**中心服务文档v0.1**

（release日期2018年）

|  |  |  |  |
| --- | --- | --- | --- |
| 编 写 | jack wu | 编写 时间 | 2018-04-09 |
| 审 核 | 审批人（职务） | 审核 时间 | yyyy-mm-dd |
| 文档版本 | v0.1 | | |

**java110团队版权所有**

文档中的全部内容属java110团队所有，

未经允许，不可全部或部分发表、复制、使用于任何目的。

**文档修订摘要**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **日期** | **修订号** | **描述** | **著者** | **审阅者** | **日期** |
| 2018/04/09 | 0.1 | 文档初稿创建 |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

目录

**[二、 协议 2](#_Toc15801)**

[1、 协议结构 2](#_Toc21640)

[2、 订单信息 2](#_Toc25911)

[3、 业务信息 3](#_Toc14374)

[4、数据格式约定 4](#_Toc5592)

**[三、 表模型 4](#_Toc2290)**

[1、 c\_orders表模型 4](#_Toc7174)

[2、 c\_orders\_attrs表模型 5](#_Toc10964)

[3、 c\_business表模型 6](#_Toc16056)

[4、 c\_business\_attrs表模型 6](#_Toc14547)

[5、 c\_status表模型 7](#_Toc1312)

[6、 c\_order\_type表模型 7](#_Toc7311)

[7、 c\_business\_type表模型 7](#_Toc7337)

[8、 spec表模型 7](#_Toc22229)

[9、 c\_route表模型 7](#_Toc4056)

[10、 c\_query\_sql表模型 7](#_Toc10606)

# 系统流程图

1. 外围系统调用
2. 调用生成o\_id b\_id attr\_id等
3. 保存c\_orders和属性数据
4. 根据表c\_route路由表获取业务类型及信息保存c\_business数据及属性数据
5. 订单和业务项保存后先调规则服务，将报文传给规则进行校验，如果校验失败，直接作废订单和业务项，返回外围系统失败，如果成功继续后面流程
6. 根据在表c\_route 表中配置的同步还是异步方式，先处理同步和seq小的业务项，然后调用报文重置逻辑修改json报文信息（主要是根据获取到的同步业务项返回值，修改其他业务项需要修改的值，如报文中有绑定卡信息，同时也有支付信息，首先需要将绑定卡信息设置为同步，先执行，将返回的值修改支付信息中的账号，然后进行支付）
7. 异步的方式处理，则将消息发送到kafka平台，异步处理数据，等数据处理完成后，下游系统，发送kafaka通知消息，centerservice 服务需要接受并修改业务项状态，业务状态都完成了，修改订单状态，如果下游系统处理失败，同样通知centerservice 服务，centerservice 服务作废订单项和业务项，然后发送作废订单项kafka广播，下游系统作废数据（注这里作废成功与失败，不做处理，如果后期有问题，配置侦听失败的再次调用作废）。
8. 如果c\_route 表配置是查询的接口，则直接调用接口，不保存订单和业务项信息。

# 协议

## 协议结构

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **参数名称** | **约束** | **类型** | **长度** | **描述** | **取值说明** |
| orders | 1 | string |  | 订单信息 |  |
| business | ? | string |  | 业务信息 | 用于交互请求、应答的业务数据 |

## 订单信息

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **父节点** | **参数名称** | **约束** | **类型** | **长度** | **描述** | **取值说明** |
| - | orders | 1 | - | - | 订单信息 |  |
| tcpcont | appid | 1 | string | 10 | 外系统id，分配得到 | 有中心服务平台生成提供 |
| transactionid | 1 | string | 30 | 交易流水号 | 唯一的交易流水号。规则【appid+yyyymmddhh24miss +六位序列】100000000020180409224736000001 |
| userid | 1 | string | 12 | 用户id | 具体取值参考用户服务系统 |
| requesttime | 1 | string | 14 | 请求时间 | yyyymmddhh24misssss |
| remark | ? | string | 200 | 备注 |  |
| sign | 1 | string | 32 | 签名字符串 | transactionid+business节点内容+秘钥 md5签名，秘钥中心服务平台生成提供 |
| ordertypecd | 1 | string | 4 | 订单类型 | 参考ordertype模型 |
|  | attrs | ? | jsonarray | - | 订单属性 |  |
| attrs | speccd | 1 | string | 12 | 属性规格 | 参考spec模型 |
| attrs | value | 1 | string | 50 | 属性值 |  |

## 业务信息

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **父节点** | **参数名称** | **约束** | **类型** | **长度** | **描述** | **取值说明** |
| - | business | 1 | - | - | 订单信息 |  |
| tcpcont | servicecode | 1 | string | 20 | 服务编码 | 有中心服务平台提供（接口编码） |
| servicename | 1 | string | 50 | 服务名称 | 有中心服务平台提供（接口名） |
| remark | 1 | string | 200 | 备注 | 业务备注 |
| datas | 1 | jsonarray | 14 | 业务数据 | 业务数据具体查看相关文档，如客户服务文档 |
|  | attrs | ? | jsonarray | - | 业务属性 |  |
| attrs | speccd | 1 | string | 12 | 属性规格 | 参考spec模型 |
| attrs | value | 1 | string | 50 | 属性值 |  |

## 4、数据格式约定

|  |  |  |
| --- | --- | --- |
| 格式类型 | 格式符号 | 说明 |
| 约束 | ? | 0..1 |
| \* | 0..n |
| + | 1..n |
| 1 | 1 |
| 类型 | string,int…… |  |

特别说明：

* 对于jsonarray类型节点，只要节点存在，下面至少要有一个jsonobject类型的节点。
* 对于jsonarray类型节点，其约束只有两种情况：?、1。

# 表模型

## c\_orders表模型

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **字段** | **是否空** | **类型** | **长度** | **描述** | **取值说明** |
| **o\_id** | **否** | **varchar** | **18** | **订单id** | **系统自动生成**  **（10+**yyyymmdd+八位序列**）** |
| app\_id | 否 | **varchar** | 10 | 外系统id |  |
| ext\_transaction\_id | 否 | **varchar** | 30 | 交易流水号 | 对应于 transactionid |
| user\_id | 否 | **varchar** | 12 | 用户id |  |
| request\_time | 否 | Varchar | 16 | 请求时间 |  |
| create\_time | 否 | date |  | 创建时间 | 系统自动生成 |
| order\_type\_cd | 否 | **varchar** | 4 | 订单类型 | 参考c\_order\_type表 |
| finish\_time | 是 | date |  | 完成时间 |  |
| remark | 是 | **varchar** | 200 | 备注 |  |
| status\_cd | 否 | **varchar** | 2 | 数据状态 | 详细参考c\_status表 |

## c\_orders\_attrs表模型

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **字段** | **是否空** | **类型** | **长度** | **描述** | **取值说明** |
| **o\_id** | **否** | **varchar** | **18** | **订单id** | **关联c\_orders表o\_id** |
| attr\_id | 否 | **varchar** | 18 | 属性id | **系统自动生成**  **（11+**yyyymmdd+八位序列**）** |
| spec\_cd | 否 | **varchar** | 12 | 规格id | 参考spec表 |
| value | 否 | **varchar** | 50 | 属性值 |  |

## c\_business表模型

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **字段** | **是否空** | **类型** | **长度** | **描述** | **取值说明** |
| **b\_id** | **否** | **varchar** | **18** | **业务id** | **系统自动生成**  **（20+**yyyymmdd+八位序列**）** |
| o\_id | 否 | **varchar** | 10 | **订单id** | **关联c\_orders表o\_id** |
| create\_time | 否 | date |  | 创建时间 | 系统自动生成 |
| business\_type\_cd | 否 | **varchar** | 4 | 业务项类型 | 参考c\_business\_type表 |
| finish\_time | 是 | date |  | 完成时间 |  |
| remark | 是 | **varchar** | 200 | 备注 |  |
| status\_cd | 否 | **varchar** | 2 | 数据状态 | 详细参考c\_status表 |

## c\_business\_attrs表模型

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **字段** | **是否空** | **类型** | **长度** | **描述** | **取值说明** |
| **b\_id** | **否** | **varchar** | **18** | **订单id** | **关联c\_business表b\_id** |
| attr\_id | 否 | **varchar** | 18 | 属性id | **系统自动生成**  **（21+**yyyymmdd+八位序列**）** |
| spec\_cd | 否 | **varchar** | 12 | 规格id | 参考spec表 |
| value | 否 | **varchar** | 50 | 属性值 |  |

## c\_status字典表模型

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **字段** | **是否空** | **类型** | **长度** | **描述** | **取值说明** |
| **id** | **否** | **int** |  | **id** | **数据库自增** |
| status\_cd | 否 | **varchar** | 4 | 状态 |  |
| name | 否 | **varchar** | 50 | 名称 |  |
| description | 是 | **varchar** | 200 | 描述 |  |
| create\_time | 否 | date |  | 创建时间 |  |

## c\_order\_type字典表模型

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **字段** | **是否空** | **类型** | **长度** | **描述** | **取值说明** |
| **id** | **否** | **int** |  | **id** | **数据库自增** |
| order\_type\_cd | 否 | **varchar** | 4 | 订单类型 | 查询-q，受理-d，还是其他类型 |
| name | 否 | **varchar** | 50 | 名称 |  |
| description | 否 | **varchar** | 200 | 描述 |  |
| create\_time | 是 | date |  | 创建时间 |  |

## c\_business\_type字典表模型

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **字段** | **是否空** | **类型** | **长度** | **描述** | **取值说明** |
| **id** | **否** | **int** | **18** | **id** | **数据库自增** |
| business\_type\_cd | 否 | **varchar** | 4 | 业务项类型 | 对应servicecode |
| name | 否 | **varchar** | 50 | 名称 |  |
| description | 否 | **varchar** | 200 | 描述 |  |
| create\_time | 是 | date |  | 创建时间 |  |

## spec字典表模型

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **字段** | **是否空** | **类型** | **长度** | **描述** | **取值说明** |
| **id** | **否** | **int** | **18** | **id** | **数据库自增** |
| spec\_cd | 否 | **varchar** | 10 | 规格类型 | 规格编码，从x00020001开始每次加一就可以(约定，x=10表示c\_orders\_attrs 中属性，x=11表示c\_business\_attrs 中的属性) |
| name | 否 | **varchar** | 50 | 名称 |  |
| description | 否 | **varchar** | 200 | 描述 |  |
| create\_time | 是 | date |  | 创建时间 |  |

## c\_route表模型

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **字段** | **是否空** | **类型** | **长度** | **描述** | **取值说明** |
| **id** | **否** | **int** |  | **id** | **数据库自增** |
| app\_id | 否 | **varchar** | 10 | 外部应用id |  |
| service\_id | 否 | **Int** |  | 下游接口配置 | c\_invoke\_interface表中查看 |
| order\_type\_cd | 否 | **varchar** | 4 | 订单类型 | 查看c\_order\_type字典表 |
| invoke\_limit\_times | 是 | **int** |  | 接口调用一分钟调用次数 |  |
| create\_time | 是 | date |  | 创建时间 |  |
| status\_cd | 否 | **varchar** | 2 | 数据状态 | 详细参考c\_status表，0在用，1失效，2 表示下线（当组件调用服务超过限制时自动下线） |

说明：

1、接口需要判断，当前业务项是否全部在c\_route中配置了，如果存在没有配置，

全部都失败。

1. 路由匹配是用 app\_id service\_code order\_type\_cd字段
2. 接口 只能受理一种订单类型，business 中的数据要么全是查询，要么全是受理

## c\_service表模型

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **字段** | **是否空** | **类型** | **长度** | **描述** | **取值说明** |
| service\_id | **否** | **int** | **18** | **id** | **数据库自增** |
| service\_code | 否 | **varchar** | 50 | 服务编码 | 自定义，命名方式查询类query.+目标系统+.+业务名称  保存类 save.+目标系统+.+业务名称  修改类 modify.+目标系统+.+业务名称  删除类 remove.+目标系统+.+业务名称  例如：query.user.userinfo  save.user.adduserinfo |
| invoke\_model | 否 | **varchar** | 1 | 是否同步 | S-同步方式  A-异步方式 |
| business\_type\_cd | 否 | **varchar** | 4 | 对应业务项类型 | 参考c\_business\_type表，注意只有在c\_orders 表和c\_business 表中保存数据时，才有用 |
| name | 否 | **varchar** | 50 | 服务名称 |  |
| seq | 否 | **int** |  | 顺序 | 只有同步方式下根据seq从小到大调用接口 |
| messageQueueName | 是 | **Varchar** | 50 | 消息队里名称 | 只有异步时有用 |
| url | 否 | **varchar** | 200 | 目标地址 |  |
| method | 是 | **varchar** | 50 | 方法 | 空 为http post  其他为webservice方式调用 |
| timeout | 否 | **int** |  | 超时时间 |  |
| retry\_count | 否 | **int** |  | 重试次数 |  |
| provide\_app\_id | 否 | **varchar** | 10 | 提供方的 app\_id |  |
| create\_time | 是 | date |  | 创建时间 |  |
| status\_cd | 否 | **varchar** | 2 | 数据状态 | 详细参考c\_status表，0在用，1失效 |

## c\_mapping表模型

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **字段** | **是否空** | **类型** | **长度** | **描述** | **取值说明** |
| id | **否** | **int** |  | **id** | **数据库自增** |
| domain | **否** | **varchar** | **50** | **域** | **自定义** |
| name | 否 | **varchar** | 50 | 名称 |  |
| key | 否 | **varchar** | 100 | key |  |
| value | 否 | **varchar** | 100 | value |  |
| remark | 否 | **varchar** | 200 | 描述 |  |
| status\_cd | 否 | **varchar** | 1 | 数据状态 | 0表示在用，1表示失效 |
| create\_time | 是 | date |  | 创建时间 |  |

## c\_app表模型

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **字段** | **是否空** | **类型** | **长度** | **描述** | **取值说明** |
| id | **否** | **int** |  | **id** | **数据库自增** |
| app\_id | **否** | **varchar** | **10** | **分配应用id** | **系统分配** |
| name | 否 | **varchar** | 50 | 名称 | 对应系统名称 |
| [security](http://www.baidu.com/link?url=VkmEESlfoyBfOVHHjk4yZgUXdogR7c0PeUenbdic6Jj6lYnxEctAl1qCxhrWmgUXPWuTB0xNGt20mnir6ZuF1K" \t "https://www.baidu.com/_blank)\_code | 否 | **varchar** | 64 | 签名码 | sign签名时用 |
| while\_list\_ip | 是 | **varchar** | 200 | 白名单ip | 多个之间用;隔开 |
| black\_list\_ip | 是 | **varchar** | 200 | 黑名单ip | 多个之间用;隔开 |
| remark | 否 | **varchar** | 200 | 描述 |  |
| status\_cd | 否 | **varchar** | 1 | 数据状态 | 0表示在用，1表示失效 |
| create\_time | 是 | date |  | 创建时间 |  |

## c\_service\_sql表模型

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **字段** | **是否空** | **类型** | **长度** | **描述** | **取值说明** |
| id | **否** | **int** |  | **id** | **数据库自增** |
| service\_code | **否** | **varchar** | **50** | 服务编码 |  |
| name | 否 | **varchar** | 50 | 名称 | 服务名称 |
| params | 否 | **varchar** | 500 | 参数 | 参数之间用，分隔 |
| query\_model | 否 | **Varchar** | 1 | 查询方式 | 1、sql,2、存储过程，3、java脚本 |
| sql | 是 | **Longtext** |  | 执行sql |  |
| proc | 是 | **varchar** | 200 | 存储过程名称 |  |
| template | 否 | **longText** |  | 输出模板 |  |
| remark | 否 | **varchar** | 200 | 描述 |  |
| status\_cd | 否 | **varchar** | 1 | 数据状态 | 0表示在用，1表示失效 |
| create\_time | 是 | date |  | 创建时间 |  |

# 流程详细说明

1. 根据app\_id service\_code 和 order\_type\_cd 到redis（对应这个表c\_route）中查询报文中的服务是否存在，如果有一个不存在直接报错返回。
2. 调用规则服务，业务规则校验，如果失败直接返回(这里加入开关需不需要调用规则)。
3. 先处理同步的服务（seq 升序），调用每一个同步接口后，查看changejsonid是否为空，如果不为空，调服务编排处理（这里还没有想好，怎么搞，最好是写代码的方式搞，比较灵活，或者直接搞成一个服务也行）。
4. 如果是同步方式，order\_type\_cd字段字段判断是否需要创建订单和业务项数据。
5. 需要创建订单和业务项，则调用id生成服务生成主键（这里加入开关，如果开关是关闭状态则系统自己生成主键）。
6. 从redis中获取（下游系统信息，c\_invoke\_interface）。
7. 如果是同步方式，直接调用目标系统。
8. 如果是异步方式，直接发送kafka消息。
9. 如果是异步方式，kafka 监听修改状态消息，如果，下游处理失败，则作废订单和业务项数据，发送作废所有下游消息（这里传b\_id 和o\_id）。

# 外系统请求json协议模板

## 1、请求模板

{

"orders": {

"appId": "外系统id，分配得到",

"transactionId": "100000000020180409224736000001",

"userId": "用户Id",

"orderTypeCd": "订单类型,查询,受理",

"requestTime": "20180409224736",

"remark":"备注",

"sign": "这个服务是否要求md5签名",

"attrs": [{

"specCd": "配置的字段id",

"value": "具体值"

}]

},

"business": [{

"serviceCode": "querycustinfo",

"serviceName": "查询客户",

"remark": "备注",

"datas": [{

//这里是具体业务

}],

"attrs": [{

"specCd": "配置的字段id",

"value": "具体值"

}]

}]

}

## 2、返回模板

{  
 **"orders"**: {  
 **"transactionId"**: **"100000000020180409224736000001"**,  
 **"responseTime"**: **"20180409224736"**,  
 **"sign"**: **"这个服务是否要求md5签名"**,  
 **"response"**: {*//这个是centerorder 返回的状态结果* **"code"**: **"1999"**,  
 **"message"**: **"错误信息"** }  
 },  
 **"business"**:[{*//这个是相应的业务系统返回的结果，（受理为空，查询时不为空）* **"response"**: {  
 **"code"**: **"1999"**,  
 **"message"**: **"错误消息"** }  
 *//相应内容* }]  
}

# 系统调用方式



# 请求下游系统json协议模板

## 1、请求模板

{  
 **"orders"**: {  
 **"transactionId"**: **"100000000020180409224736000001"**,  
 **"requestTime"**: **"20180409224736"**,

**"orderTypeCd"**:**"订单类型,查询,受理"**,

**"dataFlowId"**:**"20020180000001"**,  
 **"businessType"**:**"Q"** },  
 **"business"**: [{  
 **"bId"**:**"12345678"**,  
 **"serviceCode"**: **"querycustinfo"**,  
 **"serviceName"**: **"查询客户"**,  
 **"remark"**: **"备注"**,  
 **"datas"**: [{  
 *//这里是具体业务* }],  
 **"attrs"**: [{  
 **"specCd"**: **"配置的字段id"**,  
 **"value"**: **"具体值"** }]  
 }]  
}

**businessType Q 表示 请求报文 N通知报文**

## 2、返回或通知模板

{  
 **"orders"**: {  
 **"transactionId"**: **"100000000020180409224736000001"**,  
 **"responseTime"**: **"20180409224736"**,

**"orderTypeCd"**:**"订单类型,查询,受理"**,

**"dataFlowId"**:**"20020180000001"**,  
 **"businessType"**:**"N"** },  
 **"business"**:[{*//这个是相应的业务系统返回的结果，（受理为空，查询时不为空）* **"response"**: {  
 **"code"**: **"1999"**,  
 **"message"**: **"错误消息"** },  
 **"bId"**:**"12345678"**,  
 **"serviceCode"**: **"querycustinfo"** *//其他字段* }]  
}

注意：code 为0000 表示成功

## 3、竣工模板

{  
 **"orders"**: {  
 **"transactionId"**: **"100000000020180409224736000001"**,  
 **"requestTime"**: **"20180409224736"**,

**"orderTypeCd"**:**"订单类型,查询,受理"**,

**"dataFlowId"**:**"20020180000001"**,  
 **"businessType"**:**"N"** },  
 **"business"**:[{*//这个是相应的业务系统返回的结果，（受理为空，查询时不为空）* **"response"**: {  
 **"code"**: **"1999"**,  
 **"message"**: **"错误消息"** },  
 **"bId"**:**"12345678"**,  
 **"serviceCode"**: **"querycustinfo"** *//其他字段* }]  
}

注意：code 为0000 表示成功

# 签名与加密说明

## 请求sign说明

外系统请求centerService 服务时，sign 的生成：

inStr = transactionId + appId+business（内容）+security\_code(系统分配) DigestUtils.*md5Hex*(inStr.getBytes(**"UTF-8"**));

## 返回sign说明

CenterService服务返回时 sign生成：

inStr = transactionId + responseTime+business（内容）+security\_code(系统分配)

DigestUtils.*md5Hex*(inStr.getBytes(**"UTF-8"**));

注意：当传入AppId 不正确，或者请求报文解密失败的情况下，返回sign 不做加密处理，其值为空

## 请求报文加密

如果http post 请求时header 中有 ENCRYPT 并且值为ON，时启用密文传输方式，即请求报文和返回报文都为密文，如果没有 ENCRYPT 或 值不为ON 则明文传输。

加密代码参考：

*/\*\*  
 \* 加密  
 \** ***@param data*** *\** ***@param publicKey*** *\** ***@param keySize*** *\** ***@return*** *\** ***@throws*** *Exception  
 \*/***public static byte**[] encrypt(**byte**[] data, PublicKey publicKey, **int** keySize)  
 **throws** Exception  
{  
 Cipher cipher = Cipher.*getInstance*(**"RSA/ECB/PKCS1PADDING"**, **"BC"**);  
 cipher.init(Cipher.***ENCRYPT\_MODE***, publicKey);  
  
 **int** blockSize = (keySize >> 3) - 11;  
  
 **int** inputLen = data.**length**;  
 ByteArrayOutputStream out = **new** ByteArrayOutputStream();  
 **int** offSet = 0;  
 **int** i = 0;  
 **while** (inputLen - offSet > 0) {  
 **byte**[] buf;  
 **if** (inputLen - offSet > blockSize) {  
 buf = cipher.doFinal(data, offSet, blockSize);  
 }**else** {  
 buf = cipher.doFinal(data, offSet, inputLen - offSet);  
 }  
 out.write(buf, 0, buf.**length**);  
 ++i;  
 offSet = i \* blockSize;  
 }  
 **byte**[] result = out.toByteArray();  
  
 **return** result;  
}

*/\*\*  
 \* 加载公钥  
 \** ***@param keyData*** *\** ***@return*** *\** ***@throws*** *Exception  
 \*/***public static** PublicKey loadPubKey(String keyData)  
 **throws** Exception  
{  
 **return** *loadPemPublicKey*(keyData, **"RSA"**);  
}

*/\*\*  
 \* 加载公钥  
 \** ***@param privateKeyPem*** *\** ***@param algorithm*** *\** ***@return*** *\** ***@throws*** *Exception  
 \*/***public static** PrivateKey loadPrivateKeyPkcs8(String privateKeyPem, String algorithm)  
 **throws** Exception  
{  
 String privateKeyData = privateKeyPem.replace(**"-----BEGIN PRIVATE KEY-----"**, **""**);  
 privateKeyData = privateKeyData.replace(**"-----END PRIVATE KEY-----"**, **""**);  
 privateKeyData = privateKeyData.replace(**"\n"**, **""**);  
 privateKeyData = privateKeyData.replace(**"\r"**, **""**);  
  
 **byte**[] decoded = Base64.*getDecoder*().decode(privateKeyData.getBytes());  
  
 PKCS8EncodedKeySpec pkcs8KeySpec = **new** PKCS8EncodedKeySpec(decoded);  
 KeyFactory keyFactory = KeyFactory.*getInstance*(algorithm);  
  
 **return** keyFactory.generatePrivate(pkcs8KeySpec);  
}

//加密

reqJson = **new** String(*encrypt*(resJson.getBytes(**"UTF-8"**), *loadPubKey*(“公钥”)  
 , 2048)),**"UTF-8"**);

其中 keySize 如果设置要重新设置则 http post header 中传ENCRYPT\_KEY\_SIZE来设置 不传则去默认值，默认值配置在映射表中，key为 KEY\_DEFAULT\_DECRYPT\_KEY\_SIZE

## 返回报文解密

如果http post 请求时header 中有 ENCRYPT 并且值为ON，时启用密文传输方式。

解密代码参考

*/\*\*  
 \* 解密  
 \** ***@param data*** *\** ***@param privateKey*** *\** ***@param keySize*** *\** ***@return*** *\** ***@throws*** *Exception  
 \*/***public static byte**[] decrypt(**byte**[] data, PrivateKey privateKey, **int** keySize)  
 **throws** Exception  
{  
 Cipher cipher = Cipher.*getInstance*(**"RSA/ECB/PKCS1PADDING"**, **"BC"**);  
 cipher.init(Cipher.***DECRYPT\_MODE***, privateKey);  
 **int** blockSize = keySize >> 3;  
  
 ByteArrayInputStream byteArrayInputStream = **new** ByteArrayInputStream(data);  
 ByteArrayOutputStream byteArrayOutputStream = **new** ByteArrayOutputStream();  
  
 **byte**[] buf = **new byte**[blockSize];  
 **int** len = 0;  
 **while** ((len = byteArrayInputStream.read(buf)) > 0) {  
 byteArrayOutputStream.write(cipher.doFinal(buf, 0, len));  
 }  
  
 **return** byteArrayOutputStream.toByteArray();  
}

*/\*\*  
 \* 加载私钥  
  
 \** ***@param keyData*** *\** ***@return*** *\** ***@throws*** *Exception  
 \*/***public static** PrivateKey loadPrivateKey(String keyData) **throws** Exception {  
 **return** *loadPrivateKeyPkcs8*(keyData, **"RSA"**);  
}

*/\*\*  
 \* 加载私钥  
 \** ***@param privateKeyPem*** *\** ***@param algorithm*** *\** ***@return*** *\** ***@throws*** *Exception  
 \*/***public static** PrivateKey loadPrivateKeyPkcs8(String privateKeyPem, String algorithm)  
 **throws** Exception  
{  
 String privateKeyData = privateKeyPem.replace(**"-----BEGIN PRIVATE KEY-----"**, **""**);  
 privateKeyData = privateKeyData.replace(**"-----END PRIVATE KEY-----"**, **""**);  
 privateKeyData = privateKeyData.replace(**"\n"**, **""**);  
 privateKeyData = privateKeyData.replace(**"\r"**, **""**);  
  
 **byte**[] decoded = Base64.*getDecoder*().decode(privateKeyData.getBytes());  
  
 PKCS8EncodedKeySpec pkcs8KeySpec = **new** PKCS8EncodedKeySpec(decoded);  
 KeyFactory keyFactory = KeyFactory.*getInstance*(algorithm);  
  
 **return** keyFactory.generatePrivate(pkcs8KeySpec);  
}

//解密

resJson = **new** String(*decrypt*(reqJson.getBytes(**"UTF-8"**), “私钥”  
 , 2048)),**"UTF-8"**);

说明：加密和解密的公钥和私钥，由centerService提供。

# 九、状态码说明

|  |  |
| --- | --- |
| **状态编码** | **描述** |
| **0000** | **成功** |
| **1999** | **未知失败** |
| **1998** | **系统内部错误** |
| **1997** | **调用下游系统超时** |

{

"JSON\_PARAMS": [

{

"param1": "#ProdInst#Object"

},

{

"param2": "ProdInst#FuncProdInst#Array"

},

{

"param3": "ProdInst.FuncProdInst#AttrInfo#Array"

}

]

}

{

"param1": "

select op.prod\_id ProdInstId,

op.redu\_prod\_spec\_id ProductNbr,

ps.name ProductName,

p.party\_id CustId,

p.Name CustName

from inst.offer\_prod op, crm.area a, prod.prod\_spec ps, crm.party p

where op.redu\_access\_number = #AccNbr#

and op.area\_id = a.AREA\_ID

and (a.Zone\_Number = #RegionCode# or #RegionCode# is null)

and (op.redu\_prod\_spec\_id = #ProductNbr# or #ProductNbr# is null)

and op.redu\_prod\_spec\_id = ps.prod\_spec\_id

and op.redu\_owner\_id = p.party\_id

",

"param2": "

select os.serv\_id FunProdInstId, ss.name FunProdName, ss.serv\_spec\_id FunProdNbr

from inst.offer\_serv os, prod.serv\_spec ss

where os.prod\_id = #SAOP\_PARENT\_ProdInstId#

and os.status\_cd in (10, 11, 12, 13)

and os.serv\_spec\_id = ss.serv\_spec\_id

",

"param3": "

select osi.PROD\_INST\_ATTR\_ID AttrId,

osi.item\_spec\_id AttrNbr,

osi.value AttrValue,

iss.name AttrName

from inst.offer\_serv\_item osi, bss\_sys.item\_spec iss

where osi.serv\_id = #SAOP\_PARENT\_FunProdInstId#

and osi.item\_spec\_id = iss.item\_spec\_id

and osi.status\_cd in (10, 11, 12, 13)

"

}