文档编号	V1.1
文档负责人	
文档名称	审批流设计

审批流回调接口设计

文档历史

修订日期	修订内容	修订版本	修订人
2020-08-20		V1.1	张正义

第 1 章　系统概述

1.1　业务背景

参见审批流需求 v1.1

1.2　部署架构图

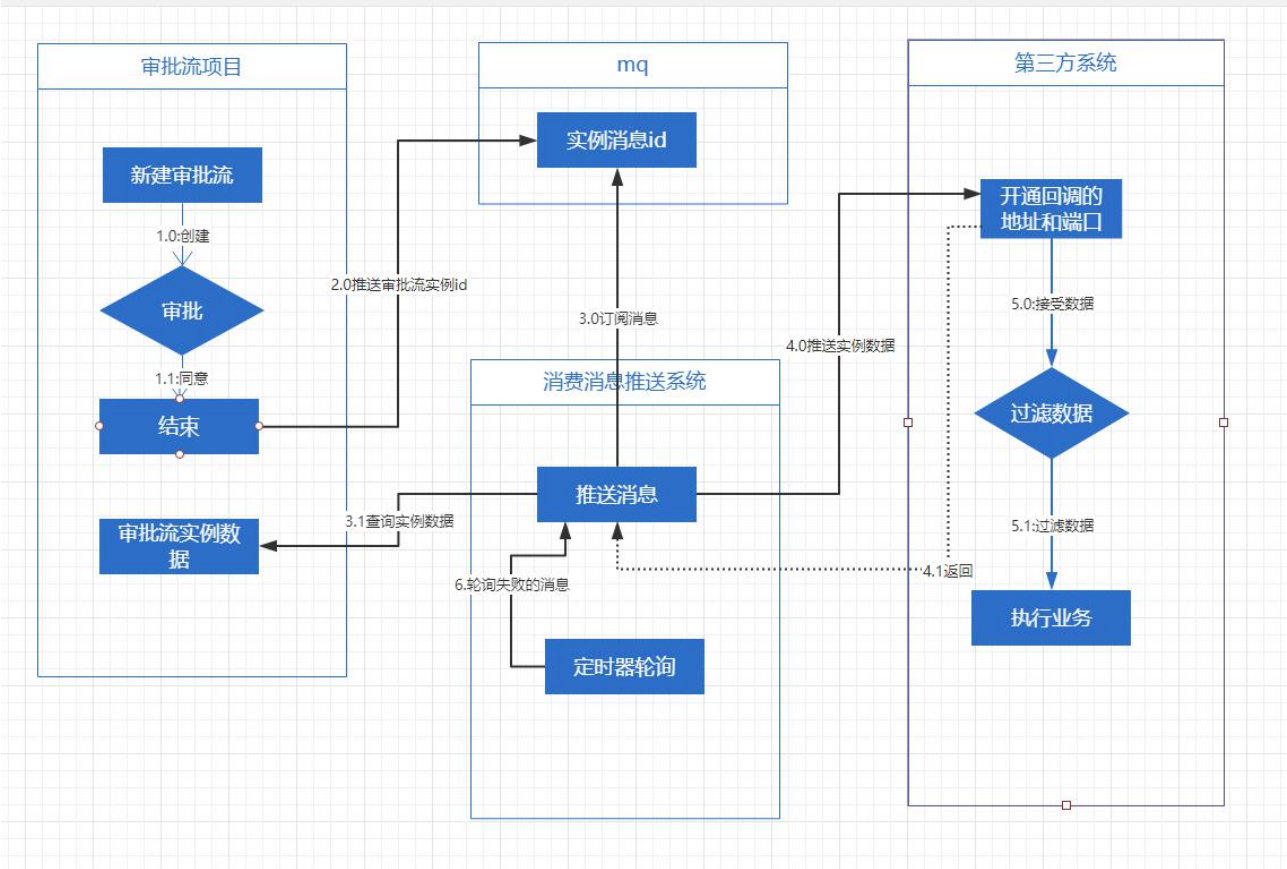
结合实际的，需要了解到实际线上的部署架构，同时需要了解到模块间的调用关系，这样在升级某个模块时才能评估到对其他模块的影响

无

第 2 章　业务实现

2.1 流程实例业务

2.1.1 回调流程



第 3 章　外部接口设计

3.1 回调接口

说明:提供 java 实例代码

3.1.1 请求参数

请求参数说明	名称	长度	示例
消息 id	messageId	C(0,32)	1293110195810992128
请求时间	time	N(0,32)	毫秒
内容	body	C(0,4000)	传送的内容(DES 加密)

Body（解密后 json 格式如下）：

请求参数说明	名称	长度	示例
组织 id	organizationId	C(0,32)	1293110195810992128
组织 name	organizationName	C(0,200)	
审批结果	approvalResult	N(2)	1-待审批、2-已通过、3-已拒绝
申请人 id	applyId	C(0,32)	4314886048649199622
申请人	applyName	C(0,200)	张三
申请时间	approvalStartTime	C(0,32)	2020-08-25 14:18:20
审批完成时间	approvalEndTime	C(0,32)	2020-08-25 14:18:20
业务数据	buzyData	C(0,4000)	Json 数据的字符串
审批流实例 id	approvalFlowId	C(0,32)	1295277199103885312
审批流模板 id	modelId	C(0,32)	1281468315591507968

请求示例:

{

“messageId”:” ”,

“time”:” ”,

“body”:“DES 解密后字符串”,

}

“body”解密后:{

“organizationId”:” ”,

“organizationName”:” ”,

“approvalResult”:” ”,

“applyId”:” ”,

“applyName”:” ”,

“approvalStartTime”:” ”,

“approvalEndTime”:” ”,

“approvalFlowId”:” ”,

“modelId”:” ”,

“buzyData”:” {"booktcnt":{"identification":"booktcnt","dataValue":"3"},"bookname":{"identification":"bookname","dataValue":"java 编程思想"}}”,

}

3.1.2 返回数据

返回参数说明	名称	长度	示例
返回信息	utMsg	C(0,100)	失败具体信息
返回码	utCode	N(2)	0-成功 非 0 失败

备注:如果系统返回码不是 0 成功 回调接口会一直反复调用 所以接口方要注意接口幂等规则

返回示例:

{

“utMsg”:” ok”

“utCode”:0

}

3.1.3 异常定义

返回码	名称	示例
0	成功码	
1	失败码	解析数据错误...

3.1.4 加解密工具

数字签名

为了保证数据传输过程中的数据真实性和完整性，我们需要对数据进行数字加密

加解密工具。

DESUtil.java

```
public class DESUtil {

    public static void main(String[] args) {
        String source  = "测试 des 加密";
        String key = "fshdhfhds343";
        String result = encrypt(source, key);
        //加密结果
        System.out.println(result);
        //解密
        System.out.println(decrypt(result, key));
    }

    /**
     * DES 加密操作
     * @param source 要加密的源
     * @param key    约定的密钥
     * @return
     */
    public static String encrypt(String source,String key){
        //强加密随机数生成器
        SecureRandom random = new SecureRandom();
        try {
            //创建密钥规则
            DESKeySpec keySpec = new DESKeySpec(key.getBytes());
            //创建密钥工厂
            SecretKeyFactory keyFactory = SecretKeyFactory.getInstance("DES");
```

```
        //按照密钥规则生成密钥
        SecretKey secretKey =  keyFactory.generateSecret(keySpec);
        //加密对象
        Cipher cipher = Cipher.getInstance("DES");
        //初始化加密对象需要的属性
        cipher.init(Cipher.ENCRYPT_MODE, secretKey, random);
        //开始加密
        byte[] result = cipher.doFinal(source.getBytes());
        //Base64 加密
        return  new BASE64Encoder().encode(result) ;
    } catch (Exception e) {
        e.printStackTrace();
    }
    return null;
}
/**
 * 解密
 * @param cryptograph 密文
 * @param key         约定的密钥
 * @return
 */
public static String decrypt(String cryptograph,String key){
    //强加密随机生成器
    SecureRandom random  =  new SecureRandom();
    try {
        //定义私钥规则
        DESKeySpec keySpec = new DESKeySpec(key.getBytes());
        //定义密钥工厂
        SecretKeyFactory factory = SecretKeyFactory.getInstance("DES");
        //按照密钥规则生成密钥
        SecretKey secretkey = factory.generateSecret(keySpec);
        //创建加密对象
        Cipher cipher = Cipher.getInstance("DES");
        cipher.init(Cipher.DECRYPT_MODE, secretkey, random);
        //Base64 对
        byte[] result = new BASE64Decoder().decodeBuffer(cryptograph);
```

```
            return new String(cipher.doFinal(result));
        } catch (Exception e) {
            e.printStackTrace();
        }
        return null;
    }
```

3.1.5 测试接口

```
@RestController
@RequestMapping("/api")
public class ApiTestController {


    /**
     * 审批流接口接收数据

     *messageId 需要保证幂等去重 防止重复调用消费
     */
    @PostMapping("/test")
    public ResponseInfo test(HttpServletRequest request)  {

        System.out.println(request);

        System.out.println("获取参数");

        System.out.println("------messageId-------"
+request.getParameter("messageId"));
        System.out.println("------time-------" +request.getParameter("time"));
        System.out.println("-------body------" +request.getParameter("body"));

        //解密 secret 密钥

        String secret = "fshdhfhds343";

        System.out.println("------解密-body------" +

DESUtil.decrypt(request.getParameter("body"),secret));

        //返回值

        return ResponseInfo.convertOK();
    }

}
```