay.details
version=1.0
    timestamp  =  2022-11-23 11:35:00
data=5tOZLBQUFAtiVrfAC9mHKMOgZUiR848C6Js2SVyBUJ2KmOlP2iOYBwu49V3AxJKdLogPJnsBumQxMOjuUbFJ+A==
signType = RSA2
encryptType = AES
```

按照签名规则处理后最终待签名字符串为：

app_id=33bfc65fe8e843eaaad0bb6a0eee9a41&data=5tOZLBQUFAtiVrfAC9mHKMOgZUiR848C6Js2SVyBUJ2KmOlP2iOYBwu49V3AxJKdLogPJnsBumQxMOjuUbFJ+A==&encryptType=AES&method=bank.query.pay.details&signType=RSA2&timestamp=2022-11-23 11:35:00&version=1.0

签名得到 sign:

J2e7T++/XoizRP5Do3yignmQ4okiFcAPZdT8lEfmNUOA89gXma84UySXofMIEGd7h7O1nmghi9EOlsyiDF7g1WHcffio7Ga+AC6RsFWZXE/sLVw9tkkJUQxoiWCzrzlrHa/n+rXsvmWThQ19/JN5C2cs+GoZvTFx7dYBUQrq+aI89C08gBS/dzuDdykPBvnHpSeiSuqfvh18rCSwde4KQNui5spyQjDzLNa/PREkk/s6SR5B7XyYsqcU+dju2RncLLgM3gpS1Wv5N1dUhyKj4idduAKHLyJAYy7ECBZXvUN2Ovxga10+6ESn+36oj9bpZFqL33o2zjPwZTjV7zLYdA==

最终得到接口请求的数据：

{app_id = "33bfc65fe8e843eaaad0bb6a0eee9a41"

method = "bank.query.pay.details"

sign = "J2e7T++/XoizRP5Do3yignmQ4okiFcAPZdT8lEfmNUOA89gXma84UySXofMIEGd7h7O1nmghi9EOlsyiDF7g1WHcffio7Ga+AC6RsFWZXE/sLVw9tkkJUQxoiWCzrzlrHa/n+rXsvmWThQ19/JN5C2cs+GoZvTFx7dYBUQrq+aI89C08gBS/dzuDdykPBvnHpSeiSuqfvh18rCSwde4KQNui5spyQjDzLNa/PREkk/s6SR5B7XyYsqcU+dju2RncLLgM3gpS1Wv5N1dUhyKj4idduAKHLyJAYy7ECBZXvUN2Ovxga10+6ESn+36oj9bpZFqL33o2zjPwZTjV7zLYdA=="

timestamp = "2022-11-23 11:35:00"

version = "1.0"

data = "5tOZLBQUFAtiVrfAC9mHKMOgZUiR848C6Js2SVyBUJ2KmOlP2iOYBwu49V3AxJKdLogPJnsBumQxMOjuUbFJ+A=="

signType = "RSA2"

encryptType = "AES"}

SM4 签名：

◆代码示例

```
1.  /**
2.       * 签名
3.       *
4.       * @param content    待签名数据
5.       * @param privateKey 私钥
6.       * @param smUuid 通用唯一识别码
7.       * @return
8.       * @throws CapitalException
9.       */
```

```
    public static String sign(String content, String privateKey,String smUuid) throws CapitalException {
        try {
            PrivateKey sm2PrivateKey = getPrivateKey(privateKey);
            // 签名
            byte[] message = content.getBytes(CapitalConstants.DEFAULT_CHARSET_UTF8);
            Signature signature = Signature.getInstance(SIGN_SM3SM2_ALGORITHM);
            signature.setParameter(new SM2ParameterSpec(Strings.toByteArray(smUuid)));
            signature.initSign(sm2PrivateKey, new SecureRandom());
            signature.update(message);
            byte[] sign = signature.sign();
            //将签名结果转换为 Base64
            String signData = Base64.encodeBase64String(sign);
            return signData;
        } catch (Exception e) {
            throw new CapitalException(e);
        }
    }
```

◆请求示例

例如下面的请求示例，参数值都是示例值，业务系统参考格式即可：

```
app_id=   f764cc22dd3a4977bb4079c148827afe
method= bank.query.pay.details
version=1.0
    timestamp  =  2022-11-23 11:15:55

data=e56fdfa58d175a223d0a007cae978eedbc53701860beadcc412e326494d054f4164b7cf41cbd2df8c6d2baa447fded4fcf0c9a341536d07333d1cec598c28232b2e1788bce58c9fc4921ba124583b549

signType = SM2
encryptType = SM4
```

按照签名规则处理后最终待签名字符串为：

app_id=f764cc22dd3a4977bb4079c148827afe&data=e8cca98f8074333e45a044bd1f015076eb380c3edf0bb6a572ba4acc00c5f03422eecd4a935cd966d2a22145b7e2859ecc802f423c8f041f3f2719fdf4e632d5288c8873ec52b2f4625287d42cb41dad&encryptType=SM4&method=bank.query.pay.details&signType=SM2&timestamp=2022-11-23 11:29:10&version=1.0

签名得到 sign:
MEUCIQCgnh6HYGeFX4kcTVcaOtly72ch2qMfI67EYDwdCi/YzQIgAaSViNFSfC2jPKO5IVS46112B5ZpAOA9XIUlqGeYElY=

最终得到接口请求的数据：

{app_id = "f764cc22dd3a4977bb4079c148827afe"

method = "bank.query.pay.details"

sign = "MEUCIQCgnh6HYGeFX4kcTVcaOtly72ch2qMfI67EYDwdCi/YzQIgAaSViNFSfC2jPKO5IVS46112B5ZpAOA9XIUlqGeYElY="

timestamp = "2022-11-23 11:29:10"

version = "1.0"

signType = "SM2"

data = "e8cca98f8074333e45a044bd1f015076eb380c3edf0bb6a572ba4acc00c5f03422eecd4a935cd966d2a22145b7e2859ecc802f423c8f041f3f2719fdf4e632d5288c8873ec52b2f4625287d42cb41dad"

encryptType = "SM4"

}

1.5.2.2 接口返回参数验签

某些错误返回 sign 的值可能为空，此时不用验签，如 20003 错误。

接口返回包括同步返回（调用接口马上得到回复报文）和异步通知返回（推送的数据缴费后，缴费平台通过业务系统提供的接口主动通知），两者验签方式一致。业务系统只对缴费平台接口返回的 json 中 response 的值做验签。

◆获取待验签的值

返回的 JSON 中参数 response 对象的值，即为待验签的值。

◆调用验签函数

使用各自语言对应的 RSA2（即 SHA256WithRSA)或者 SM2 签名函数和缴费平台分配给业务系统的缴费平台公钥对待验签字符串进行验签，根据返回结果判定是否验签通过。

如 bus.unpay.data.sync 接口返回内容为：

{"response":"3uKoF8VSquhsJmGbho0ie1RCSVsU9Xl7wPApwBpFaVbubZYvwUSr+iZcmw7E0H7HzaZCBEzlg8Uneq1VlrulJTYB8yhSADYd1b/HM18IHMKQhMN64UV3TQqfTghhdaBVvzDx08nOw9FCgEbGj56WDN0Os+6nahEqrFzUF+f8kB5zMBtis10Ar4iey5sh1HGwSjGIKa7ndCJTnEIrctLbKmXfFNB5jCvIADOpH+RRFALdJ1aVnrAkZ6GK8IDGnRPd+AHqgu4KQbYxzV0fF0JS+rfuzJErAMdeYdtGaVGV6CF4670pm5pkMARgz3gXPGxs1+H4sp+pw8c01NCkHEF0MIeSJfZ9xwpT0y2J4A0z4X2UxK+SOxW3wzhQ3IiojJaxBp+ncppK0Ji/6TCx0YRsbp1pq/f50uLLGefSj1a/kefhbhQRXIOUFJE0hgeyV/8hM1mO4MxhkQCxUiCzE54Jt7mOZ9UZqM6tllKtOLhruj7DoKUKbhZUL8Jxx0jIhZs3Yc7NnteKsFNaNhltOsBoqDkWb3ZrUlmG2p187wozb3nBjj4upOL8Avb7INb8rIu8QnIU7AKkqPrxo5QXFHV5SG5/9PR1MF+APcq5H7pT/8nNaSq2D/C7IJFt1u+9eYkaG7wbzS/4ewmZ7xIslii9hd/TC9hXdPtKVpOwSkzFwcQchhJ7mElm3zPcgi5Keb1FhYmeBY19tj2qtZeG7dIWy393UG1AxrEpcssgRx3TvsxkgMVRXS76oEyG6hPAH1Zcg+rq4XL3feGaHcFKEPXbNgO5T+c8D58DSzbSV/iPile1OqJsCSv3DVH2M33ZKUsFdbflawueZBfDrtSwtjmviODIa4+lwkJZAIf1kY+hXgin+NY6cmKwSnO6pRSlmDCT8zjiSRm8DlgK2FGU+OHhO+Y6LdGYC6+ISuNbXY+3plxSNGwRp8MposqjLXNNs4TG6/y7e8CcimpNDGItkhQqEshE9q1lbBHQrL/7LHsN97BvWg5JprB7VeoFzk1W85+zVBTTxU7nTUy9KrmHLTLy9OtrOLpWok5QIsNkch2zg1AJ2QhEvAW0BB8wJRGoisET7zGqqpJ+oYEk8o7eZ71N+EHXZpDisKdVLVk8/LCV7uFb+ko5/FH4XXw8Yz3mttjNSEJp5dNaFErnfdZtVqmEAW7dpZT6Rp1slHPcYOEStSHzAYPa91RL1R06La/CHQOOc+mMi3MaCpsOB1pRp1GSuzu2vttpOkpN2RvQJB8Fin1aZFWJ2gh92E6dZpi8z4clkx+P47fY8WSWwqqIlBKRnqOOODxC5E2NWjXRlqIszi TSJ+J85UVSKnyTuS1V+4bAa3ABn1ysqRdF2FVhQNY89waMzqY4qRsrblC1s2INRER72iwAYbJ1HYQd4qoEPBCh/3IQC/TCfBocHGbPxBozRU8waieEXH8vxmyBfbTIuXqpZ7QkOA2ChL566FDiHxFOc9nxrYvWExzdANJFfQEUbrsH59nd2vC/+Avafzqt/HE6c0QarrYwONQjASfonj8Ill5WtW4RGJ8hy8bKsouRs+lDLOhfxOHIFVcfW1E2C+5nusVEO1n3kfw11C9R14rDo8GXNq2hbfVPxlpHzsMiKJOkG/rjVt4NEWAuVjiAoPFhEMihtcWldmSyYmuOuiR271Br8crtAKe6p3AZBRbG2hY6ShYB9BhQkAfcTVW1Izha7bH8XlFHAsqeWSXfanD15Wxq49Js3ARPwj7lPtaesxpkH2z5NKN/TuS/95YLmZ7MmJApBW82e+kQ6cupYALSSn7PNR+fHNWq05vEhV21HBzaekkPusiUSnoB2KzM9Yql/i/6xQLc8IbywuH/ByXpyRqvA4P63ZONHW7ZDSSHNttpg2kvKlth785wtItgDXEU7A1DnWy5AKg/QabS2dB70nv2MDX+n7OWLFjtcXRwLvTORNDVhuTc1sCU45CYn7NXP+ZX8pX/R1xXZuU8K/YPlupdCHP+ZMwEOtYYiegNXyMcMd1v4Pwnh5ntc7lC8Pcyb4SwCM9JYgIJj1/GMoqjVSNlNAWwqdcfIz/Hcrnx9nRYaFFQluRYX1VE04DHizY35jhbft0xi5iewUe0IuwdQ9oovsxz2/zSONpxm+yJmbKov85c2+gQQ/rWKkXbdwz1VEKmEmoJQQ2bxcN4TrIj1dUHSMjGHBdwrp5pFWbjmj3xLb8knJCYes9EO74kB8lrTfhxAXeWw9Dyh+b27K21dn6HU0YemE89fgc/eBkJH9oj6UYtHYOgAo3FCJIQqhIvln/Frtx/P8MXNn8ywVai4UY5/A8cTjlvd7SMI1WQkgWzuAxwWUDc/CNJydhCA8t08aBX7di02493bLwHFGChltHNFttoBw6r+c0hVHcdpNC9I71VS/LQBb1tdZDnRIARY9u6+elvJa2NH3XclcVXk8VXg8rIYDAFDts7v3iRUg==","sign":"YKcOosSoKLn+7WzWy306uIh8XyTQs7F17CodOP6dkHoWEWg86L+GCpS/WtwNt6Xb9DHsYHD9UMF9sBESFmQg4DCfLouwAZBBWJRHjeKe75tUSmahHdptt2QjOR6AfB5bSqyIXQJ3M4VKQ2xhOfTFL8TRq4hg7AJaotkKquZcvSQ="}

则待验签字符串为：

3uKoF8VSquhsJmGbho0ie1RCSVsU9Xl7wPApwBpFaVbubZYvwUSr+iZcmw7E0H7HzaZCBEzlg8Uneq1VlrulJTYB8yhSADYd1b/HM18IHMKQhMN64UV3TQqfTghhdaBVvzDx08nOw9FCgEbGj56WDN0Os+6nahEqrFzUF+f8kB5zMBtis10Ar4iey5sh1HGwSjGIKa7ndCJTnEIrctLbKmXfFNB5jCvIADOpH+RRFALdJ1aVnrAkZ6GK8IDGnRPd+AHqgu4KQbYxzV0fF0JS+rfuzJErAMdeYdtGaVGV6CF4670pm5pkMARgz3gXPGxs1+H4sp+pw8c01NCkHEF0MIeSJfZ9xwpT0y2J4A0z4X2UxK+SOxW3wzhQ3IiojJaxBp+ncppK0Ji/6TCx0YRsbp1pq/f50uLLGefSj1a/kefhbhQRXIOUFJE0hgeyV/8hM1mO4MxhkQCxUiCzE54Jt7mOZ9UZqM6tllKtOLhruj7DoKUKbhZUL8Jxx0jIhZs3Yc7NnteKsFNaNhltOsBoqDkWb3ZrUlmG2p187wozb3nBjj4upOL8Avb7INb8rIu8QnIU7AKkqPrxo5QXFHV5SG5/9PR1MF+APcq5H7pT/8nNaSq2D/C7IJFt1u+9eYkaG7wbzS/4ewmZ7xIslii9hd/TC9hXdPtKVpOwSkzFwcQchhJ7mElm3zPcgi5Keb1FhYmeBY19tj2qtZeG7dIWy393UG1AxrEpcssgRx3TvsxkgMVRXS76oEyG6hPAH1Zcg+rq4XL3feGaHcFKEPXbNgO5T+c8D58DSzbSV/iPile1OqJsCSv3DVH2M33ZKUsFdbflawueZBfDrtSwtjmviODIa4+lwkJZAIf1kY+hXgin+NY6cmKwSnO6pRSlmDCT8zjiSRm8DlgK2FGU+OHhO+Y6LdGYC6+ISuNbXY+3plxSNGwRp8MposqjLXNNs4TG6/y7e8CcimpNDGItkhQqEshE9q1lbBHQrL/7LHsN97BvWg5JprB7VeoFzk1W85+zVBTTxU7nTUy9KrmHLTLy9OtrOLpWok5QIsNkch2zg1AJ2QhEvAW0BB8wJRGoisET7zGqqpJ+oYEk8o7eZ71N+EHXZpDisKdVLVk8/LCV7uFb+ko5/FH4XXw8Yz3mttjNSEJp5dNaFErnfdZtVqmEAW7dpZT6Rp1slHPcYOEStSHzAYPa91RL1R06La/CHQOOc+mMi3MaCpsOB1pRp1GSuzu2vttpOkpN2RvQJB8Fin1aZFWJ2gh92E6dZpi8z4clkx+P47fY8WSWwqqIlBKRnqOOODxC5E2NWjXRlqIszi TSJ+J85UVSKnyTuS1V+4bAa3ABn1ysqRdF2FVhQNY89waMzqY4qRsrblC1s2INRER72iwAYbJ1HYQd4qoEPBCh/3IQC/TCfBocHGbPxBozRU8waieEXH8vxmyBfbTIuXqpZ7QkOA2ChL566FDiHxFOc9nxrYvWExzdANJFfQEUbrsH59nd2vC/+Avafzqt/HE6c0QarrYwONQjASfonj8Ill5WtW4RGJ8hy8bKsouRs+lDLOhfxOHIFVcfW1E2C+5nusVEO1n3kfw11C9R14rDo8GXNq2hbfVPxlpHzsMiKJOkG/rjVt4NEWAuVjiAoPFhEMihtcWldmSyYmuOuiR271Br8crtAKe6p3AZBRbG2hY6ShYB9BhQkAfcTVW1Izha7bH8XlFHAsqeWSXfanD15Wxq49Js3ARPwj7lPtaesxpkH2z5NKN/TuS/95YLmZ7MmJApBW82e+kQ6cupYALSSn7PNR+fHNWq05vEhV21HBzaekkPusiUSnoB2KzM9Yql/i/6xQLc8IbywuH/ByXpyRqvA4P63ZONHW7ZDSSHNttpg2kvKlth785wtItgDXEU7A1DnWy5AKg/QabS2dB70nv2MDX+n7OWLFjtcXRwLvTORNDVhuTc1sCU45CYn7NXP+ZX8pX/R1xXZuU8K/YPlupdCHP+ZMwEOtYYiegNXyMcMd1v4Pwnh5ntc7lC8Pcyb4SwCM9JYgIJj1/GMoqjVSNlNAWwqdcfIz/Hcrnx9nRYaFFQluRYX1VE04DHizY35jhbft0xi5iewUe0IuwdQ9oovsxz2/zSONpxm+yJmbKov85c2+gQQ/rWKkXbdwz1VEKmEmoJQQ2bxcN4TrIj1dUHSMjGHBdwrp5pFWbjmj3xLb8knJCYes9EO74kB8lrTfhxAXeWw9Dyh+b27K21dn6HU0YemE89fgc/eBkJH9oj6UYtHYOgAo3FCJIQqhIvln/Frtx/P8MXNn8ywVai4UY5/A8cTjlvd7SMI1WQkgWzuAxwWUDc/CNJydhCA8t08aBX7di02493bLwHFGChltHNFttoBw6r+c0hVHcdpNC9I71VS/LQBb1tdZDnRIARY9u6+elvJa2NH3XclcVXk8VXg8rIYDAFDts7v3iRUg==

/6TCxOYRsbp1pq/f50uLLGefSj1a/kefhbhQRXIOUFJEOhgeyV/8hM1mO4MxhkQCxUiCzE54Jt7mOZ9
UZqM6tllKtOLhruj7DoKUKbhZUL8Jxx0jIhZs3Yc7NnteKsFNaNhltOsBoqDkWb3ZrUlmG2p187wozb
3nBjj4upOL8Avb7INb8rIu8QnIU7AKkqPrxo5QXFHV5SG5/9PR1MF+APcq5H7pT/8nNaSq2D/C7IJFt
1u+9eYkaG7wbzS/4ewmZ7xIslii9hd/TC9hXdPtKVpOwSkzFwcQchhJ7mElm3zPcgi5Keb1FhYmeBY1
9tj2qtZeG7dIWy393UG1AxrEpcssgRx3TvsxkgMVRXS76oEyG6hPAH1Zcg+rq4XL3feGaHcFKEPXbNg
O5T+c8D58DSzbSV/iPile1OqJsCSv3DVH2M33ZKUsFdbflawueZBfDrtSwtjmviODIa4+lwkJZAIf1k
Y+hXgin+NY6cmKwSnO6pRSlmDCT8zjiSRm8DlgK2FGU+OHhO+Y6LdGYC6+ISuNbXY+3plxSNGwRp8Mp
osqjLXNNs4TG6/y7e8CcimpNDGItkhQqEshE9q1lbBHQrL/7LHsN97BvWg5JprB7VeoFzk1W85+zVBT
TxU7nTUy9KrmHLTLy9OtrOLpWok5QIsNkch2zg1AJ2QhEvAWOBB8wJRGoisET7zGqqpJ+oYEk8o7eZ7
1N+EHXZpDisKdVLVk8/LCV7uFb+ko5/FH4XXw8Yz3mttjNSEJp5dNaFErnfdZtVqmEAW7dpZT6Rp1sl
HPcYOEStSHzAYPa91RL1RO6La/CHQOOc+mMi3MaCpsOB1pRp1GSuzu2vttpOkpN2RvQJB8Fin1aZFWJ
2gh92E6dZpi8z4clkx+P47fY8WSWwqqIlBKRnqOOODxC5E2NWjXRlqIsziTSJ+J85UVSKnyTuS1V+4b
Aa3ABn1ysqRdF2FVhQNY89waMzqY4qRsrblC1s2INRER72iwAYbJ1HYQd4qoEPBCh/3IQC/TCfBocHG
bPxBozRU8waieEXH8vxmyBfbTIuXqpZ7QkOA2ChL566FDiHxFOc9nxrYvWExzdANJFfQEUbrsH59nd2
vC/+Avafzqt/HE6cOQarrYwONQjASfonj8II15WtW4RGJ8hy8bKsouRs+lDLOhfxOHIFVcfW1E2C+5n
usVEO1n3kfw11C9R14rDo8GXNq2hbfVPxlpHzsMiKJOkG/rjVt4NEWAuVjiAoPFhEMihtcWldmSyYmu
OuiR271Br8crtAKe6p3AZBRbG2hY6ShYB9BhQkAfcTVW1Izha7bH8XlFHAsqeWSXfanD15Wxq49Js3A
RPwj71PtaesxpkH2z5NKN/TuS/95YLmZ7MmJApBW82e+kQ6cupYALSSn7PNR+fHNWq05vEhV21HBzae
kkPusiUSnoB2KzM9Yql/i/6xQLc8IbywuH/ByXpyRqvA4P63ZONHW7ZDSSHNttpg2kvKlth785wtItg
DXEU7A1DnWy5AKg/QabS2dB7Onv2MDX+n7OWLFjtcXRwLvTORNDVhuTc1sCU45CYn7NXP+ZX8pX/R1x
XZuU8K/YPlupdCHP+ZMwEOtYYiegNXyMcMd1v4Pwnh5ntc7lC8Pcyb4SwCM9JYgIJjl/GMoqjVSNlNA
WwqdcfIz/Hcrnx9nRYaFFQluRYX1VE04DHizY35jhbft0xi5iewUe0IuwdQ9oovsxz2/zSONpxm+yJm
bKov85c2+gQQ/rWKkXbdwz1VEKmEmoJQQ2bxcN4TrIj1dUHSMjGHBdwrp5pFWbjmj3xLb8knJCYes9E
O74kB8lrTfhxAXeWw9Dyh+b27K21dn6HUOYemE89fgc/eBkJH9oj6UYtHYOgAo3FCJIQqhIvln/Frtx
/P8MXNn8ywVai4UY5/A8cTjlvd7SMI1WQkgWzuAxwWUDc/CNJydhCA8t08aBX7di02493bLwHFGChlt
HNFttoBw6r+c0hVHcdpNC9I71VS/LQBb1tdZDnRIARY9u6+elvJa2NH3XclcVXk8VXg8rIYDAFDts7v
3iRUg==

◆验签代码示例

RSA 验签：

```
/**
     * 验签
     *
     * @param content   待验签字符串
     * @param sign
     * @param publicKey 公钥
     * @param charset   字符编码
     * @return
     *
     * @throws IppApiException
```

```
    */
   private static boolean rsa256CheckContent(String content, String sign, String publicKey, String charset)
           throws IppApiException {
       try {
           PublicKey pubKey = getPublicKeyFromX509(IppConstants.SIGN_TYPE_RSA,
                   new ByteArrayInputStream(publicKey.getBytes()));

           Signature signature = Signature.getInstance(IppConstants.SIGN_SHA256RSA_ALGORITHMS);
           signature.initVerify(pubKey);

           if (BaseStringUtils.isEmpty(charset)) {
               signature.update(content.getBytes());
           } else {
               signature.update(content.getBytes(charset));
           }
           return signature.verify(Base64.decodeBase64(sign.getBytes()));

       } catch (Exception e) {
           throw new IppApiException("RSAcontent = " + content + ",sign=" + sign + ",charset = " + charset, e);
       }
   }
```

SM4 验签：

```
/**
     * 验签
     *
     * @param content   待验签数据
     * @param sign      签名串
     * @param publicKey 公钥
     * @param smUuid 通用唯一识别码
     * @return
     */
    public static boolean verifySign(String content, String sign, String publicKey,String smUuid) throws CapitalException {
        try {
            PublicKey sm2PublicKey = getPublicKey(publicKey);
            // 验签
```

```
            byte[] signature = Base64.decodeBase64(sign.getBytes(CapitalConstants.DEFAULT_CHARSET_UTF8));
            byte[] message = content.getBytes(CapitalConstants.DEFAULT_CHARSET_UTF8);
            Signature sm2SignEngine = Signature.getInstance(SIGN_SM3SM2_ALGORITHM);
            sm2SignEngine.setParameter(new SM2ParameterSpec(Strings.toByteArray(smUuid)));
            sm2SignEngine.initVerify(sm2PublicKey);
            sm2SignEngine.update(message);
            ///sm2SignEngine.update(message, 0, message.length);
            return sm2SignEngine.verify(signature);
        } catch (Exception e) {
            throw new CapitalException(e);
        }
    }
```

1.5.3 业务系统通知接口签名说明

业务系统收到缴费平台的异步通知（缴费确认或退付确认）处理完业务后，需封装参数并返回给缴费平台，缴费平台收到返回数据并确认成功后，则停止向业务系统定时发送该数据的异步通知。如果一直没收到返回或一直返回失败，则最多通知 5 次，这时需要业务系统主动调用查询缴费状态接口或查询退付状态接口，查询相关业务状态。

加密，签名规则：

1,先按照接口（3.2.1 或 3.2.2）封装返回的数据得到：

{ "code":"10000","msg":"success","doc_number":"8423edc0c3114812987c096233c814fa","refund_number":""}

2,加密得到 response 的值：

2,1AES 加密得到 response 的值：

IKnDQpbKICfglvonxFBuILGgPzjZgnJ0DqdZo/GPpdK3dQK7n+/bGfwO0ecCwbyJDMgBzJ+TU+6syrjgth5wW/YEsBmTPvROgvYRPCXGhEDywaDD47HEs/HWsNYrJAgMBAAECgYAoEu5d3u2uqtoSGuU84VU/7ck6BE9lri+EZH83bDy8YJcOJ5zGWyOsRUkoTm/qJgtjIN5BXAVjgoYrr8D4i2BiwmCTYyxe+ydmlzGQRNFMCZh7eQ9EVXB/8mNphC1gECeoIROjTF71e/CRtQd1ngxkyNgq/nQ38Rewlu0GLOojkQJ

2,2SM4 加密得到 response 的值：

d2b6e587e7f5f08e4e23af75b2698e0ef268e57ce5dba37921c8645fa6ef2267e140622f97cccbb72e24569ad6b6e164fc84653329de4d9fd

3,对 response 的值使用签名，得到 sign:

3,1RSA2 签名：

mw/ffRziT3nWvDxt6NLbOhtpN7Khf2GAA+ns7F8cuT97IHZ7pOkwgOvbaAjrsgyP4xeXrYVQEBbAkAkfW1edd+1QcFCcVkCDVTcIvqc2mkEI8cRbGoKOLaSuZoXAVzuIr2VJENJSvnxGvL4eUBbaIAOCMG6j3fPUHCDAkEAsOkOH4ugrpSUaMZYggAkhs1rDvmRs3pfQJuaJBB7aCOA6JCXhGUuYj3DZUOquNznLXHx3Lou49UexsnDu1NBRwJAZ/XGiBrp+FG6dhkskTBzKOMzO1kOeBUwR4gPW8FQnex6t3ha9aVPbBOz1vOOBktKIsFQqcx9+lxksDDQ765LVg

3,2 SM4 签名：

MEQCIA7mIPfIKewGCjeMSerprsjBZsyJTBHAlyVffIpOj9tUAiA/lefucO8wQTHMO8KvHacpRuMlip8ZmR5KN/k61MgUkA==

4,最后封装得到返回给缴费平台的数据：

RSA 方式：

{"response":"IKnDQpbKICfglvonxFBuILGgPzjZgnJODqdZo/GPpdK3dQK7n+/bGfwOOecCwbyJDMgBzJ+TU+6syrjgth5wW/YEsBmTPvROgvYRPCXGhEDywaDD47HEs/HWsNYrJAgMBAAECgYAoEu5d3u2uqtoSGuU84VU/7ck6BE91ri+EZH83bDy8YJcOJ5zGWyOsRUkoTm/qJgtjIN5BXAVjgoYrr8D4i2BiwmCTYyxe+ydmlzGQRNFMCZh7eQ9EVXB/8mNphC1gECeoIROjTF71e/CRtQdlngxkyNgq/nQ38Rewlu0GLOojkQJ",
"sign":"mw/ffRziT3nWvDxt6NLbOhtpN7Khf2GAA+ns7F8cuT97IHZ7pOkwgOvbaAjrsgyP4xeXrYVQEBbAkAkfW1edd+1QcFCcVkCDVTcIvqc2mkEI8cRbGoKOLaSuZoXAVzuIr2VJENJSvnxGvL4eUBbaIAOCMG6j3fPUHCDAkEAsOkOH4ugrpSUaMZYggAkhs1rDvmRs3pfQJuaJBB7aCOA6JCXhGUuYj3DZUOquNznLXHx3Lou49UexsnDu1NBRwJAZ/XGiBrp+FG6dhkskTBzKOMzO1kOeBUwR4gPW8FQnex6t3ha9aVPbBOz1vOOBktKIsFQqcx9+lxksDDQ765LVg"}

国密方式：

{"response":"d2b6e587e7f5f08e4e23af75b2698e0ef268e57ce5dba37921c8645fa6ef2267e140622f97cccbb72e24569ad6b6e164fc84653329de4d9fd01b81b4e738f9d8a44a6803d9a8330e9fab1e4456c1f66d","sign":"MEQCIA7mIPfIKewGCjeMSerprsjBZsyJTBHAlyVffIpOj9tUAiA/lefucO8wQTHMO8KvHacpRuMlip8ZmR5KN/k61MgUkA=="}

如果使用了 sdk 直接调用 sdk 提供的方法：

```
1.  String appPrivateKey = "业务系统私钥";
2.  String enKey= "加密秘钥";
3.  //封装返回给缴费平台的数据
4.  String params = "返回给缴费平台的数据";
5.  String signType = "签名方式";
6.  String encryptType = "加密方式";
7.  try {
8.      Map<String,String> map = IppSignature.notifyRsaSign(params,appPrivateKey,
    enKey,signType,encryptType);
9.      return map;
10. } catch (IppApiException e) {
```

```
11.     e.printStackTrace();
12. }
```

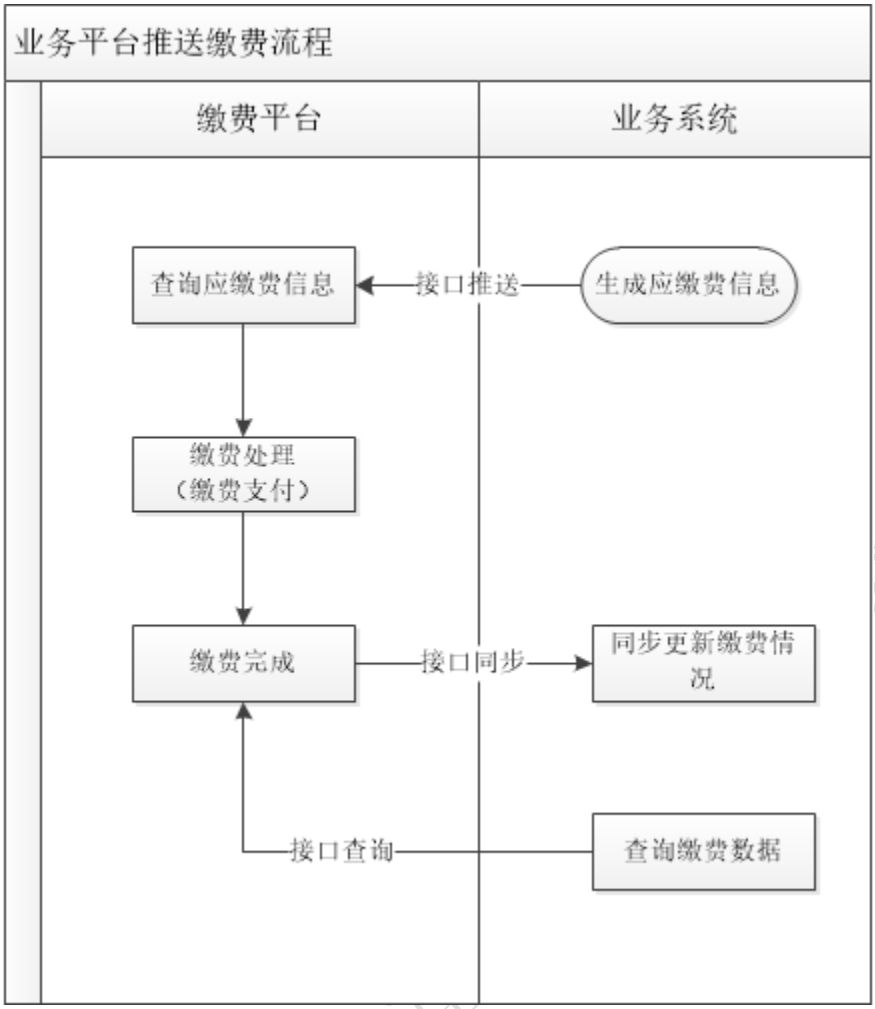
2 业务流程

2.1 接口列表

类别	接口名称	接口服务标识
缴费平台接口	待缴费数据推送接口	bus.unpay.data.sync
	查询缴费状态接口（根据业务标识查询）	bus.query.pay.status
	查询缴费状态接口（根据收费项目查询）	bus.query.pay.status2
	调用缴费平台 h5 支付界面接口	public.open.pay.h5
	查询某天已缴费数据接口	public.query.paid.data
	查询某天已缴费数据接口（多单位分页）	public.query.paid.data.page
	退付数据推送接口	bus.refund.pay
	查询退付状态接口	bus.query.refund.status
业务系统接口	缴费成功通知接口	
	退付成功通知接口	
	回调票据信息通知接口	

2.2 流程说明

业务系统产生待缴费数据后，通过缴费平台提供的待缴费数据推送接口把待缴费数据推送到缴费平台，缴费平台进行收缴操作，缴费完成后通过业务系统提供的推送接口把已缴费数据状态推送到业务系统，业务系统也可通过查询缴费状态接口主动发起查询缴费状态。



3 接口描述

所有接口中参数后面的数字表示该参数最大长度，如果没有则表示该参数最大长度不限。

3.1 缴费平台接口

缴费平台提供的接口。

3.1.1 待缴费数据推送接口

当业务系统生成待缴费数据后调用该接口，把数据推送至缴费平台。如同一条数据重复推送且缴费平台未缴费，则会用最后一次推送的数据覆盖之前推送的数据，如已交费则返回错误信息。

3.1.1.1 服务标识 bus.unpay.data.sync

3.1.1.2 请求业务参数

参数	类型	是否必填	描述	示例值
region	String(6)	是	行政区划	500000
dept_id	String(32)	是	执收单位编码	50011111220000000009
doc_number	String(64)	是	数据标识，需保证在业务系统不重复，每条缴费数据唯一	2019011201 00234
payment_unit	String(50)	是	缴款人（单位）	张三
extra_payment_unit	String(50)	否	额外缴款人（单位），如果值存在，支付时会提示用户选择哪个缴款单位开票	
payment_total	Price(11)	是	应收金额，单位为元，精确到小数点后两位，取值范围[0.01,100000000]	99.99
data_type	String(1)	是	资金性质，详见附录4	1 非税，2 往来
phone	String(11)	否	手机号码，电子票据推送主要通道	
id_card	String(32)	否	证件号码	
notify_url	String(256)	否	异步通知地址，缴费平台缴费后通过该地址异步通知业务系统，详见 1.4.2.2，为空则不通知，但可以通过查询缴费状态接口查询。注：该地	http://api.xx.com/v1/pay_notify.do

			址必须是 http 协议，不能是 soap 协议。	
ticket_notify_url	String(256)	否	向业务系统回调电子票据信息。注：该地址必须是 http 协议，不能是 soap 协议。	
punish_decision_no	String(32)	否	处罚决定书号	
remark	String(150)	否	备注	
is_apply_virtual_account	String(1)	否	是否申请虚拟账户，1 申请，0 不申请	0
is_apply_pay_code	String(1)	否	是否申请缴款码，1 申请，0 不申请	0
items	josnArray	是	该数据对应的缴费项目列表信息,详见下面表格说明	[{},{},…]

items 字段说明：

注意：计费数量 *缴费标准=实收金额;所有缴费项目实收金额之和等于应收金额

收费项目类型必须和资金性质一致

参数	类型	是否必填	描述	示例值
item_code	String(100)	是	缴费项目编码	103021901
bi_number	Number(5)	是	计费数量，精确到小数点后两位	1.00
standard	Price(11)	是	缴费标准，单位为元，精确到小数点后两位，取值范围[0.01,100000000]	99.99
actual_amt	Price(11)	是	实收金额，单位为元，精确到小数点后	99.99

			两位，取值范围[0.01,100000000]	

3.1.1.3 返回结果参数

参数	类型	是否必填	描述	示例值
doc_number	String(64)	是	数据标识，需保证在业务系统不重复	
h5_pay_url	String(256)	否	该数据对应的 H5 支付页面 URL，默认有效时间为一天	
qr_code	String	否	h5_pay_url 对应的二维码，base64 图片格式。默认有效时间为一天	...
virtual_bank_acc	String(50)	否	虚拟户账号，申请时返回	
virtual_bank_acc_name	String(60)	否	虚拟户账号名称，申请时返回	
pay_code	String(20)	否	缴款码，申请时返回	0

3.1.1.4 参数示例

- 请求参数

{

"sign": "eMv19z5cXaXUBbrM9VXxZQHl0816CUZ83G9l3uTBoXyxb5HULkgg5Uimzvdg0s8C8KoLHOq1Rb/HeQyi1ZKV4t1r2KZ6Wa4XRVd2OAOJ8+DFzMmsGTFXEEBcST/v2jLCYCMCjkcnYwVtHbgZNPrJKZozQ0OmNPtsPAY+sNqIg9s=",

"method": "bus.unpay.data.sync",

"data": "加密后字符串",

"app_id": "516670cac3e1451f9ad3d6ffd0ca120b",

"version": "1.0",

"timestamp": "2019-07-31 17:18:59"

}

- 返回参数

成功

{
"response": "返回内容加密后字符串",
"sign":"fmSNQ5xv7V1nEVQpCkBFNJy4COIm4vm1q6tedFxMdOdONB/MVrKF8zU2iaTmjWC5PfSYDlgaCFxT3lCsP6Uxa2nPIeNJLwNR7nLsM6M8OgqCdr9IBIUSyXdJlIZpPG9oi7Xv1o6XDO4Duv9mUvF/VCUgDiOTLYZY2/4CfH650DA="
}

失败

{
"response":"返回内容加密后字符串",
"sign":"UbYw9HmTkzVm5cyojeqfu27jh+4gorbWxNB/T8TPFq1gKBi2wKZQ8F9nCLDYYtsvvht/wti+RWga4nT91XaxQgWXfWc6STSGtssZlRgIxH+8pzg4r0E06W23ybSltnM9od1/JZm2zHSSNO6dfUM4S/tcCdU/mR/c2Y1rP9dOFaA="
}

3.1.2 查询缴费状态接口（根据业务标识查询）

业务系统通过该接口主动向缴费平台查询指定数据的缴费状态。

3.1.2.1服务标识　bus.query.pay.status

3.1.2.2请求业务参数

参数	类型	是否必填	描述	示例值
doc_number	String(64)	是	数据标识，需保证在业务系统不重复	
dept_id	String(32)	是	执收单位编码	5001111122000000009

3.1.2.3　返回结果参数

参数	类型	是否必填	描述	示例值

doc_number	String(64)	是	数据标识，不能包含中文，需保证在业务系统不重复	
payment_total	Price(11)	是	应缴金额，单位为元，精确到小数点后两位，取值范围[0.01,100000000]	99.99
is_confirm	String(1)	是	缴费状态，0 未缴费，1 已缴费	0
pay_channel	String(2)	否	缴款渠道，缴费后返回	见附录 1
confirm_date	String(19)	否	缴费时间，缴费后返回，yyyy-MM-dd HH:mm:ss	2019-07-23 11:35:22
order_no	String(32)	否	缴费平台订单号，缴费后返回。不唯一，合并支付时多条业务数据对应一个订单号	
pay_code	String(32)	否	缴款码，缴费后返回	
bill_batch_code	String(32)	否	票据代码，交易含电子票业务，缴费后且开票成功后返回	
bill_no	String(32)	否	票据号码，交易含电子票业务，缴费后且开票成功后返回	
bill_random	String(32)	否	票据校验码，交易含电子票业务，缴费后且开票成功后返回	
bill_h5_url	String(300)	否	电子票据 H5 查看地址，交易含电子票业	

			务，缴费后且开票成功后返回	

3.1.3 查询缴费状态接口（多条件查询）

业务系统通过该接口主动向缴费平台查询指定证件号码或收费项目数据的缴费状态。

3.1.3.1服务标识　bus.query.pay.status2

3.1.3.2请求业务参数

参数	类型	是否必填	描述	示例值
dept_id	String(32)	是	执收单位编码	5001111122
id_card	String(20)	否	证件号码，和 item_code 至少填一个	
item_code	String(32)	否	收费项目编码，和 id_card 至少填一个	
start_date	String(10)	是	开始日期。yyyy-mm-dd 格式，开始日期和结束日期最大间隔一年	
end_date	String(10)	是	结束日期。yyyy-mm-dd 格式，开始日期和结束日期最大间隔一年	

3.1.3.3　返回结果参数

参数	类型	是否必填	描述	示例值
dept_id	String(32)	是	执收单位编码	5001111122000000009
id_card	String(20)	否	证件号码	

item_code	String(32)	否	收费项目编码	
list	集合数组	是	Json 格式，list 中的数据按下面表格字段组成	[{},{},…]

list 参数具体内容如下：

参数	类型	是否必填	描述	示例值
doc_number	String(64)	是	数据标识	
payment_total	Price(11)	是	应缴金额，单位为元，精确到小数点后两位，取值范围[0.01,100000000]	99.99
is_confirm	String(1)	是	缴费状态，0 未缴费，1 已缴费,9 支付中	0
pay_channel	String(2)	否	缴款渠道，缴费后返回	见附录 1
confirm_date	String(19)	否	缴费时间，缴费后返回，yyyy-MM-dd HH:mm:ss	2019-07-23 11:35:22
order_no	String(32)	否	缴费平台订单号，缴费后返回。不唯一，合并支付时多条业务数据对应一个订单号	
pay_code	String(32)	否	缴款码，业务支持在缴费后返回	
bill_batch_code	String(32)	否	票据代码，交易含电子票业务，缴费后且开票成功后返回	

bill_no	String(32)	否	票据号码，交易含电子票业务，缴费后且开票成功后返回	
bill_random	String(32)	否	票据校验码，交易含电子票业务，缴费后且开票成功后返回	
bill_h5_url	String(300)	否	电子票据 H5 查看地址，交易含电子票业务，缴费后且开票成功后返回	
id_card	String(20)	否	证件号码，数据存在证件号码时返回	

3.1.4 获取缴费平台 H5 支付 URL 接口

业务系统支付前调用该接口,成功后返回对应的 url,访问返回的 url，即可打开支付页面

3.1.4.1服务标识　public.open.pay.h5

3.1.4.2　请求业务参数

参数	类型	是否必填	描述	示例值
doc_number	String(64)	特殊可选	数据标识，需保证在业务系统不重复。和证件号码、学生姓名不能同时为空。	2019091101000002
dept_id	String(32)	是	执收单位编码	50011111220000000009
effective_time	Number(5)	否	有效时间,单位分钟,整数，范围[5, 86400]	5

3.1.4.3　返回结果参数

参数	类型	是否必填	描述	示例值

doc_number	String(64)	是	数据标识，需保证在业务系统不重复	
h5_pay_url	String(256)	是	h5 支付页面的 URL	
qr_code	String	是	h5_pay_url 对应的二维码，base64 图片格式	...

3.1.5 查询某天已缴费数据接口

业务系统查询某天在缴费平台的已缴费数据

3.1.5.1服务标识　public.query.paid.data

3.1.5.2　请求业务参数

参数	类型	是否必填	描述	示例值
confirm_date	String(10)	是	缴费日期，yyyy-MM-dd	2019-07-23
dept_id	String(32)	是	单位编码	50011111220000000009
id_card	String(20)	否	证件号码	
item_code	String(32)	否	收费项目编码	

3.1.5.3　返回结果参数

参数	类型	是否必填	描述	示例值
dept_id	String(32)	是	执收单位编码	50011111220000000009
list	String	是	Json 数组字符串，list 中的数据按下面表格字段组成	[{},{},…]

list 参数具体内容如下：

参数	类型	是否必填	描述	示例值
doc_number	String(64)	是	数据标识，需保证在业务系统不重复	
payment_total	Price(11)	是	应缴金额，单位为元，精确到小数点后两位，取值范围[0.01,100000000]	99.99
is_confirm	String(1)	是	缴费状态，默认 1	1
pay_channel	String(2)	是	缴款渠道	见附录 1
confirm_date	String(19)	是	缴费时间，yyyy-MM-dd HH:mm:ss	2019-07-23 11:35:22
order_no	String(32)	是	缴费平台订单号	
pay_code	String(32)	否	缴款码	
bill_batch_code	String(32)	否	票据代码，交易类型含电子票业务，且开票成功时返回	
bill_no	String(32)	否	票据号码，交易类型含电子票业务，且开票成功时返回	
bill_random	String(32)	否	票据校验码，交易类型含电子票业务，且开票成功时返回	
bill_h5_url	String(300)	否	电子票据 H5 查看地址，开票成功时返回	
id_card	String(20)	否	证件号码，数据存在证件号码时返回	

remark	String(150)	否	缴费数据备注	

3.1.6 查询某天已缴费数据接口（多单位分页）

业务系统查询某天在缴费平台的已缴费数据（分页查询）。注意：请求业务参数在一个流程中除 current_page 可变更外，其他参数不能变动；避免返回数据缺失或重复。

3.1.6.1 public.query.paid.data.page

3.1.6.2 请求业务参数

参数	类型	是否必填	描述	示例值
current_page	Integer	是	指定显示返回结果中的第几页	1
page_size	Integer	是	指定返回结果中每页显示的记录数量。范围 1000~10000	1000
confirm_date	String(10)	是	缴费日期，yyyy-MM-dd	2019-07-23
dept_ids	ArrayList	是	单位编码集合。范围 1~500 个	['1','2']
id_card	String(20)	否	证件号码	
item_code	String(32)	否	收费项目编码	

3.1.6.3 返回结果参数

参数	类型	是否必填	描述	示例值
current_page	Integer	是	当前页号	1
page_size	Integer	是	每页数量	1000
total	Integer	是	总条数	20000
page_count	Integer	是	总页数	20

data	ArrayList	是	缴费数据集合	

data 参数具体内容如下：

参数	类型	是否必填	描述	示例值
dept_id	String(32)	是	执收单位编码	5001111122000000009
doc_number	String(64)	是	数据标识，需保证在业务系统不重复	
payment_total	Price(11)	是	应缴金额，单位为元，精确到小数点后两位，取值范围[0.01,100000000]	99.99
is_confirm	String(1)	是	缴费状态，默认 1	1
pay_channel	String(2)	是	缴款渠道	见附录 1
confirm_date	String(19)	是	缴费时间，yyyy-MM-dd HH:mm:ss	2019-07-23 11:35:22
order_no	String(32)	是	缴费平台订单号	
pay_code	String(32)	否	缴款码	
bill_batch_code	String(32)	否	票据代码，交易类型含电子票业务，且开票成功时返回	
bill_no	String(32)	否	票据号码，交易类型含电子票业务，且开票成功时返回	
bill_random	String(32)	否	票据校验码，交易类型含电子票业务，且开票成功时返回	

bill_h5_url	String(300)	否	电子票据 H5 查看地址，开票成功时返回	
id_card	String(20)	否	证件号码，数据存在证件号码时返回	
remark	String(150)	否	缴费数据备注	

3.1.7 作废待缴费数据接口

业务系统通过调用该接口作废推送的未缴费数据，作废后的数据不能再缴费。

3.1.7.1服务标识 bus.invalid.unpay.data

3.1.7.2 请求业务参数

参数	类型	是否必填	描述	示例值
doc_number	String(64)	是	数据标识，需保证在业务系统不重复	2019011201 00234
dept_id	String(32)	是	执收单位编码	5001111122 000000009
operator_id	String(32)	否	业务系统操作人 id	

3.1.7.3 返回结果参数

参数	类型	是否必填	描述	示例值
doc_number	String(64)	是	数据标识，需保证在业务系统不重复	

3.1.8 退付数据推送接口

当业务系统需要给已缴费的数据进行退付时调用该接口。

业务系统先确认是否需要实际的退付功能，还是只在缴费平台记录退付数据，确认后缴费平台开启对应功能。

3.1.8.1服务标识　bus.refund.pay

3.1.8.2　请求业务参数

参数	类型	是否必填	描述	示例值
dept_id	String(32)	是	执收单位编码	5001111122000000009
doc_number	String(64)	是	数据标识，需保证在业务系统不重复	
refund_number	String(64)	是	退付订单号，需保证在业务系统不重复	
payment_total	Price(11)	是	退付金额，不能大于支付金额。单位为元，精确到小数点后两位，取值范围[0.01,100000000]	99.99
refund_des	String(100)	是	退付原因	
data_type	String(1)	是	资金性质，详见附录4	1
order_no	String(32)	是	缴费平台订单号	20190722140010030302 00089909
id_card	String(18)	否	证件号码	
phone	String(11)	否	手机号码	
pay_code	String(32)	否	缴款码，如果有则必填	
refund_url	String(256)	否	通知地址，实际退付填写，只做记录则不用填写且只有非税业务才回使用，其他业务同步返回退费状态	http://api.xx.com/v1/pay_refund.do

items	ArrayList	是	退费项目	

items 字段说明：

注意：计费数量 *缴费标准=退费金额;所有缴费项目退费金额之和等于退费金额

参数	类型	是否必填	描述	示例值
item_code	String(100)	是	退费项目编码	103021901
item_name	String(30)	是	退费项目名称	
bi_number	Number(5)	是	计费数量，精确到小数点后两位	1.00
standard	Price(11)	是	缴费标准，单位为元，精确到小数点后两位，取值范围[0.01,100000000]	99.99
actual_amt	Price(11)	是	项目退费金额，单位为元，精确到小数点后两位，取值范围[0.01,100000000]	99.99

3.1.8.3 返回结果参数

参数	类型	是否必填	描述	示例值
doc_number	String(64)	是	业务系统传递的数据标识	
refund_number	String(64)	是	业务系统传递的退付订单号	

3.1.9 查询退付状态接口

业务系统通过该接口向缴费平台主动查询指定数据的退付状态。

3.1.9.1 服务标识 bus.query.refund.status

3.1.9.2 请求业务参数

参数	类型	是否必填	描述	示例值
dept_id	String(32)	是	执收单位编码	5001111122000000009
doc_number	String(64)	是	数据标识，需保证在业务系统不重复	
refund_number	String(64)	是	退付订单号，需保证在业务系统不重复	

3.1.9.3 返回结果参数

参数	类型	是否必填	描述	示例值
doc_number	String(64)	是	业务系统传递的数据标识	
refund_number	String(64)	是	业务系统传递的退付订单号	
payment_total	Price(11)	是	退付金额，单位为元，精确到小数点后两位，取值范围[0.01,100000000]	100.00
refund_status	String(1)	是	退付状态，0 未退付，1 已退付，2 退付中，3 退付失败	1
refund_time	String(19)	否	退付时间，yyyy-MM-dd HH:mm:ss	2019-07-22 14:09:39

3.2 业务系统接口

业务系统提供的接口。包括缴费成功通知接口和退付成功通知接口。

业务系统收到缴费平台的异步通知（缴费确认或退付确认）处理完业务后，需封装参数并返回给缴费平台，缴费平台收到返回数据并确认成功后，则停止向业务系统定时发送该数据的异步通知。如果一直没收到返回或一直返回失败，则最多通知 5 次，这时需要业务系统主动调用查询缴费状态接口或查询退付状态接口，查询相关业务状态。

3.2.1 缴费成功通知接口

缴费平台将缴费结果通知业务系统。业务系统推送的数据缴费成功后（未缴费不会通知），缴费平台会向业务系统提供的接口（待缴数据推送接口中的 notify_url 字段）发异步通知，业务系统收到通知后验签、解密、处理业务后，需告知缴费平台。

注：notify_url 不为空时该接口必须提供。

3.2.1.1 接口地址 待缴数据推送接口中的 notify_url 字段

3.2.1.2 请求业务参数

参数	类型	是否必填	描述	示例值
response	String	是	加密前参数详见下表	
sign	String(350)	是	验签详见 1.5.5	

response 解密后详细参数：

参数	类型	是否必填	描述	示例值
amt	Price(9)	是	已缴金额，单位为元，精确到小数点后两位，取值范围[0.01,100000000]	100.00
confirm_date	String(19)	是	缴费时间，yyyy-MM-dd HH:mm:ss	2016-07-22 14:09:39
doc_number	String(32)	是	业务系统唯一标识	
notify_time	String(19)	是	异步通知时间	2016-07-22 14:10:49

order_no	String(32)	是	缴费系统订单号	2019072214 0010030302 00089909
pay_channel	String(2)	是	缴款渠道	见附录 1
pay_code	String(32)	否	缴款码	
bill_batch_code	String(32)	否	票据代码，交易类型含电子票业务，且开票成功时返回	
bill_no	String(32)	否	票据号码，交易类型含电子票业务，且开票成功时返回	
bill_random	String(32)	否	票据校验码，交易类型含电子票业务，且开票成功时返回	
bill_h5_url	String(300)	否	电子票据 H5 查看地址，开票成功时返回	

3.2.1.3 返回结果参数

参数	类型	是否必填	描述	示例值
response	String	是	详见下表 response 参数	
sign	String(350)	是	签名详见 1.5.6	

response 加密前详细参数：

参数	类型	是否必填	描述	示例值
code	String(32)	是	成功 10000	10000
msg	String(100)	否	失败时返回	

doc_number	String(32)	是	业务系统唯一标识	

3.2.2 退付成功通知接口

缴费平台将退付结果异步通知业务系统。退付处理完成后，缴费平台会向业务系统提供的接口（退付数据推送接口的 refund_url 字段）发异步通知，业务系统收到通知后验签、解密、处理业务后，需告知缴费平台。

注：refund_url 不为空时该接口必须提供。

3.2.2.1 接口地址 退付数据推送接口的 refund_url 字段

3.2.2.2 请求业务参数

参数	类型	是否必填	描述	示例值
response	String	是	加密前参数详见下表	
sign	String(350)	是	验签详见 1.5.5	

参数	类型	是否必填	描述	示例值
doc_number	String(32)	是	交易业务系统唯一标识	
refund_number	String(32)	是	退付业务系统唯一标识	
amt	Price(9)	是	退付金额，单位为元，精确到小数点后两位，取值范围[0.01,100000000]	100.00
refund_date	String(19)	是	退付时间，yyyy-MM-dd HH:mm:ss	2016-07-22 14:09:39
notify_time	String(19)	是	异步通知时间	2016-07-22 14:10:49

3.2.2.3 返回结果参数

参数	类型	是否必填	描述	示例值
response	String	是	详见 response 参数	
sign	String(350)	是	签名详见 1.5.6	

response 参数：

参数	类型	是否必填	描述	示例值
code	String(32)	是	成功 10000	10000
msg	String(100)	否	失败时返回	
doc_number	String(32)	是	业务系统唯一标识	
refund_number	String(32)	是	退付业务系统唯一标识	

3.2.3 回调票据信息通知接口

3.2.3.1 接口地址 待缴数据推送接口中的 ticket_notify_url 字段

3.2.3.2 请求业务参数

参数	类型	是否必填	描述	示例值
response	String	是	加密前参数详见下表	
sign	String(350)	是	验签详见 1.5.5	

response 解密后详细参数：

参数	类型	是否必填	描述	示例值
doc_number	String(32)	是	业务系统唯一标识	
notify_time	String(19)	是	异步通知时间	2016-07-22 14:10:49
order_no	String(32)	是	缴费系统订单号	2019072214001003030200089909
pay_channel	String(2)	是	缴款渠道	见附录 1
pay_code	String(32)	否	缴款码	
bill_batch_code	String(32)	否	票据代码，交易类型含电子票业务，且开票成功时返回	
bill_no	String(32)	否	票据号码，交易类型含电子票业务，且开票成功时返回	
bill_random	String(32)	否	票据校验码，交易类型含电子票业务，且开票成功时返回	
bill_h5_url	String(300)	否	电子票据 H5 查看地址，开票成功时返回	

3.2.3.3 返回结果参数

参数	类型	是否必填	描述	示例值
response	String	是	详见下表 response 参数	
sign	String(350)	是	签名详见 1.5.6	

response 加密前详细参数：

参数	类型	是否必填	描述	示例值
code	String(32)	是	成功 10000	10000
msg	String(100)	否	失败时返回	

4 调用步骤说明

4.1 调用完整流程

下面以待缴费数据推送接口为例：

◆第一步，根据业务接口请求参数封装得到待加密 JSON 字符串：

{"id_card":"500223199911223344","doc_number":"441cc0fc34714d9ebebca630a3278baa","extra_payment_unit":"","remark":"备注","payment_total":0.01,"notify_url":"http://api.bosssoftcq.com.cn:8006/epay/pay_notify","is_apply_virtual_account":"0","phone":"18983129127","data_type":"1","region":"500000","dept_id":"10000","payment_unit":"甲兜儿","items":[{"standard":0.01,"item_code":"100086112","bi_number":1.0,"actual_amt":0.01}]}

◆第二步，对待加密 JSON 字符串加密得到 data:

AES 算法加密：

fc0/poGxLjI4goJXe+KNAFfX9+NDWxEn7og+TgbWQKDUySEn7ol8X5uAAcBg3zUySGMUmZWe1nFa/RL1JIIECfo3Xy1SiQ6pQVgowfnIkbuOQqWgnHgSpihEwflrtAWAMRYg2oBDHtv0KBACdzDIm8bOtrtAxdxziQzg27CMNlaEeduQWbW259XpKN67CFgBY54vQDQPRDAupxQff/lvfn/eR0QxN3SORagLhDfgd0hCdL27PZGyHMSX3nKK8bSNmZ/cseA87kUtQvl1B0A74bQ5vJSBeqdxRT2sj9wfsxM3yn08uTg/LFfV6m6gxrJHaV+95DPh6AK9maTY2ZvdIwH11ZJD0RLZLHVgZmbFTiV1Dy++twcvUJKdN8XTQGHK79lkmh7rGqw2IdGZA5ISHYLuVqgYbXSHybHldTvhlfHNjGFleB1CA0YG30+jnd11kp+fr/V7CLgfJnmtvp+nqMeQBPxU8Q+v1QKgAspmVhh/hsm6VwJ8W2J4rOdC59vtiCDs5EYgOqTr6FPQ7F/k0lwK7hBoy6PO48ObG29GDAcC0QNKXIYoyWvATxFw+yuB

SM4 算法加密：

baf35a4268315d5c06e08b72e524e4fe623e13a2a46c8e74ce1f068a2a8f7e782cd6e96d7ef0462a97d0752ff7ed51f6d43f1effb9f813baa24014a70e18aba2f6051f2167903937e157f78a7c40db15b4aafa2b9928bc16013f05eb2788343469ec3eb7db442a90bfc6e07d902cc8e3b2fccfeb46d83dd233db1f149ead4aa1e19684a0bba2f7dfa571adb73c0788c5c9e66d335f25b835ac3a1a187813633a364

4ab10e80faed7dc6e260a40ecfc26a7ff477ed85967c25e9ad9817a813d6d0f70d59d5ddb597c2998d96aa34355815cfa424cede99339d973d412ef44f2943d349afd91390b50303ef8034d30339b74558b74861a965b543e3df9f05404a125d975ae3d297b954582c902baa9e93109e0abcccd8133e86bebef49aded475bc9fd3f633df757753a5ec3fc1f8f02002e279f5fc871c98672030c9f5e1f22d7cb4382a657a6c57d7c0685e4436a7856d29f5d9be9aa8a9eccfbe0cd0a00afe56ae8391db58530f1b0d74a2a94fedaffa298bfba5adfee641e0c8b68817d457628289d623ee0e370eb2d2b856867e13f70e80ff667a2e6f7eaf2f48942179042accc12266676e4b1dc598e4a4799f7ef4f3b12764400dcdf8b0994509 3e0a0c1

◆第三步，按照签名规则对公共请求参数排序，得到待签名字符串

RSA 签名规则：

app_id=33bfc65fe8e843eaaad0bb6a0eee9a41&data=fc0/poGxLjI4goJXe+KNAFfX9+NDWxEn7og+TgbWQKDUySEn7oI8X5uAAcBg3zUySGMUmZWe1nFa/RL1JIIECfo3Xy1SiQ6pQVgowfnIkbuOQqWgnHgSpihEwflrtAWAMRYg2oBDHtv0KBACdzDIm8bOtrtAxdxziQzg27CMNlaEeduQWbW259XpKN67CFgBY54vQDQPRDAupxQff/lvfn/eR0QxN3SORagLhDfgd0hCdL27PZGyHMSX3nKK8bSNmZ/cseA87kUtQvl1B0A74bQ5vJSBeqdxRT2sj9wfsxM3yn08uTg/LFfV6m6gxrJHaV+95DPh6AK9maTY2ZvdIwH11ZJD0RLZLHVgZmbFTiV1Dy++twcvUJKdN8XTQGHK79lkmh7rGqw2IdGZA5ISHYLuVqgYbXSHybHldTvhlfHNjGFleB1CA0YG30+jnd11kp+fr/V7CLgfJnmtvp+nqMeQBPxU8Q+v1QKgAspmVhh/hsm6VwJ8W2J4rOdC59vtiCDs5EYgOqTr6FPQ7F/k0lwK7hBoy6PO48ObG29GDAcC0QNKXIYoyWvATxFw+yuB&encrypt_type=AES&method=bus.unpay.data.sync&sign_type=RSA2&timestamp=2022-11-24 17:09:11&version=1.0

SM2 签名规则：

app_id=f764cc22dd3a4977bb4079c148827afe&data=baf35a4268315d5c06e08b72e524e4fe623e13a2a46c8e74ce1f068a2a8f7e782cd6e96d7ef0462a97d0752ff7ed51f6d43f1effb9f813baa24014a70e18aba2f6051f2167903937e157f78a7c40db15b4aafa2b9928bc16013f05eb2788343469ec3eb7db442a90bfc6e07d902cc8e3b2fccfeb46d83dd233db1f149ead4aa1e19684a0bba2f7dfa571adb73c0788c5c9e66d335f25b835ac3a1a187813633a3644ab10e80faed7dc6e260a40ecfc26a7ff477ed85967c25e9ad9817a813d6d0f70d59d5ddb597c2998d96aa34355815cfa424cede99339d973d412ef44f2943d349afd91390b50303ef8034d30339b74558b74861a965b543e3df9f05404a125d975ae3d297b954582c902baa9e93109e0abcccd8133e86bebef49aded475bc9fd3f633df757753a5ec3fc1f8f02002e279f5fc871c98672030c9f5e1f22d7cb4382a657a6c57d7c0685e4436a7856d29f5d9be9aa8a9eccfbe0cd0a00afe56ae8391db58530f1b0d74a2a94fedaffa298bfba5adfee641e0c8b68817d457628289d623ee0e370eb2d2b856867e13f70e80ff667a2e6f7eaf2f48942179042accc12266676e4b1dc598e4a4799f7ef4f3b12764400dcdf8b09945093e0a0c1&encrypt_type=SM4&method=bus.unpay.data.sync&sign_type=SM2&timestamp=2022-11-24 17:14:16&version=1.0

◆第四步，对待签名字符串进行签名，得到 sign:

RSA 签名：

nBuPuAUUlPtGZPin+Sscg0sFARaAFQdQ7BEZsh76V5prdQbdWPHhz7t3ITliAGq2T2fzUjLAaLHjtWcD
DULbqx97cposLuCrXbQpDY5A7htNUBvUhVhc7MOFtI2L3vAmouREYGD6GxcAmKR5ENJEY3n1IcCF
/uE52R8q/fBL6ZfluCLsOAo4CCEkmFCh/25I3oR85ENmDyHxsrRavpPIFN1Q4a6iYtbLcFuUBn3z5hM
Pc+Kl/CQcby6L6zBrK4NgUEp9d9D/w1FO18oFwdeCw4VitAPUGm56KjFzpMMrUywZ0coCciy4i4rq
65GSR/yacigETTQgGEXMtSRpfOGBXQ==

SM2 签名：

MEUCIAZ/L9GrtA1djJ9+XbJ5h56biaIQi6EYwCpU8f4lAO4xAiEAhYp++JoERLW6IK+YNPy9BN1UzfjRu
+7KNCgdjViXh/c=

◆第五步，封装最终请求参数：

RSA 方式：

{"sign":"nBuPuAUUlPtGZPin+Sscg0sFARaAFQdQ7BEZsh76V5prdQbdWPHhz7t3ITliAGq2T2fzUjLAa
LHjtWcDDULbqx97cposLuCrXbQpDY5A7htNUBvUhVhc7MOFtI2L3vAmouREYGD6GxcAmKR5ENJE
Y3n1IcCF/uE52R8q/fBL6ZfluCLsOAo4CCEkmFCh/25I3oR85ENmDyHxsrRavpPIFN1Q4a6iYtbLcFuUB
n3z5hMPc+Kl/CQcby6L6zBrK4NgUEp9d9D/w1FO18oFwdeCw4VitAPUGm56KjFzpMMrUywZ0coC
ciy4i4rq65GSR/yacigETTQgGEXMtSRpfOGBXQ==","data":"
fc0/poGxLjI4goJXe+KNAFfX9+NDWxEn7og+TgbWQKDUySEn7ol8X5uAAcBg3zUySGMUmZWe1nFa
/RL1JIIECfo3Xy1SiQ6pQVgowfnIkbuOQqWgnHgSpihEwflrtAWAMRYg2oBDHtv0KBACdzDIm8bOtrt
AxdxziQzg27CMNlaEeduQWbW259XpKN67CFgBY54vQDQPRDAupxQff/lvfn/eR0QxN3SORagLhDf
gd0hCdL27PZGyHMSX3nKK8bSNmZ/cseA87kUtQvl1B0A74bQ5vJSBeqdxRT2sj9wfsxM3yn08uTg/L
FfV6m6gxrJHaV+95DPh6AK9maTY2ZvdIwH11ZJD0RLZLHVgZmbFTiV1Dy++twcvUJKdN8XTQGHK79
lkmh7rGqw2IdGZA5ISHYLuVqgYbXSHybHldTvhlfHNjGFleB1CA0YG30+jnd11kp+fr/V7CLgfJnmtvp+
nqMeQBPxU8Q+v1QKgAspmVhh/hsm6VwJ8W2J4rOdC59vtiCDs5EYgOqTr6FPQ7F/k0lwK7hBoy6P
O48ObG29GDAcC0QNKXIYoyWvATxFw+yuB","app_id":"33bfc65fe8e843eaaad0bb6a0eee9a41","
version":"1.0","timestamp":"2022-11-23 15:44:36","sign_type ":"RSA2","encrypt_type" : "AES"}

国密方式：

{"app_id" : "f764cc22dd3a4977bb4079c148827afe"," method" :"bus.unpay.data.sync"," sign":
"MEUCIAZ/L9GrtA1djJ9+XbJ5h56biaIQi6EYwCpU8f4lAO4xAiEAhYp++JoERLW6IK+YNPy9BN1UzfjR
u+7KNCgdjViXh/c=","timestamp" :"2022-11-23
15:31:44","version" :"1.0","data" :"baf35a4268315d5c06e08b72e524e4fe623e13a2a46c8e74ce1
f068a2a8f7e782cd6e96d7ef0462a97d0752ff7ed51f6d43f1effb9f813baa24014a70e18aba2f6051f
2167903937e157f78a7c40db15b4aafa2b9928bc16013f05eb2788343469ec3eb7db442a90bfc6e0
7d902cc8e3b2fccfeb46d83dd233db1f149ead4aa1e19684a0bba2f7dfa571adb73c0788c5c9e66d3
35f25b835ac3a1a187813633a3644ab10e80faed7dc6e260a40ecfc26a7ff477ed85967c25e9ad981

7a813d6d0f70d59d5ddb597c2998d96aa34355815cfa424cede99339d973d412ef44f2943d349afd
91390b50303ef8034d30339b74558b74861a965b543e3df9f05404a125d975ae3d297b954582c90
2baa9e93109e0abcccd8133e86bebef49aded475bc9fd3f633df757753a5ec3fc1f8f02002e279f5fc
871c98672030c9f5e1f22d7cb4382a657a6c57d7c0685e4436a7856d29f5d9be9aa8a9eccfbe0cd0a
00afe56ae8391db58530f1b0d74a2a94fedaffa298bfba5adfee641e0c8b68817d457628289d623ee
0e370eb2d2b856867e13f70e80ff667a2e6f7eaf2f48942179042accc12266676e4b1dc598e4a4799
f7ef4f3b12764400dcdf8b09945093e0a0c1","sign_type":"SM2","encrypt_type ":"SM4"}

◆第六步，发起 post 请求并得到接口回复报文：

RSA 方式：

{"response":"FS9PfL2TUOAWgXdM/9Qfpx5jKZK5GvZF90r6jvUm+6ivkA3DSfwRLNq9K776zBn
PPwvaA9mDr4GiIxnBiruzQRKhEuROKcq4c5EB0B8BC0wu/18PFy+PJ7gLVsDyvuqRBY+6cwN8m
of+4fOX/JveX7N4ZdsxrPItDRoMP/EdQEvtiYFreuegXVn/0Vln/HwGNqY8136q/mZu+O05Lga0Bi
jyPXJXbzDMMUioV7LOkoAC++h7AWQc6CLtJq4OrSTNJBkWuzSeQW2LubM7kbjAuJDMKNJhg
CGjU/HTURxTK+a70e2oVQ859lflwOXummqeNFWscvhOWLMo/BnN64Wjwe8HC8Y8ufzqxnIT
zxsYOGZEu3NSi6ltJtGr56L/tIH3AosK6aypnHvifWAsAmvaeo5SvYF9vMrMM7kf5gBq+6HPAv/R
+q++2DQZLO1xast3KlU29I6393Cm0sNGnkxcqOPEJrFE8tRpWbs+gLDm8IU8qjNvUBE4qpFIAyq
cQCH76QkUde6r9wExwnSVI/gOretwQPf3iKvsS48enzcMTGXcci1IUr4dIYlRP+BQClfMt7oYrAYh
7xocOmu9d1b/ngszhqKRtme1ulimqPOZVmcJMRUMgpFEKml+awHWI62MzlOT3oFnKVlE+rnf
HUCLiTmrKoOWUh0lM5rKhUdxx3bM9sh0Sf5exFFBXhGlOmosBgxnHyEBMow4ey0F96OiKFec
dypJh43pHSyGf9H0LndimOlFlijWUpHqeLLBZqNeEzn8v2PJ6X+vY93QNJejWtOzstTX0EFpWsW
FpW28VrNQQkpBFoPrhObgGVw6jqtq9YllSYWqhbfY8aRcgPbRvJDSVGJUqI0TNGTgAehQ/4D2
MhsWDnB56i/Cer5GqW35Grtxn7AJDct9YaZnIc0yrQYU4hA0BQHMUM5sygy/B/XNgNzbWrTT
IS8EhcrONGOBzrL3xcslFCRAKAyF4VHtcBvgUdAJTskqiXQsnLSzOL6x78W3Yrlqp0Z+8u/+HPnPX
KyW6VyegvlScmyvgkKlGkF1QNKtgwX+bqAJY1RbbVWRtsJx906as1piZXCd8JjWfGurQlFaLsAulr
Ze8SH5ZshbICOmdxGTHW1VB32nvz3LItA+rmkTHjYWZeqSlEl/Gdcsu34F+jAZ9y60fErGiescQcl
5biRjPStau9N0NgE5yrk+PMYqkKNbswOFB88NgIEZ7aokDq1lCPHzmAZFrOOu3HiBEJ0eU+hClY
NodPUWy9X24MvTxJOEEPoILHu99eBeTH+xmkdEPVsvf3eJgjIPvPV3Vse6oWkiExG98oYMp4o/
dvKmnaDspYqn7FJyT4vcCpWDt7a6/A/nX64vp6XYshCliyV3JAav2JVaeMpcLCBXah7REOQHzUc
JypR1Qj+elGp2tWC62nEhBUVnhI6z00V47JZg3c0axrTW1MfMyBtC3iBTr4NRuQ9qPc3X0n86u
4jLwwyy33C9SSAxhJx3hB53FqhkJu1mMCD9EPtrXaIB/oBDx3hK7/L0Vg0JIEg+3y7lFlUh0X2MH
aVkdAiq/AQZoA835Yf4kHvUkS5rtG1kMrgC7SECU96K1hm5FNQRGSGt6WxS3cCya8d2creuT9
uvUc6lvVsuhSTF+INPCMUvEh3pzW6V2lpuKXaw0m4pqQJCGQckGRPWibT6tOFB58PfsQOGur
LHcQ6X10PJyUXZvvHI1k6fdbgB7FVxv8SNeC6PVlDZYz6bT/hX1mZBN9UwrGb9Epp7jSwW1Trj
TsJ7a+IYMsp9FdyZ+wHN37U8a0k3LFitYt5zHwPykYdEZAyaKK3WYdrrg0VoNKz957+MBRU9cc
WvEgGq5nUlIYzsKXwVa0KOPtQTpklbx/hmINifv3Z36m6XktonvFWpTSyOHLZgDbPcWhLhLs3s
aAMm3eyAuYVXMB449ogETOOfXyoWh0XwV0IUqWKUbwm+kRusU42RYkj3Y/JZ7CArL3L/rKC
Wv7iqVCGRJLdP7vvqqs71vJwC3mIB/tnI8D4Gzo77YR5lBzlvnO0F35MP2CAKOMRxNOysUCTq/
Hc5Y53T/o4tou3elHMHv3Q8tZ8oSuUwdlAK410JWzVn1ncvVXUvbj0r0chOJ7N4/AiWcGvy0r+Z

+FJwIirIo4ODWhWrqWFkq/Yb/R04M5vHqisvvRTcEfwGpDwTCARjvQdFYpUQYND6gpjpL6wJG
s/U27kHJlScRSB9PLOvEsZnhrjtFyM/7C5Nu149QymvQ9ZZYLu9q8x4xmNRRJygWt6TtWXkUIYI
Q9O6EaZ5RUjzYDl5ejNhsXECjsYitG5fG2zcLZbwguldflRK1Y4mD9CbizEnUAUkdWIIXevYzgLStaP
Qy2DeWMtV3MDSwlRaUlgzF0L8yz4RHq8QV/ookVEnWBvLaonn5UzlTp52CsdnBy7KxMUxtQg
u/HB11iXuNuQ/0Z9mOKznOJGFK1N+K7mWg7auS+yZouLgjinBoA8W0CsaS8aC8p1uP28yI4SbE
S4yqg==","sign":"ARt0PU5XYBG+Lol6kdtVjHtA44N9XB78S45e9D1SEMpBFaBrMYHJBCeviPK
Gds1jW9k7ZTMi9fEDxTGD6kY24yaor9b5FXUvdwtSDC2CJCPPoEHhzUbXAmglgrbKL16xgt8kb8
r49jjjEgoELyyi9FFN510btMrkzAYSRVPATEK4b6xCUKWsjInew9A2Zu5y1nCxEld3du0dUuJ1h6x
hGgyS4HG+bt1y7KF7HlvK02KN2S7RtAN0vYRicscJgPoZ1AtoMi/2GcYGYtFTwhdDHiua9QWX6
WfC0fKsc0WtSX+gBHkgFkR711MhnMq7bhRgHphUK4OqPvBeLjTUGObKSg=="}

国密方式：

{"response":"0151cceae4a06f69325f22a66d684192c21a32ae8a65d202ea0411f10623df1867
842d2504916f93e9049440e29263d180dd00d78474aa48ee6f5e3282c5496004a3be33e5436
1469bb28873bfe52f1bb38493d5836f8db160ef8c77f57ff55cd8927651767c0effdd6ac4a8a84
239a894f0d0737f084e842e3e4b3be2ff7f6a6e20b3cd44aa62674ef0cca554a80760712033b6
98f552e234a6ae6d96db70dad9a64d23496768c8f071e51c2f3b3d23c449e5fc4770082870c5
82c1ae0d83fb4a63e4748bef0315d3c5939cea82418d32150308ca59fa990fa28a1074b9a2890
5d576bb64a3012e249b2874fa7140d291e6e7d8074687646d65794c4bc917017c96395aacc8
d51a1ade36138ee98f5f942216e70443f68a5acaf2f863aa2e327c8f0714aa86bf3a88a028b451
028835cf70e89f8973732d357f8388321192f10265046bdf9522bbbafd0dc37c2e2e9de24f6b3
417613714b6b5d61a9c5f18633aedeee60567b6e1bf94cb529dc9e3ff83dbfd7ded0d77e9a5b
df44d2af60ee6a99707168ca4bb9a764355af7bd78153c890e7963cef3d05e1743c4916677cb
3858541242c55a0d852f539a292df8c076c75aa56c4351bf476070f909c4f415c72b7c208d4e1
a2b2f89eb982c9222ee439654f744a79bab4fa3f19da24fad05b5eb39c12bb222b207757d036
ee348aa2a7d576420f722d51065145ddc8380d9444fe14263c1867d32c656a56f17b8935543a
af82be620287cffbe45c440138a14caa525cc23ab64fdd97f6f1a6a2f4d5730c400c9541d6cf082
a3967e2c34e1f5fbce81e841284afcf53c2b0d86286359efed96f53559770feb18bb8b2e6b911
19ba0fd56c29c7db32d6dbae94883cb6a1f8299bf1c2252b3f9d329a2be3810125894effc3538
b3036a4457f258ad9164064be85603f3da9de89c6662efd70fb06b1d839a38796a61d3ad39ef
34495dbd7f1c3782e8e6f055bc298bff2094fcd95ebb4739faf6a74cdb00875c9fff96ccff043f70
ec3f48782b10f6080871af66ba69a0063e1f2aa9491c337ec29677c6d23babd80e6758c745d69
e50a7948b463f44fc0cf243bedeed06130e87def4f977127c4e51dadd1219add77119765cc3ad
36c3fb55d99b93cfbdd350448f06b8593f291b096fd24d960575661b5d11b119410d6ab28192
2fb75151a6089d6ebb8c52db9ed84348abf2b11410efa0fc4ad74eeb572dddddc21d47ecaa22
92cd49bb2e799692a244f665fe38d53abbba3f3b70197d00407dd98d9e748b49a7612394254
7126ab39278211c638cad2a3455434c8055c76f899f8fb93080c29fd42c4fb55292f771c8d7e63
fe9082cc3f400809b449fdac59e8294a9d32a7b8e6f9440aca3f447857de1bca9633000b2fd1b
87013de3731502bf8a2ee4ec413a9108c79f13c8998d2bb67b901931f8237741d389c01e3502
055f13b4a38926ba37946d970544270f4a8907f3c44cde213960ae46e0f4189c4d3ec143922f0

ba86a22aa615baf3164d5177173ac4dad53fd8a175e0212e0496d95c3bb6f4e7240a21ec0561
6c2d0e535ccfddaba6a6ff15fa6af1be7d0d96ef2565d531fe72ac76fba4314c5d117021bc33a7
9fccd463fbb47a7bddba83766c1461ec4516c9623f4ad40a88605af29adf9dad9e431db8fa805
6d1bdd874fe8d265e2b66e618d9aac4fb4b09246a1d9d0b21c5f693bfa7b69f785110c6a4226
42c61d183fe2f22ffa9d9d2276608191ff582f5bbb64a2ef78855f170bbab6ae76b6314d9a453e
b642e75043748d63089ea60bce88ca6a46132a35d797375242d40071007ffc433a62ecda4421
50dc3abd388ebb62a41cfb2629436367e83d59794b3596a72cd13f0b9cff9d293e0a64809b3c
7314d0b24f766433850349219e8a5dbfcd6ce24d4b835dd8283e136c9866082cc56dd932cfd7
5c2489768faa03ddba3247747fd22007111445dc2677d5becd6f78627728963c23904bd0d841
aa2a6579eedf1a2e9213fd7f1dfd39b0d78036d64a42cbc1c4a4fd210ef4cb602e324f85c1a21f
b803d732987677e2ee50cdc125972621d6664edfd2c9ad1fb9f870d336d6505ee0feb922cba1
72699f86e0e006b22e4782782e6d157e888de549cc585cf7032d989589062671bdf9659e5a53
bc333adc176983d61857636399bc2256d4bfadb40b28b6a049c3ce4f99ee76c3e8acb5eb6a08
eb60c742f4ab9ea155d4f360cdd9cb3c0d0cf4e3e23b315a29608847a6fcd6d2159463ae5d735
9ec502def8c885b189b6811eb7986316f37df6bf9161c2e25ef7c3c5f91f74001186033c691945
3f0f2eead16f596c140e838fad588253dce869d55e9c5a1859289a099f9e1a11ee40cfe560785
bbeac6da64a50032558c801150e7188e0371d505c10c2e789008cd1901628bc899d20fbdb51
4cb3fe43b37e70557aa789b77d64cdcf97821b4dce38d38eff0bdc2f48b74b2f2d5f1cac8782f1
eb0b49e3f6e6a66e500940f5ef6fab1a862cdce86f77000b83a83c73af3accc93c0d8be2e8d2e1
8d7143e33788aa4e21a122f67afac58aab9fd18ada3294f7fe28d69c8b77807744241a806cf99
b31015fd4e7925c0e30c6cf34967b8e7adf68a37083efc3e9608e4e48586bee0ed3b8a065fc0a
9f238bd4fc67f3bb91aec20b34","sign":"MEYCIQD9Rlx3TrfvcajXN9qa0lIq+xTYN6L21YSyW5gF
TfZX3QIhAKFNMSm8nlaGZDK3cq4q3cvEPxin3izt0KHZv9/ME2ud"}

◆第七步，对接口回复的报文按照验签规则进行验签，验签通过后对公共响应参数 response 的值解密得到具体返回内容

RSA 验签规则以及 AES 解密 response 数据得到具体返回内容

{"code":"10000","docNumber":"fbe88f550e904187bfd6ac0592a3ce50","h5PayUrl":"https://
ipp.bosssoftcq.com.cn:51103/h5/pages/results/results?bizId=ba582b60eb2548db87b0caa0c
b2d3905&fromtype=2&vde=10000&sysType=2&sign=f0d0be1ed75476993d730d44232cc66
d","msg":"新增或修改成功
","qrCode":"
aVxAAAEUklEQVR42u3dPXarMBCGYVGxDJaKl8oyqJgbJM2PMHZS3MKTvC584hgeVzqSRp9Ekf
/3KlhYWFhYWFhYWFhYWFhYWH/NOkp/zSKPZS9l3ebz//t0vn39f6mXzF9vi35cN71pxcJKarXP
W7063DfJ4/RlK1NvKedfjyq0n7N7sbASWpt9Xe9rN59WBavf3rRdrfWHz9/Ewspu1V6kt5zKaLs6
P1pH419gYf0GS1p7WaVdLa2faF1JffUxk2Bh/QbtXMWaVJst789jJiysLbYzNjHTL0wqssAHbQx0+Zll
ltZ8fJpgF+pwSX+kXWLjKCyslJb0tuEl/wDaeldtXMWaVJst789jJiysLbYzNjHTL0wqssAHbQx0+Zll

SysjFZvPnpzbEP+I31JTA5tOa0DKXf1VSysBFaMO2g/YcV/7160x1i9t5GnuQIWViKrNhCbCg+pBu
kLAt5oNr3kRU0UCyuDdYSKUJ8ItAXg3kXoTKLEQVL/eGBhpbV66dOr/cP/FOxz5CC8GH9hYSW
wngJsfXwUBlPWgXgS6GVNFAsridUHSddpQu9A+tAoJDrluUCEhZXMsq/FkszFog271v3DvpYyVo
mwsJJai5Z74jLAPlmmudeGxDJuPdhcsLDyWmvcpzWu/baJcgkl0ocNoW6zPlhYaSxLKHuZszeaE
HLwaP/TxVhYWS3PbIr4fXeZZjmsQHRgYaW1ioWYY6bZ97XY4pivAGy6jxELK6cVa/zFzuLxJHPvL
Gax2LNeXMp0k2nGwspgHfaN3SLj1sTZs8/WlfgKGRZWTqssPlG+zII96T/E/fU6LKyslkb2/RbPNA
ff18D8nJIZCyuxZT1G8XPWwgLwMD66bFwULKyU1hGPXQtJh7DeZctkxQ6nml/m5bCwklgh7jB
LCDEXjzuEbbpKy8vzrLCwUlh2knjYxeUDojCEKnpA4TB5xsJKaBWfDfSrpzHr4/RwyvLdmAkLK4cl
Y1Z53MAlMfQZj5zVBrdiYaW1iq1tWe9gW9HLKIRi6ZuzeLCwPtzS0E7Yex6unuIO9mUf8m334y
8srDSW9Q7zeJa+xds8378MZxNiYeW0bInXHoQl4TUcubONkTfZX+xRx8LKYPkCl5+6ac+Di+u8e
xnjzAcWVlLLZry+Q3c8Z9z2sFy3r9z1Q1hYOazYT2jcYXyGxOV5iZfZBRZWRmuo/izDkpjGOiUe0H
B5thAWVlJrtfaiZ+RPNj229WA/XtYPYLvd746FlcSyZ+EOmeZhX4tXTENh9O1zdbGw8liWYrMcp6
4F2BHjwzKZYGHlt0Qs+BCKQfHkZX+S1u1zt7Cw0lge+LEtWus4KdZSkRdLpzf5aCysBNY1zeOjIot
w+pnixbfpbm+fJ42F9cnWf3lhYWFhYWFhYWFhYWH9HesfdgQEPiYmVAIAAAAASUVOR
K5CYII="}

SM2 验签以及 SM4 解密 response 数据得到具体内容：

{"code":"10000","docNumber":"22c55775ee8945a2ba94adaafa90955c","h5PayUrl":"https:/
/ipp.bosssoftcq.com.cn:51103/h5/pages/results/results?bizId=ea246dd5049e409796d65b2c
e298987c&fromtype=2&vde=10000&sysType=2&sign=6a3e523073418b486140f4a00923d0
e5","msg":"新增或修改成功
","qrCode":"
aVxAAAEN0lEQVR42u3dTXarMAxAYTFiGSwVlsoyGOFHsK0fAn0ddBC1N4M0aeDLyEe2LCtSfu4
hWFhYWFhYWFhYWFhYWFhYf83apT3G4816/J3X49UynR9ux9vzaZXh+N/xSqbjjnntN81YWEm
t+n4d98ZIt+qr+nYbXt90gNK/zt2LhZXQ0pFT7Kmc1jlyXsOsPr3AOqTG19AbsbDyW3uNE2KDplR
GQ4mCWFi/xuoBpNSrS40T5y310eZMBQvrN1g2Zyp15PSw0dYPQxtDTbDwgoWV1xIdKvP63af
/rbexsD7b8o9z5JyDpvilQ7+5rRWGb+VXsbA+2WpjqF1dw4HOo+pnGk/G9ump9uUEFlZOqyV+
2kKgZ4TiorjvBbSVhLsECyul1TKcNg2yTa/iw0ZbMttu2HazB4aFlcRqWc/6yR4Hkg8lbs60WloICy
ujVfzHMSda2jyqXbJojOkB5G6NjIWVw3KR4Iwik+aLphJ3gXuwaKuGt7UCFlYeq2d/elqoaBTRWp
9ig0Yvuc+JYmElsRbdxF10Q6Bv9g7uf26S1MLGjoWV1LJg4auWW560h4hF18hOWEWwsJJaRdc
Ag48iGlR6TlSZL3KiWFh5rLC7q1lP2yHTIy1W63NNEGFh5bNKzOyvenULG73AswWQwY86LKy8l
oWD3dXyL7oKbkVAupKYrBoUCyujtftsv8Ra/qmf1bUUqTvM+FDrg4WVwNKjKpd6NmkrhL54ttJ
+CQUSWFg5LbcAbulQve+uplnrIO7yq1hYOaxwVlcXyu4Uuivtv36nYGEltfw1/QCXNiGx/jy2KRx2
hgsWVlKrZX9c9aYeWRefIu3xZP9q/oWFlcLq5Q4+4VnCK3dCd9Bds4f1NhZWFksm14TEqvUD6
DrwhD4lMxZWUst3EAy1PmKHcy1pFA8uFiyslFZIc2pVf18ed3pyG2HFZUdnLKyk1jJt4YCKO6FbL
sHCWm/6+RYWVjqrB4ZVrGxNs56j/7q2eLZmJePNuV8srBRWcWdwReJmr5sfWRue0KcECyuxd
TnAVbQtg53kHULL2fGhXg4LK4cVe5LYeRU7rhgE9wsTU8HCymrZURUNIO5qLfrUE43ucbePjIW
VxArtp5566Y/lZh6FhZXZ0sJmd4BLtI5zG669S/pemTzNv7CwPt+S0IHHuvIsrsm++PK25/poLKwc
1h5+DlRHzioxVVTej6+st/lVLKwM1vuC4a2YU0PJduk7O2BhJbXC7q67cNSyTgkNGi6/LYSFldTyn
Ujc0mHwxfuuvazrRiVYWHkt91u4rrtaKG2wzrIuMXoz/8LCymhtoSFP7MrTe5f0gh833cLCSmy5

1bK2XZu1K49YTXPdFcDCymxpot9VLYul/N3PJ1qydLivacbCymL5NfIWcqJ6ftdtBU+h3B8LK6f1Iw8sLCwsLCwsLCwsLCwsLKy/Y/0DBy9+W3uyX9AAAAAASUVORK5CYII="}

◆第八步，处理业务

4.2 示例

4.2.1 请求服务示例（JAVA）

4.2.1.1使用缴费平台 sdk

目前 sdk 只提供 java 版。缴费平台 SDK 封装了加密、签名实现，只需在创建 Default 对象时设置请求服务地址（serverUrl)、应用 id（app_id)、应用私钥（private_key)、缴费平台公钥（ipp_public_key)、加密密钥（aes_key)、国密签名 userId（sm_uuid）、签名方式（sign_type）、加密方式（encrypt_type）即可，报文请求时会自动进行签名。

```
1.  IppClient ippClient = new DefaultIppClient(serverUrl, appId,
2.          privateKey, publicKey, aesKey, smUuid, signType, encryptType);
3.  // 实例化具体 API 对应的 request 类,类名称和接口名称对应,当前调用接口名称：
    bus.unpay.data.sync
4.  BusUnpayDataSyncRequest request = new BusUnpayDataSyncRequest();
5.  // SDK 已经封装了公共参数，这里只需要传入业务参数
6.  request.setData(
7.          "{\"phone\":\"18983129127\",\"id_card\":\"500223199912128899\",\"data
    _type\":\"1\"," +
8.                  "\"doc_number\":\"201901040000011234000234\",\"sys_type\":\"1\"
    ," +
9.                  "\"payment_unit\":\"黑我
    \",\"dept_id\":\"158001\",\"region\":\"520000\"," +
10.                 "\"payment_total\":0.01,\"notify_url\":\"http://api.xxx.com/e
    pay/pay_notify\"," +
11.                 "\"items\":[{\"standard\":0.01,\"item_code\":\"0011801020\",\
    "bi_number\":1.0," +
12.                 "\"actual_amt\":0.01}],\"is_invalid\":\"0\"}");
13. //或者使用 setBizModel 传入业务参数，和 setData 不可同时使用
14. BusUnpayDataSyncModel model = new BusUnpayDataSyncModel();
15. model.setDeptId("158001");
16. model.setDocNumber("2019010400000011234000234");
17. // 省略其他参数
18. request.setBizModel(model);
19. BusUnpayDataSyncResponse response;
```

```
try {
//返回的数据已验签，解密
    response = ippClient.execute(request);
    log.info(JSON.toJSONString(response));
} catch (IppApiException e) {
    e.printStackTrace();
    log.error(e.getErrMsg());
}
```

4.2.1.2未使用缴费平台 sdk

如果未使用缴费平台 SDK，需要自行实现加密、签名过程。详情请参考加密与签名规则和调用完整流程

4.2.2 验签、解密示例（JAVA）

4.2.2.1使用缴费平台 sdk

缴费平台 sdk 已封装了验签、解密实现，通过 sdk 请求后会自动对返回的数据进行验签，解密,业务系统专注业务处理即可。

如果是异步通知，则调用 sdk 的 IppSignature.notifyRsaCheck(Map(String,String) params,String publicKey)方法进行验签、解密:

```
//异步回调的数据
Map<String, String> recMap = new HashMap<String, String>();
String ippPublicKey = "缴费平台公钥";
String appPublicKey = "业务系统私钥";
String enKey= "加密秘钥";
String signType = "签名方式";
String encryptType = "加密方式";
String data;
try {
    //验签并解密
    data = IppSignature.notifyRsaCheck(recMap, ippPublicKey, enKey, signType, encryptType);
} catch (IppApiException e) {
    e.printStackTrace();
}
//处理业务...
//封装返回给缴费平台的数据
String params = "返回给缴费平台的数据";
try {
    Map<String,String> map = IppSignature.notifyRsaSign(params,appPrivateKey, enKey, signType, encryptType);
    return map;
```

```
33. } catch (IppApiException e) {
34.     e.printStackTrace();
35. }
```

4.2.2.2未使用缴费平台 sdk

如果未使用缴费平台 SDK，需要自行实现验签过程。详情请参考加密与签名规则

附录 1：交易方式编码

编码	名称
01	POS 刷卡
02	柜台
03	微信
04	APP
05	现金
06	支付宝
07	银联

附录 2：数据格式

日期：固定使用 yyyy-MM-dd 格式（10 位）

日期时间：固定使用 yyyy-MM-dd HH-mm-ss 格式(19 位)

金额：有小数则保留小数，没有以 0 填充，最大保留 2 位小数，不带千位符。

例：0.01，100.00，999.99

附录 3：业务类别编码

编码	名称
1	教育业务
2	通用业务

附录 4：资金性质编码

编码	名称
1	非税

2	往来
3	义教
4	捐赠
5	民办
6	停车收费
7	其他
8	税务
9	伙食费