globalMessages_zh_CN.properties

```
https://192.0.0.241/APP-Financial/branches/base_develop/iVMS-8000-CVTMS/Dev_v1.0.3/web/finance-cvtms-web/src/main/resources/langua

login.usernameorpassworderror=用户名或密码错误
login.userroleerror=用户未绑定角色
login.validate.code=验证码输入错误
login.pwd.default=密码为初始密码，请修改密码
login.login.first=首次登录平台，请修改密码
```

pom.xml

```
https://192.0.0.241/APP-Financial/branches/base_develop/iVMS-8000-CVTMS/Dev_v1.0.3/web/finance-cvtms-web/pom.xml

<!-- fastjson -->
<dependency>
    <groupId>com.alibaba</groupId>
    <artifactId>fastjson</artifactId>
    <version>1.2.28</version>
</dependency>

<!-- kaptcha -->
<dependency>
    <groupId>com.google.code.kaptcha</groupId>
    <artifactId>kaptcha</artifactId>
    <version>2.3.2</version>
    <classifier>jdk16</classifier>
</dependency>

<!-- Struts2 jar 升级 -->
<dependency>
    <groupId>commons-fileupload</groupId>
    <artifactId>commons-fileupload</artifactId>
    <version>1.3.1</version>
</dependency>

<dependency>
    <groupId>org.freemarker</groupId>
    <artifactId>freemarker</artifactId>
    <version>2.3.22</version>
</dependency>

<dependency>
    <groupId>ognl</groupId>
    <artifactId>ognl</artifactId>
    <version>3.1.12</version>
</dependency>

<dependency>
    <groupId>org.apache.struts</groupId>
    <artifactId>struts2-core</artifactId>
    <version>2.3.32</version>
</dependency>

<dependency>
    <groupId>org.apache.struts</groupId>
    <artifactId>struts2-json-plugin</artifactId>
    <version>2.3.31</version>
</dependency>

<dependency>
    <groupId>org.apache.struts</groupId>
    <artifactId>struts2-spring-plugin</artifactId>
    <version>2.3.32</version>
```

```
</dependency>

<dependency>
    <groupId>org.apache.struts.xwork</groupId>
    <artifactId>xwork-core</artifactId>
    <version>2.3.32</version>
</dependency>

<!-- 漏洞扫描 jar包升级 -->
<dependency>
    <groupId>commons-collections</groupId>
    <artifactId>commons-collections</artifactId>
    <version>3.2.2</version>
</dependency>
<!-- 漏洞扫描 jar包升级  END-->
```

swdf-task.xml：新增两个定时任务

```
https://192.0.0.241/APP-Financial/branches/base_develop/iVMS-8000-CVTMS/Dev_v1.0.3/web/finance-cvtms-web/src/main/resources/spring

<!-- userlogincheck定期清除任务  周期：天 -->
<bean id="userCheckTask" class="com.hikvision.finance.cvtms.system.task.UserCheckTask" autowire="byName">
    <property name="schedulingPattern" value="* 0 1 * * *"></property>
</bean>

<!-- 每30分钟执行一次 -->
<bean id="memoryTimeTask" class="com.hikvision.finance.cvtms.common.task.MemoryTimeTask" autowire="byName">
    <property name="schedulingPattern" value="0 */30 * * * *" />
</bean>
```

spring-hibernate.xml

```
https://192.0.0.241/APP-Financial/branches/base_develop/iVMS-8000-CVTMS/Dev_v1.0.3/web/finance-cvtms-web/src/main/resources/spring

<!-- 配置事务的传播特性 -->
<tx:advice id="txAdvice" transaction-manager="hibernateTxManager">
    <tx:attributes>
        //... 保持不变 新增validateLoginUser
        <tx:method name="validateLoginUser" propagation="REQUIRED" rollback-for="Exception"/>
        <tx:method name="*" read-only="true" />
    </tx:attributes>
</tx:advice>
```

spring-config-bean.xml：增加三个bean的配置

```
https://192.0.0.241/APP-Financial/branches/base_develop/iVMS-8000-CVTMS/Dev_v1.0.3/web/finance-cvtms-web/src/main/resources/spring

<bean id="userLoginCheckDao" class="com.hikvision.finance.cvtms.modules.users.dao.impl.UserLoginCheckDaoImpl" autowire="byName"></
<bean id="userIdAndSessionDao" class="com.hikvision.finance.cvtms.modules.users.dao.impl.UserIdAndSessionDaoImpl" autowire="byName
<bean id="userIdAndSessionService" class="com.hikvision.finance.cvtms.modules.users.service.impl.UserIdAndSessionServiceImpl" auto
```

添加工具类

```
https://192.0.0.241/APP-Financial/branches/base_develop/iVMS-8000-CVTMS/Dev_v1.0.3/web/finance-cvtms-web/src/main/java/com/hikvisi

ValidateUtil.java
Base64.java
PasswordEncryptUtil.java
XSSTranslator.java
```

修改Constants.SysConfigType

```
https://192.0.0.241/APP-Financial/branches/base_develop/iVMS-8000-CVTMS/Dev_v1.0.3/web/finance-cvtms-web/src/main/java/com/hikvisi

//ADD 2017-02-22
public static final int USER_AND_IP_LOCK                    = 7001;//用户和IP锁定功能
public static final int FAIL_TIMES                          = 7002;//连续失败次数
public static final int LOCK_MINUTES                        = 7003;//用户和IP锁定时长
```

增加类

```
https://192.0.0.241/APP-Financial/branches/base_develop/iVMS-8000-CVTMS/Dev_v1.0.3/web/finance-cvtms-web/src/main/java/com/hikvisi

https://192.0.0.241/APP-Financial/branches/base_develop/iVMS-8000-CVTMS/Dev_v1.0.3/web/finance-cvtms-web/src/main/java/com/hikvisi

https://192.0.0.241/APP-Financial/branches/base_develop/iVMS-8000-CVTMS/Dev_v1.0.3/web/finance-cvtms-web/src/main/java/com/hikvisi

https://192.0.0.241/APP-Financial/branches/base_develop/iVMS-8000-CVTMS/Dev_v1.0.3/web/finance-cvtms-web/src/main/java/com/hikvisi

https://192.0.0.241/APP-Financial/branches/base_develop/iVMS-8000-CVTMS/Dev_v1.0.3/web/finance-cvtms-web/src/main/java/com/hikvisi

https://192.0.0.241/APP-Financial/branches/base_develop/iVMS-8000-CVTMS/Dev_v1.0.3/web/finance-cvtms-web/src/main/java/com/hikvisi

https://192.0.0.241/APP-Financial/branches/base_develop/iVMS-8000-CVTMS/Dev_v1.0.3/web/finance-cvtms-web/src/main/java/com/hikvisi

https://192.0.0.241/APP-Financial/branches/base_develop/iVMS-8000-CVTMS/Dev_v1.0.3/web/finance-cvtms-web/src/main/java/com/hikvisi

https://192.0.0.241/APP-Financial/branches/base_develop/iVMS-8000-CVTMS/Dev_v1.0.3/web/finance-cvtms-web/src/main/java/com/hikvisi

https://192.0.0.241/APP-Financial/branches/base_develop/iVMS-8000-CVTMS/Dev_v1.0.3/web/finance-cvtms-web/src/main/java/com/hikvisi

https://192.0.0.241/APP-Financial/branches/base_develop/iVMS-8000-CVTMS/Dev_v1.0.3/web/finance-cvtms-web/src/main/java/com/hikvisi

https://192.0.0.241/APP-Financial/branches/base_develop/iVMS-8000-CVTMS/Dev_v1.0.3/web/finance-cvtms-web/src/main/java/com/hikvisi
```

LoginAction

```
https://192.0.0.241/APP-Financial/branches/base_develop/iVMS-8000-CVTMS/Dev_v1.0.3/web/finance-cvtms-web/src/main/java/com/hikvisi

public class LoginAction extends BaseAction<Users> {
    //... 不变
    private String passwordSecurityLevel;

    private String randomCode;
    private String ip;
    private IUserIdAndSessionService userIdAndSessionService;

    /**
     * Web登录方法
     * @author chenliujie 2015年7月14日 下午10:40:00
     * @return
     */
    public String login() {
        String method = ServletActionContext.getRequest().getMethod();
        if (!"POST".equals(method)) {// AppScan扫描出“查询中接受的主体参数”缺陷，需要禁用GET方法
            throw new ExpectedException("", "请求方法错误");
        }

        username = entity.getStrName();
        String result = userLoginCheck(ip);
        if(null != result ){
            return result;
        }

        ajaxData = loginService.validateLoginUser(entity, ip);
        if(ajaxData.getSuccess()) { // 登录成功，设置session
```

```
        //... 不变
    }
    ajaxData.getData().remove("user");
    ajaxData.getData().remove("role");
    ajaxData.getData().remove("menus");
    ajaxData.getData().remove("menuscs");
    return AJAX;
}

private String userLoginCheck(String ip){
    if(!"yzm".equals(randomCode) ){//只进行验证码验证
        if (ValidateUtil.isValidateCode(getRequest(), randomCode)) {
            ajaxData.setAutoSet(false);
            ajaxData.setFailureMsg(GlobalMessageUtil.getMessage("login.validate.code"));
            return AJAX;
        }
    }
    UserLoginCheck userLoginCheck = loginService.getUserLoginCheck(username, ip);
    if(userLoginCheck != null){
        Integer times = userLoginCheck.getTimes();//失败次数
        Date updateTime = userLoginCheck.getUpdateTime();//最后登录失败的时间

        SysconfigDto sysconfigDto = sysconfigService.getSysconfig();
        Integer maxFailNum = sysconfigDto.getMaxFailNum();//最大连续失败次数
        Integer lockMinutes = sysconfigDto.getLockMinutes();//锁定时间
        long minusTime = System.currentTimeMillis() - updateTime.getTime();//毫秒数  当前时间和最后失败时间的时间差
        if("1".equals(String.valueOf(sysconfigDto.getEnableLock())) && times >= maxFailNum){//判断用户或IP是否锁定，是否开启锁定功能
            if(minusTime < 1000L*60*lockMinutes){
                long leftTime = 1000L*60*lockMinutes - minusTime;//还有多久解锁

                long seconds = leftTime / 1000 ;
                if(leftTime % 1000 != 0){
                    seconds += 1;
                }
                long leftMinutes = seconds / 60;
                long leftSeconds = seconds % 60;
                ajaxData.setAutoSet(false);
                if(leftMinutes == 0 && leftSeconds != 0){
                    ajaxData.setFailureMsg("账户已被锁定，请"+leftSeconds+"秒后再试");
                }else if(leftMinutes != 0 && leftSeconds == 0){
                    ajaxData.setFailureMsg("账户已被锁定，请"+leftMinutes+"分钟后再试");
                }else if(leftMinutes != 0 && leftSeconds != 0){
                    ajaxData.setFailureMsg("账户已被锁定，请"+leftMinutes+"分钟"+leftSeconds+"秒后再试");
                }
                return AJAX;
            }else{
                loginService.unlockUser(username, ip);
            }

        }
    }
    return null;
}

/**
 * 修改密码
 * @author fuqunqing 2015年8月17日 下午7:52:16
 * @return
 */
public String changePassword() {
    //... 不变
    if(ajaxData.getSuccess()) {
        OperationLogUtils.setContent(GlobalMessageUtil.getMessage("log.others.password.change"));
        sessionProcessor.removeSession(getResponse(), getRequest());
    }
    return AJAX;
}
```

```
    /**
     * 构建session信息
     * @author fuqunqing 2015年8月28日 上午10:55:23
     * @param ajaxData
     * @return
     */
    @SuppressWarnings("unchecked")
    private HikWebSession buildSession(AjaxData ajaxData) {
        HikWebSession hikSession = new HikWebSession();
        Users user = (Users)ajaxData.get("user");
        UserSession userSession = (UserSession)ajaxData.get(SessionAttrbutes.USER_SESSION);
        List<MenuBean> menus = (List<MenuBean>)ajaxData.get("menus");
        List<MenuBean> menuscs = (List<MenuBean>)ajaxData.get("menuscs");

        String sessionId = UUIDUtils.getUUID();
        hikSession.setAttribute(HikWebSession.KEY_SESSION_ID, sessionId);
        hikSession.setAttribute(HikWebSession.KEY_SESSION_USER, user.getId().toString());
        hikSession.setAttribute(HikWebSession.KEY_SESSION_NAME, user.getStrName());
        hikSession.setAttribute(SessionAttrbutes.MENUS, menus);
        hikSession.setAttribute(SessionAttrbutes.MENUS_CS, menuscs);
        hikSession.setAttribute(SessionAttrbutes.USER_SESSION, userSession);

        UserIdAndSessionQo userIdAndSessionQo = new UserIdAndSessionQo();
        userIdAndSessionQo.setUserId(user.getId());
        userIdAndSessionQo.setClientType(1);
        UserIdAndSession userIdAndSession = userIdAndSessionService.queryUnique(userIdAndSessionQo);
        if(null == userIdAndSession){
            userIdAndSession = new UserIdAndSession();
            userIdAndSession.setUserId(user.getId());
            userIdAndSession.setClientType(1);
            userIdAndSession.setSessionId(sessionId);
            userIdAndSessionService.save(userIdAndSession);
        }else{
            userIdAndSession.setSessionId(sessionId);
            userIdAndSessionService.update(userIdAndSession);
        }

        // 该IP是通过前端页面获取的UMCC服务器访问IP地址，用于定位网域
        String ip = StringUtils.defaultIfBlank(getRequest().getParameter("ip"), "").trim();
        if(StringUtils.indexOf(ip, ":") > -1) {
            ip = ip.substring(0, ip.indexOf(":"));
        }
        hikSession.setAttribute(HikWebSession.KEY_SESSION_IP, ip);
        return hikSession;
    }
}

private boolean setSession(HikWebSession hikSession) {
    String expire = ConfigManager.getConfiguration("web-session", "session.expire");
    if(StringUtils.isEmpty(expire)) {
        expire = "20";
    }
    //CookieUtil.setCookie(getResponse(), "HIK_COOKIE_STYLE", StringUtils.defaultIfBlank("", "default"), 30 * 24 * 60 * 60, "/", n
    boolean result = sessionProcessor.setSession(getRequest(), getResponse(), hikSession, Integer.parseInt(expire.trim()) * 60 * 1
    return result;
}
```

ILoginService接口增加方法和修改方法

```
https://192.0.0.241/APP-Financial/branches/base_develop/iVMS-8000-CVTMS/Dev_v1.0.3/web/finance-cvtms-web/src/main/java/com/hikvisi

public interface ILoginService {

    /**
     * 通过用户名和IP去获取该用户的登录校验信息
```

```
     * @author wangxinglei 2017年2月22日 下午2:08:58
     * @param username
     * @return
     */
    UserLoginCheck getUserLoginCheck(String username, String ip);

    /**
     * 解锁用户
     * @author wangxinglei 2017年3月16日 上午11:20:16
     * @param username
     * @param ip
     */
    void unlockUser(String username, String ip);

    /**
     * 验证登录用户
     * @author chenliujie 2015年7月15日 下午6:29:31
     * @param entity
     * @return
     */
    AjaxData validateLoginUser(Users entity, String ip);

}
```

LoginServiceImpl

```
https://192.0.0.241/APP-Financial/branches/base_develop/iVMS-8000-CVTMS/Dev_v1.0.3/web/finance-cvtms-web/src/main/java/com/hikvisi

private IUserLoginCheckDao userLoginCheckDao;

public UserLoginCheck getUserLoginCheck(String username, String ip) {
    UserLoginCheckQo qo = new UserLoginCheckQo();
    qo.setUsername(username);
    qo.setUserIp(ip);
    UserLoginCheck userLoginCheck = userLoginCheckDao.queryUnique(qo);
    return userLoginCheck;
}
public void unlockUser(String username, String ip) {
    userLoginCheckDao.unlockUser(username, ip);
}

private String getWarning(Integer maxFail, Integer curFail){
    if(maxFail <= curFail){
        return "该账户已被锁定";
    }else{
        return "还有"+(maxFail-curFail)+"次机会";
    }
}
private void resetUserLoginCheck(String username, String ip){
    UserLoginCheck userLoginCheck = getUserLoginCheck(username, ip);
    if(userLoginCheck != null){
        userLoginCheck.setTimes(0);
        userLoginCheck.setUpdateTime(new Date());
        userLoginCheckDao.saveOrUpdate(userLoginCheck);
    }
}
private Integer saveUserCheckInfo(Users user, String ip){
    UserLoginCheck userLoginCheck = getUserLoginCheck(user.getStrName(), ip);
    if(userLoginCheck == null){
        userLoginCheck = new UserLoginCheck();
        userLoginCheck.setUsername(user.getStrName());
        userLoginCheck.setUserIp(ip);
        userLoginCheck.setTimes(1);
    }else{
        userLoginCheck.setTimes(userLoginCheck.getTimes()+1);
    }
```

```
    userLoginCheck.setUpdateTime(new Date());
    userLoginCheckDao.saveOrUpdate(userLoginCheck);
    return userLoginCheck.getTimes();
}

public AjaxData validateLoginUser(Users entity, String ip) {
    AjaxData ajaxData = new AjaxData(false);

    SysconfigDto sysconfigDto = sysconfigService.getSysconfig();
    Integer maxFail = sysconfigDto.getMaxFailNum();
    // 1，授权检查
    String authorizeInfo = this.getAuthorizeInfo();
    if (authorizeInfo != null) {
        Integer curFail = saveUserCheckInfo(entity, ip);
        //return ajaxData.setFailureMsg(authorizeInfo+","+getWarning(maxFail, curFail));
        return ajaxData.setFailureMsg(authorizeInfo);
    }

    // 2，检查用户
    List<Users> list = usersDao.findBy(new String[] {"strName"}, new Object[] {entity.getStrName()});
    // 2.1，用户名不存在
    if(list == null || list.size() == 0) {
        return ajaxData.setFailureMsg(GlobalMessageUtil.getMessage("login.usernameerror"));
    } else {
        Users user = list.get(0);
        ajaxData.put("user", user);
        // 2.2，密码不匹配
        if(!user.getStrPassword().equals(entity.getStrPassword())) {
            Integer curFail = saveUserCheckInfo(user, ip);
            return ajaxData.setFailureMsg(GlobalMessageUtil.getMessage("login.pwderror")+","+getWarning(maxFail, curFail));
        } else {
            if(!user.getStrName().equals("admin")) {
                // 2.3，用户已经过期
                if(user.getDtExpirationTime().before(new Date())) {
                    Integer curFail = saveUserCheckInfo(user, ip);
                    return ajaxData.setFailureMsg(GlobalMessageUtil.getMessage("login.expire")+","+getWarning(maxFail, curFail));
                }
                // 2.4，用户已经禁用
                if(user.getNstate() == 0) {
                    Integer curFail = saveUserCheckInfo(user, ip);
                    return ajaxData.setFailureMsg(GlobalMessageUtil.getMessage("login.disable")+","+getWarning(maxFail, curFail));
                }
            }

            // 2.5，验证密码保险期
            Integer passwordFresh = Integer.parseInt(sysconfigDto.getPasswordFresh());
            if(passwordFresh != null && passwordFresh != 0) {
                Calendar calendar = Calendar.getInstance();
                calendar.add(Calendar.MONTH, -passwordFresh);
                Date date = calendar.getTime();
                if(user.getDtUpdatePwdTime().before(date)) {
                    ajaxData.put("type", FailureType.PASSWORD_UNFRESH);
                    Integer curFail = saveUserCheckInfo(user, ip);
                    return ajaxData.setFailureMsg(GlobalMessageUtil.getMessage("login.pwd.unfresh")+","+getWarning(maxFail, curFai
                }
            }

            //2.3，判断首次登录，记录是否首次登录
            if(user.getDtLastOnlineTime()==null){
                user.setDtLastOnlineTime(new Date());//设置最后的登录时间
                usersDao.update(user);

                resetUserLoginCheck(user.getStrName(), ip);

                ajaxData.setAutoSet(false);
                ajaxData.put("type", FailureType.PASSWORD_DEFAULT);
                ajaxData.setFailureMsg(GlobalMessageUtil.getMessage("login.login.first"));
```

```
                return ajaxData;
            }

            //2.2，判断密码是否为初始密码
            String passwordDefault = sysconfigDto.getPasswordDefault();
            if (user.getStrPassword().equals(passwordDefault)) {
                resetUserLoginCheck(user.getStrName(), ip);

                ajaxData.setAutoSet(false);
                ajaxData.put("type", FailureType.PASSWORD_DEFAULT);
                ajaxData.setFailureMsg(GlobalMessageUtil.getMessage("login.pwd.default"));
                return ajaxData;
            }

            // 2.6，没有绑定角色
            List<Role> roles = roleDao.getRoleByUserId(user.getId());
            if(roles == null || roles.size() == 0) {
                Integer curFail = saveUserCheckInfo(user, ip);
                return ajaxData.setFailureMsg(GlobalMessageUtil.getMessage("login.userroleerror",new String[] {entity.getStrName()
            } else {
                // 当前系统只支持单角色绑定
                Role role = roles.get(0);
                ajaxData.put("role", role);
                // 获取角色菜单权限
                List<Power> powers = null;
                if (role.getNtype() == RoleType.SYSTEM_MANAGER) {
                    powers = powerDao.getAll();
                } else {
                    powers = powerDao.getPowersByRoleId(role.getId());
                }
                List<MenuBean> menus = buildMenus(powers, "0", Constants.SYSYEMBS); // strSuper为0是一级菜单
                List<MenuBean> menuscs = buildMenus(powers, "0", Constants.SYSYEMCS); // strSuper为0是一级菜单
                UserSession userSession = buildUserSession(user, role, powers);

                user.setDtLastOnlineTime(new Date());//设置最后的登录时间

                ajaxData.put("menus", menus);
                ajaxData.put("menuscs", menuscs);
                ajaxData.put(SessionAttrbutes.USER_SESSION, userSession);

                resetUserLoginCheck(user.getStrName(), ip);
            }
        }
    }
    return ajaxData;
}
```

实体类字段添加

Users

```
https://192.0.0.241/APP-Financial/branches/base_develop/iVMS-8000-CVTMS/Dev_v1.0.3/web/finance-cvtms-web/src/main/java/com/hikvisi

//2017-02-23 ADD
private Date dtLastOnlineTime;//最后登录系统的时间
```

Workstate

```
https://192.0.0.241/APP-Financial/branches/base_develop/iVMS-8000-CVTMS/Dev_v1.0.3/web/finance-cvtms-web/src/main/java/com/hikvisi

private Integer resetFlag;//0 未重置密码 1 重置密码
```

Device

```
https://192.0.0.241/APP-Financial/branches/base_develop/iVMS-8000-CVTMS/Dev_v1.0.3/web/finance-cvtms-web/src/main/java/com/hikvisi

//2017-02-23 ADD
private Integer passwordLevel;//密码等级
```

Vehicle

```
https://192.0.0.241/APP-Financial/branches/base_develop/iVMS-8000-CVTMS/Dev_v1.0.3/web/finance-cvtms-web/src/main/java/com/hikvisi

//2017-02-23 ADD
private Integer passwordLevel;//密码等级
```

CMSException

```
https://192.0.0.241/APP-Financial/branches/base_develop/iVMS-8000-CVTMS/Dev_v1.0.3/web/finance-cvtms-web/src/main/java/com/hikvisi

/**
 * 用户密码重置
 */
public static final int PASSWORD_RESET_ERROR = 10100003;
```

IWorkStateDao 增加

```
https://192.0.0.241/APP-Financial/branches/base_develop/iVMS-8000-CVTMS/Dev_v1.0.3/web/finance-cvtms-web/src/main/java/com/hikvisi

/**
 * 修改重置密码标志
 * @author wangxinglei 2017年3月2日 下午2:04:17
 * @param ids
 */
public void updateByIds(List<Integer> ids);
```

WorkStateQo

```
https://192.0.0.241/APP-Financial/branches/base_develop/iVMS-8000-CVTMS/Dev_v1.0.3/web/finance-cvtms-web/src/main/java/com/hikvisi

private List<Integer> nuserOrSerIds;
```

WorkStateDaoImpl 增加和修改

```
https://192.0.0.241/APP-Financial/branches/base_develop/iVMS-8000-CVTMS/Dev_v1.0.3/web/finance-cvtms-web/src/main/java/com/hikvisi

protected Criteria buildCriteria(Criteria criteria, WorkStateQo qo) {
    if (qo != null) {
        //...保持不变

        //增加
        if (qo.getNuserOrSerIds() != null) {
            criteria.add(Restrictions.in("nuserOrSerId", qo.getNuserOrSerIds()));
        }
    }
    return criteria;
}

@Override
public void updateByIds(List<Integer> ids) {
    String hql = "update Workstate a set a.resetFlag = 1 where a.nuserOrSerId in (:ids)";
    Query q = this.createQuery(hql);
    q.setParameterList("ids", ids);
    q.executeUpdate();
```

```
}
```

HReqClientLoginServiceImpl

```
https://192.0.0.241/APP-Financial/branches/base_develop/iVMS-8000-CVTMS/Dev_v1.0.3/web/finance-cvtms-web/src/main/java/com/hikvisi

private IUserLoginCheckDao userLoginCheckDao;
private ISysconfigService sysconfigService;
private IUserIdAndSessionService userIdAndSessionService;
/**
 * 用户密码输入正确后，重置用户校验信息
 * @author wangxinglei 2017年3月20日 下午3:09:32
 */
private void resetUserLoginCheck(String username, String ip){
    UserLoginCheck userLoginCheck = getUserLoginCheck(username, ip);
    if(userLoginCheck != null){
        userLoginCheck.setTimes(0);
        userLoginCheck.setUpdateTime(new Date());
        userLoginCheckDao.saveOrUpdate(userLoginCheck);
    }
}

/**
 * 提示信息
 * @author wangxinglei 2017年3月20日 下午7:46:39
 * @param maxFail
 * @param curFail
 * @return
 */
private String getWarning(Integer maxFail, Integer curFail){
    if(maxFail <= curFail){
        return "该账户已被锁定";
    }else{
        return "还有"+(maxFail-curFail)+"次机会";
    }
}

/**
 * 用户登录检查项数据更新
 * @author wangxinglei 2017年2月22日 下午4:55:58
 * @param user
 * @param ip
 */
private Integer saveUserCheckInfo(String username, String ip){
    UserLoginCheck userLoginCheck = getUserLoginCheck(username, ip);
    if(userLoginCheck == null){
        userLoginCheck = new UserLoginCheck();
        userLoginCheck.setUsername(username);
        userLoginCheck.setUserIp(ip);
        userLoginCheck.setTimes(1);
    }else{
        userLoginCheck.setTimes(userLoginCheck.getTimes()+1);
    }
    userLoginCheck.setUpdateTime(new Date());
    userLoginCheckDao.saveOrUpdate(userLoginCheck);
    return userLoginCheck.getTimes();
}

/**
 * 通过用户名和IP获取UserLoginCheck信息
 * @author wangxinglei 2017年3月20日 下午7:24:09
 * @param username
 * @param ip
 * @return
 */
private UserLoginCheck getUserLoginCheck(String username, String ip){
```

```
    UserLoginCheckQo qo = new UserLoginCheckQo();
    qo.setUsername(username);
    qo.setUserIp(ip);
    UserLoginCheck userLoginCheck = userLoginCheckDao.queryUnique(qo);
    return userLoginCheck;
}

/**
 * 验证码和用户IP锁定检查
 * @author wangxinglei 2017年2月22日 下午4:37:49
 * @param ip
 * @return
 */
private String userLoginCheck(String username, String ip){
    UserLoginCheck userLoginCheck = getUserLoginCheck(username, ip);
    if(userLoginCheck != null){
        Integer times = userLoginCheck.getTimes();
        Date updateTime = userLoginCheck.getUpdateTime();

        SysconfigDto sysconfigDto = sysconfigService.getSysconfig();
        Integer maxFailNum = sysconfigDto.getMaxFailNum();
        Integer lockMinutes = sysconfigDto.getLockMinutes();
        long minusTime = System.currentTimeMillis() - updateTime.getTime();
        if("1".equals(String.valueOf(sysconfigDto.getEnableLock())) && times >= maxFailNum ){//判断用户或IP是否锁定，是否开启锁定功能
            if(minusTime < 1000L*60*lockMinutes){
                long leftTime = 1000L*60*lockMinutes - minusTime;
                long seconds = leftTime / 1000 ;
                if(leftTime % 1000 != 0){
                    seconds += 1;
                }
                long leftMinutes = seconds / 60;
                long leftSeconds = seconds % 60;
                if(leftMinutes == 0 && leftSeconds != 0){
                    return "该账户和IP已被锁定，请"+leftSeconds+"秒后再试";
                }else if(leftMinutes != 0 && leftSeconds == 0){
                    return "该账户和IP已被锁定，请"+leftMinutes+"分钟后再试";
                }else if(leftMinutes != 0 && leftSeconds != 0){
                    return "该账户和IP已被锁定，请"+leftMinutes+"分钟"+leftSeconds+"秒后再试";
                }
            }else{
                userLoginCheck.setTimes(0);
                userLoginCheckDao.saveOrUpdate(userLoginCheck);
            }

        }
    }
    return null;
}

/**
 * 处理密码保鲜
 * @author fuqunqing 2015年7月17日 下午2:10:38
 * @param user
 * @param userPwdCheck
 */
private void checkUserPswdExpiration(Users user, String ip, RspClientLoginProto.UserPwdCheck.Builder userPwdCheck, Integer maxFail
    Date updatePwdTime = null;
    Date now = new Date();

    Sysconfig config = sysconfigDao.getSysconfigByKey(SysConfigType.KEEP_PASSWORD_FRESH);

    //大于0表示启用密码保鲜
    if (config != null && StringUtils.isNotEmpty(config.getStrValue()) && (Integer.parseInt(config.getStrValue()) > 0)) {
        userPwdCheck.setIsPwdcheckEnabled(Constants.ENABLED);
        int nMonth = Integer.valueOf(config.getStrValue());
        updatePwdTime = user.getDtUpdatePwdTime();
        if(updatePwdTime==null)
```

```
        updatePwdTime = DateUtils.getDateTime(DateUtils.yyyy_MM_dd_HH_mm_ss, "2000-09-26 00:00:00");

        Date pwdFreshDate = DateUtils.addMonths(updatePwdTime, nMonth);
        //超出需要修改密码最大时间
        if(now.after(pwdFreshDate)){
            Integer curFail = saveUserCheckInfo(user.getStrName(), ip);
            throw new CMSException(CMSException.USER_PSWD_EXPIRATION, "该用户密码保鲜期已过，"+getWarning(maxFail, curFail));
        }else{
            Long diffDay = (pwdFreshDate.getTime()-now.getTime())/DateUtils.MILLIS_PER_DAY;
            userPwdCheck.setRemainingDays(diffDay.intValue());
        }
    }else{
        userPwdCheck.setIsPwdcheckEnabled(Constants.UNENABLED);
    }
}

public void notifyReqClientLogin(RpcController controller, ReqClientLogin request, RpcCallback<RspClientLogin> done) {

    //... 不变

    try {
        // 2，验证连接状态与授权信息
        verifyBasic();
        verifyAuthorize(rspClientLogin);

        String username = request.getUserName();
        String checkResult = userLoginCheck(username, clientIP);
        if(null != checkResult){
            throw new CMSException(CMSException.LOGIN_USER_UNKONWN_ERROR, checkResult);
        }

        SysconfigDto sysconfigDto = sysconfigService.getSysconfig();
        Integer maxFail = sysconfigDto.getMaxFailNum();

        // 3，验证用户
        UsersQo userqo = new UsersQo();
        userqo.setStrName(username);
        userqo.setStrPassword(request.getUserPwd());
        Users user = usersDao.queryUnique(userqo);
        // 3.1，验证用户名是否存在或密码是否匹配
        if(user == null) {
            Integer curFail = saveUserCheckInfo(username, clientIP);
            throw new CMSException(CMSException.LOGIN_USER_LOGIN_INFO__ERROR, "用户名不存在或密码错误，"+getWarning(maxFail, curFail))
        }
        if(!user.getStrName().equals("admin")) {
            // 3.2，用户已经过期
            if(user.getDtExpirationTime().before(new Date())) {
                Integer curFail = saveUserCheckInfo(username, clientIP);
                throw new CMSException(CMSException.LOGIN_USER_EXPIRE, "该用户已到期，"+getWarning(maxFail, curFail));
            }
            // 3.3，用户已经禁用
            if(user.getNstate() == 0) {
                Integer curFail = saveUserCheckInfo(username, clientIP);
                throw new CMSException(CMSException.LOGIN_USER_DISABLED, "该用户已禁用，"+getWarning(maxFail, curFail));
            }
        }
        // 3.4，验证密码保险期
        checkUserPswdExpiration(user, clientIP, userPwdCheck, maxFail);
        rspClientLogin.setUserPwdCheck(userPwdCheck);

        //2.3，判断首次登录，记录是否首次登录
        if(user.getDtLastOnlineTime()==null){
            resetUserLoginCheck(username, clientIP);
            throw new CMSException(CMSException.LOGIN_USER_UNKONWN_ERROR, "首次登录，请在web端修改用户密码");
        }

        //2.2，判断密码是否为初始密码
```

```
        String passwordDefault = sysconfigDto.getPasswordDefault();
        if (user.getStrPassword().equals(passwordDefault)) {
            resetUserLoginCheck(username, clientIP);
            throw new CMSException(CMSException.LOGIN_USER_UNKONWN_ERROR, "密码为初始密码，请在web端修改用户密码");
        }

        // 3.5，验证是否绑定角色
        List<Role> roles = roleDao.getRoleByUserId(user.getId());
        if(roles == null || roles.size() == 0) {
            Integer curFail = saveUserCheckInfo(username, clientIP);
            throw new CMSException(CMSException.LOGIN_USER_UNROLE, "该用户未绑定角色，"+getWarning(maxFail, curFail));
        }

        //... 不变
        rspClientLogin.setCopyright(getSysconfigValue(SysConfigType.PLATFORM_COPYRIGHT));// 设置版权描述信息

        resetUserLoginCheck(username, clientIP);

        // 返回数据
        done.run(rspClientLogin.build());
        logger.info("用户 " + user.getId()+":"+ user.getStrName() + " 登录成功");

        //增加在线信息
        CsClientContextUtil.addCsClientSession(user.getId(), HppConstants.TYPE_SERVER + sessionId, clientIP);

    } catch (CMSException e) {
        //...
    }
    //...
}
```

HWorkStateServiceImpl

```
https://192.0.0.241/APP-Financial/branches/base_develop/iVMS-8000-CVTMS/Dev_v1.0.3/web/finance-cvtms-web/src/main/java/com/hikvisi

public void operWWorkState(RpcController controller, WWorkState request, RpcCallback<RspServerData> done) {

    //...
        Workstate workstate = workStateDao.queryUnique(workStateQo);
        if(null != workstate && null != workstate.getResetFlag() && 1 == workstate.getResetFlag()){
            workstate.setResetFlag(0);
            workStateDao.update(workstate);

            rsp.setResult(CMSException.PASSWORD_RESET_ERROR);
            rsp.setDataId(HppConstants.RESULT_ERROR);
            rsp.setErrorMsg(PbUtil.parseString("用户密码已重置，请重新登录"));
            done.run(rsp.build());
            return;
        }

    //...

}
```

CvrDeviceDaoImpl

```
https://192.0.0.241/APP-Financial/branches/base_develop/iVMS-8000-CVTMS/Dev_v1.0.3/web/finance-cvtms-web/src/main/java/com/hikvisi

public Page queryPageCvrDevice(Integer limit, Integer start, Integer norgId, Integer type,String keyword) {
    SqlQuery sqlQuery = new SqlQuery();
    //增加密码等级
    StringBuffer sql = new StringBuffer("select ve.id AS id,ve.strUser AS strUser, ve.strIp AS strIp,ve.nport AS nport,ve.strName
            + "LEFT JOIN organization org on org.id=ve.norgId "
            + "where ve.ntype=? ");
```

```
    //...
}
```

CvrDeviceInfo

```
https://192.0.0.241/APP-Financial/branches/base_develop/iVMS-8000-CVTMS/Dev_v1.0.3/web/finance-cvtms-web/src/main/java/com/hikvisi

private Integer passwordLevel;//密码等级
```

CvrDeviceServiceImpl

```
https://192.0.0.241/APP-Financial/branches/base_develop/iVMS-8000-CVTMS/Dev_v1.0.3/web/finance-cvtms-web/src/main/java/com/hikvisi

public ResultData updateCvrDevice(CvrDeviceInfo cvrDeviceInfo) {
    //...

    // 更新CVR信息
    vehicle.setStrName(cvrDeviceInfo.getStrName());
    vehicle.setStrIp(cvrDeviceInfo.getStrIp());
    vehicle.setNport(cvrDeviceInfo.getNport());
    vehicle.setStrUser(cvrDeviceInfo.getStrUser());
    //vehicle.setStrPassword(cvrDeviceInfo.getStrPassword());
    vehicle.setPasswordLevel(cvrDeviceInfo.getPasswordLevel());
    cvrDeviceDao.update(vehicle);

    // 更新设备和服务的关联关系
    //...
}

public ResultData addCvrDevice(CvrDeviceInfo cvrDeviceInfo) {
    //...

    // 保存CVR信息
    //...
    vehicle.setStrUser(cvrDeviceInfo.getStrUser());//用户名
    String pass = SercurityUtil.encrypt(PasswordEncryptUtil.decodeBase64(cvrDeviceInfo.getStrPassword()));
    vehicle.setStrPassword(pass);//密码
    vehicle.setPasswordLevel(cvrDeviceInfo.getPasswordLevel());//设备密码等级
    Integer vehicleId = (Integer) cvrDeviceDao.save(vehicle);

    // 保存设备和服务的关联关系
    //...

}

public ResultData updateCVRPwd(CvrDeviceInfo cvrDeviceInfo) {
    ResultData resultData = new ResultData();
    // 1.验证老密码是否填写正确
    Vehicle cvrDevice = cvrDeviceDao.get(cvrDeviceInfo.getId());
    String passDB = cvrDevice.getStrPassword();// 获取数据库当中的原始密码
    String passFront = SercurityUtil.encrypt(PasswordEncryptUtil.decodeBase64(cvrDeviceInfo.getStrPassword()));// 获取前段传过来的原始
    if (!passDB.equals(passFront)) {
        throw new ExpectedException("", "原始密码输入有误");
    }
    // 2.修改新密码
    String passNew = SercurityUtil.encrypt(PasswordEncryptUtil.decodeBase64(cvrDeviceInfo.getNewPassword()));
    cvrDevice.setStrPassword(passNew);
    cvrDevice.setPasswordLevel(cvrDeviceInfo.getPasswordLevel());
    cvrDeviceDao.update(cvrDevice);
    OperationLogUtils.setContent(GlobalMessageUtil.getMessage("log.cvrDevice.updatePwd", new String[] {cvrDevice.getStrName()}));
    resultData.setId(cvrDeviceInfo.getId());
    resultData.setSuccess(true);
    return resultData;
}
```

```
public ResultData updateCVRPwdReset(CvrDeviceInfo cvrDeviceInfo) {
    ResultData resultData = new ResultData();
    //1.验证登陆密码是否填写正确
    Integer userId = SessionUtil.getUserSession().getUserId();// 通过session的uerid获取数据库当中的原始密码
    String passUser = usersDao.get(userId).getStrPassword();
    String passFrontLogin = cvrDeviceInfo.getLoginPassword();// 获取前段传过来的登陆密码,加密后和数据库登陆密码对比
    if (!passUser.equals(passFrontLogin)) {
        throw new ExpectedException("", "登陆密码输入有误");
    }
    //2.修改新密码
    Vehicle cvrDevice = cvrDeviceDao.get(cvrDeviceInfo.getId());
    String passNew = SercurityUtil.encrypt(PasswordEncryptUtil.decodeBase64(cvrDeviceInfo.getNewPassword()));
    cvrDevice.setStrPassword(passNew);
    cvrDevice.setPasswordLevel(cvrDeviceInfo.getPasswordLevel());
    cvrDeviceDao.update(cvrDevice);
    OperationLogUtils.setContent(GlobalMessageUtil.getMessage("log.cvrDevice.updatePwd", new String[] {cvrDevice.getStrName()}));
    resultData.setId(cvrDeviceInfo.getId());
    resultData.setSuccess(true);
    return resultData;
}
```

EscortplanDaoImpl

```
https://192.0.0.241/APP-Financial/branches/base_develop/iVMS-8000-CVTMS/Dev_v1.0.3/web/finance-cvtms-web/src/main/java/com/hikvisi

public void deleteByIds(Serializable[] paramArrayOfSerializable) {
    StringBuffer hql = new StringBuffer("DELETE FROM ");
    hql.append(getEntityClass().getName());
    hql.append(" WHERE id in (:ids)");
    Query query = createQuery(hql.toString());
    query.setParameter("ids", StringUtils.join(paramArrayOfSerializable));
    query.executeUpdate();
}

public List<Escortplan> queryEscortplanListBySQL(Date start, Date end) {
    String sql1 = "from Escortplan where dtStart <= ? and dtEnd > ? ";
    String sql2 = "from Escortplan where dtStart > ? and dtStart < ? ";

    List<Escortplan> list1 = this.createQuery(sql1).setParameter(0, start).setParameter(1,  end).list();
    List<Escortplan> list2 = this.createQuery(sql2).setParameter(0, start).setParameter(1,  end).list();

    list1.addAll(list2);
    return list1;
}
public List<VehicleTodayEscortStatus> isTodayEscortplan(List<Integer> vehicleIds) {
    String sql = "select distinct nvehicleId from escortplan where dutyDate = ?";
    List<Integer> todayIds = this.createSQLQuery(sql).setParameter(0, DateUtils.getStringDateTime(DateFormatUtils.getZeroTime().ge
    List<VehicleTodayEscortStatus> vtess = new ArrayList<VehicleTodayEscortStatus>();
    //...
}
```

EscortplanstaffDaoImpl

```
https://192.0.0.241/APP-Financial/branches/base_develop/iVMS-8000-CVTMS/Dev_v1.0.3/web/finance-cvtms-web/src/main/java/com/hikvisi

public void deleteByIds(Serializable[] paramArrayOfSerializable) {
    StringBuffer hql = new StringBuffer("DELETE FROM ");
    hql.append(getEntityClass().getName());
    hql.append(" WHERE id in (:ids)");
    Query query = createQuery(hql.toString());
    query.setParameter("ids", StringUtils.join(paramArrayOfSerializable));
    query.executeUpdate();
}
```

EscortplanServiceImpl

```
https://192.0.0.241/APP-Financial/branches/base_develop/iVMS-8000-CVTMS/Dev_v1.0.3/web/finance-cvtms-web/src/main/java/com/hikvisi

public Integer add(Escortplan escortplan, List<Escortplanstaff> escortplanstaffs) {
    if(XSSTranslator.clsInterceptor(escortplanstaffs)){
        throw new ExpectedException("", "参数信息错误");
    }
    // 1.1 验证押运计划是否重复
    if(!escortplanDao.validateSamePlans(escortplan)){
        throw new ExpectedException("", GlobalMessageUtil.getMessage("escortplan.exist"));
    }
    //...
}

public void update(Escortplan escortplan, List<Escortplanstaff> escortplanstaffs) {
    if(XSSTranslator.clsInterceptor(escortplanstaffs)){
        throw new ExpectedException("", "参数信息错误");
    }
    // 1.1验证押运计划是否重复
    if(!escortplanDao.validateSamePlans(escortplan)){
        throw new ExpectedException("", GlobalMessageUtil.getMessage("escortplan.exist"));
    }
    //...
}
```

AlarmeventlogDaoImpl

```
https://192.0.0.241/APP-Financial/branches/base_develop/iVMS-8000-CVTMS/Dev_v1.0.3/web/finance-cvtms-web/src/main/java/com/hikvisi

public int delExpireLog(int num) {
    String sql = "delete from alarmeventlog where To_Days(NOW()) - To_Days(dtBeginTime) > ?";
    return this.createSQLQuery(sql).setParameter(0, num).executeUpdate();
}
```

OperationlogDaoImpl

```
https://192.0.0.241/APP-Financial/branches/base_develop/iVMS-8000-CVTMS/Dev_v1.0.3/web/finance-cvtms-web/src/main/java/com/hikvisi

public void deleteByIds(Serializable[] paramArrayOfSerializable) {
    StringBuffer hql = new StringBuffer("DELETE FROM ");
    hql.append(getEntityClass().getName());
    hql.append(" WHERE id in (:ids)");
    Query query = createQuery(hql.toString());
    query.setParameter("ids", StringUtils.join(paramArrayOfSerializable));
    query.executeUpdate();
}

public int delExpireLog(int num) {
    String sql = "delete from operationlog where To_Days(NOW()) - To_Days(strDate) > ?";
    return this.createSQLQuery(sql).setParameter(0, num).executeUpdate();
}
```

RecordCvrDaoImpl

```
https://192.0.0.241/APP-Financial/branches/base_develop/iVMS-8000-CVTMS/Dev_v1.0.3/web/finance-cvtms-web/src/main/java/com/hikvisi

public List<Recordcvr> getRecordcvrByCamId(List<Integer> channelIds) {
    if (channelIds != null && channelIds.size() > 0) {
        String hql = "from Recordcvr where ncamId in (:ids)";
        return this.getHibernateTemplate().findByNamedParam(hql, "ids", channelIds.toArray());
```

```
    }
    return null;
}

public List<Integer> getCvrPlanByCamId(List<Integer> ids) {
    if(ids!=null && ids.size()>0) {
        String hql = "select nvrmid from Recordcvr where ncamId in ( :ids ) group by nvrmid";
        return this.getHibernateTemplate().findByNamedParam(hql, "ids", ids.toArray());
    }
    return null;
}

public List<Recordcvr> getRecordcvrByVrmIdAndCamId(List<Integer> ids, Integer vrmId) {
    if(ids!=null && ids.size()>0 && vrmId!=null) {
        String hql = "from Recordcvr where  ncamId in (:ids) and nvrmid= :nvrmid";
        return this.getHibernateTemplate().findByNamedParam(hql, new String[]{"ids","nvrmid"}, new Object[]{ids.toArray(), vrmId})
    }
    return null;
}
```

RoleServiceImpl

```
https://192.0.0.241/APP-Financial/branches/base_develop/iVMS-8000-CVTMS/Dev_v1.0.3/web/finance-cvtms-web/src/main/java/com/hikvisi

private Logger logger = Logger.getLogger(RoleServiceImpl.class);

public AjaxData addRole(String jsonStr) {
    AjaxData ajaxData = new AjaxData(false);
    RoleDto roleDto = JsonUtils.json2Object(jsonStr, RoleDto.class);
    Role role = roleDto.getRole();
    if(XSSTranslator.clsInterceptor(role)){
        logger.error("非法的参数信息------------------------------>"+jsonStr);
        return ajaxData.setFailureMsg("非法的参数信息！");
    }
    //验证角色名称是否存在
    if(roleDao.findUniqueBy("strName", role.getStrName()) != null){
        return ajaxData.setFailureMsg("角色名已存在");
    }
    //...
}

public AjaxData updateRole(String jsonStr) {
    AjaxData ajaxData = new AjaxData(false);
    RoleDto roleDto = JsonUtils.json2Object(jsonStr, RoleDto.class);
    Role role = roleDto.getRole();
    if(XSSTranslator.clsInterceptor(role)){
        logger.error("非法的参数信息------------------------------>"+jsonStr);
        return ajaxData.setFailureMsg("非法的参数信息！");
    }
    //...
}
```

GishistorypointinfoDaoImpl

```
https://192.0.0.241/APP-Financial/branches/base_develop/iVMS-8000-CVTMS/Dev_v1.0.3/web/finance-cvtms-web/src/main/java/com/hikvisi

public int delExpireGisPoint(int num) {
    String sql = "delete from gishistorypointinfo where To_Days(NOW()) - To_Days(dtSampleTime) > ?";
    return this.createSQLQuery(sql).setParameter(0, num).executeUpdate();
}
```

RulesServiceImpl

```
https://192.0.0.241/APP-Financial/branches/base_develop/iVMS-8000-CVTMS/Dev_v1.0.3/web/finance-cvtms-web/src/main/java/com/hikvisi

public AjaxData addPolygon(String jsonStr) {
    AjaxData ajaxData = new AjaxData(false);
    GismappolygonDto polygonDto = JsonUtils.json2Object(jsonStr, GismappolygonDto.class);
    Gismappolygon polygon = polygonDto.getPolygon();
    if(XSSTranslator.clsInterceptor(polygon)){
        logger.error("请求被拦截，某些参数不符合输入规范------------>"+jsonStr);
        ajaxData.setFailureMsg("非法的参数信息！");
        return ajaxData;
    }
    // 1，判断名称是否存在
    //...
}

public AjaxData updatePolygon(String jsonStr) {
    UserSession userSession = SessionUtil.getUserSession();
    AjaxData ajaxData = new AjaxData(false);
    GismappolygonDto polygonDto = JsonUtils.json2Object(jsonStr, GismappolygonDto.class);
    Gismappolygon polygon = polygonDto.getPolygon();
    if(XSSTranslator.clsInterceptor(polygon)){
        logger.error("请求被拦截，某些参数不符合输入规范------------>"+jsonStr);
        ajaxData.setFailureMsg("非法的参数信息！");
        return ajaxData;
    }
    // 1，判断名称是否存在
    //...
}
public AjaxData updatePolygonpoint(String jsonStr) {
    AjaxData ajaxData = new AjaxData(false);
    GismappolygonDto polygonDto = JsonUtils.json2Object(jsonStr, GismappolygonDto.class);
    Gismappolygon polygon = polygonDto.getPolygon();
    if(XSSTranslator.clsInterceptor(polygon)){
        logger.error("请求被拦截，某些参数不符合输入规范------------>"+jsonStr);
        ajaxData.setFailureMsg("非法的参数信息！");
        return ajaxData;
    }
    //...
}
```

ServiceUtil

```
https://192.0.0.241/APP-Financial/branches/base_develop/iVMS-8000-CVTMS/Dev_v1.0.3/web/finance-cvtms-web/src/main/java/com/hikvisi

public static Server getServerParam(Server server) {
    int nctrlPort = 0;
    int ndataPort = 0;
    switch(server.getNtype()) {
        //...
        case Constants.SysDictionary.ServerType.MT_SERVICE_VAG:// VAG服务
            nctrlPort = 7300;
            ndataPort = 7302;
            server.setStrLoginName(VAG_LOGIN.VagLoginName);
            server.setStrPassword(SercurityUtil.encrypt(VAG_LOGIN.VagLoginPwd));
            break;
        //...
    }
    server.setNctrlPort(nctrlPort);
    server.setNdataPort(ndataPort);
    return server;
}
```

SysconfigDaoImpl

```
https://192.0.0.241/APP-Financial/branches/base_develop/iVMS-8000-CVTMS/Dev_v1.0.3/web/finance-cvtms-web/src/main/java/com/hikvisi

public void deleteByIds(Serializable[] paramArrayOfSerializable) {
    StringBuffer hql = new StringBuffer("DELETE FROM ");
    hql.append(getEntityClass().getName());
    hql.append(" WHERE id in (:ids)");
    Query query = createQuery(hql.toString());
    query.setParameter("ids", StringUtils.join(paramArrayOfSerializable));
    query.executeUpdate();
}
```

SysconfigDto

```
https://192.0.0.241/APP-Financial/branches/base_develop/iVMS-8000-CVTMS/Dev_v1.0.3/web/finance-cvtms-web/src/main/java/com/hikvisi

private Integer enableLock;//是否启用锁定功能   1 启用  0 禁用
private Integer maxFailNum;//登录时最大连续失败次数
private Integer lockMinutes;//锁定时间
```

SysconfigServiceImpl

```
https://192.0.0.241/APP-Financial/branches/base_develop/iVMS-8000-CVTMS/Dev_v1.0.3/web/finance-cvtms-web/src/main/java/com/hikvisi

public SysconfigDto getSysconfig() {
    SysconfigDto sysconfigDto = new SysconfigDto();
    List<Integer> nkeys = Lists.newArrayList(
            //...
            SysConfigType.LIMIT_PREVIEW,
            SysConfigType.USER_AND_IP_LOCK,
            SysConfigType.FAIL_TIMES,
            SysConfigType.LOCK_MINUTES);
    SysconfigQo sysconfigQo = new SysconfigQo();
    sysconfigQo.setNkeys(nkeys);
    List<Sysconfig> configs = sysconfigDao.queryList(sysconfigQo);
    for(Sysconfig config:configs) {
        switch(config.getNkey()) {
            //...
            case SysConfigType.LIMIT_PREVIEW:
                sysconfigDto.setLimitPreview(config.getStrValue());
                break;
            case SysConfigType.USER_AND_IP_LOCK:
                sysconfigDto.setEnableLock(Integer.parseInt(config.getStrValue()));
                break;
            case SysConfigType.FAIL_TIMES:
                sysconfigDto.setMaxFailNum(Integer.parseInt(config.getStrValue()));
                break;
            case SysConfigType.LOCK_MINUTES:
                sysconfigDto.setLockMinutes(Integer.parseInt(config.getStrValue()));
                break;
        }
    }
    return sysconfigDto;
}

public void updateSysconfig(SysconfigDto sysconfigDto) {
    try {
        if(sysconfigDto!=null){
            //...
            //用户和IP锁定
            if(sysconfigDto.getEnableLock() != null){
                SysconfigQo sysconfigQo = new SysconfigQo();
                sysconfigQo.setNkey(SysConfigType.USER_AND_IP_LOCK);
                Sysconfig sysconfig = sysconfigDao.queryUnique(sysconfigQo);
                if(sysconfig != null) {
```

```
                    sysconfig.setStrValue(String.valueOf(sysconfigDto.getEnableLock()));
                    sysconfig.setStrUpdatetime(new Date());
                    sysconfigDao.update(sysconfig);
                }
            }
            //最大连续失败次数
            if(sysconfigDto.getMaxFailNum() != null){
                SysconfigQo sysconfigQo = new SysconfigQo();
                sysconfigQo.setNkey(SysConfigType.FAIL_TIMES);
                Sysconfig sysconfig = sysconfigDao.queryUnique(sysconfigQo);
                if(sysconfig != null) {
                    sysconfig.setStrValue(String.valueOf(sysconfigDto.getMaxFailNum()));
                    sysconfig.setStrUpdatetime(new Date());
                    sysconfigDao.update(sysconfig);
                }
            }
            //锁定时间
            if(sysconfigDto.getLockMinutes() != null){
                SysconfigQo sysconfigQo = new SysconfigQo();
                sysconfigQo.setNkey(SysConfigType.LOCK_MINUTES);
                Sysconfig sysconfig = sysconfigDao.queryUnique(sysconfigQo);
                if(sysconfig != null) {
                    sysconfig.setStrValue(String.valueOf(sysconfigDto.getLockMinutes()));
                    sysconfig.setStrUpdatetime(new Date());
                    sysconfigDao.update(sysconfig);
                }
            }
        }
    } catch (DataLoadException e) {
        throw new ExpectedException("3");
    }
}
```

SysdictionaryDaoImpl

```
https://192.0.0.241/APP-Financial/branches/base_develop/iVMS-8000-CVTMS/Dev_v1.0.3/web/finance-cvtms-web/src/main/java/com/hikvisi

public void deleteByIds(Serializable[] paramArrayOfSerializable) {
    StringBuffer hql = new StringBuffer("DELETE FROM ");
    hql.append(getEntityClass().getName());

    hql.append(" WHERE id in (:ids)");
    Query query = createQuery(hql.toString());
    query.setParameter("ids", StringUtils.join(paramArrayOfSerializable));
    query.executeUpdate();
}
```

DecodeDeviceServiceImpl

```
https://192.0.0.241/APP-Financial/branches/base_develop/iVMS-8000-CVTMS/Dev_v1.0.3/web/finance-cvtms-web/src/main/java/com/hikvisi

public void saveMfvcDevice(Device mfvcDevice, List<Device> decodeDevices, List<Device> encodeDevices) {
    //...

    // 保存综合平台
    Device device = new Device();
    try {
        PropertyUtils.copyProperties(device, mfvcDevice);
    } catch (Exception e) {
        throw new ExpectedException("");
    }

    device.setStrPassword(SercurityUtil.encrypt(PasswordEncryptUtil.decodeBase64(device.getStrPassword())));
    Integer nrevId = (Integer)deviceDao.save(device);
```

```
    // 循环插入数据库
    for (Device decodeDevice : decodeDevices) {
        decodeDevice.setNrev(nrevId);
        decodeDevice.setStrInterAreaCode(deviceDao.getDeviceSeril(decodeDevice.getNorgId()));
        deviceDao.save(decodeDevice);
    }
    //...
}
```

UsersServiceImpl

```
https://192.0.0.241/APP-Financial/branches/base_develop/iVMS-8000-CVTMS/Dev_v1.0.3/web/finance-cvtms-web/src/main/java/com/hikvisi

private IUserIdAndSessionDao userIdAndSessionDao;
private DataManager dataManager;
private IWorkStateDao workStateDao;

public AjaxData resetPassword(List<Integer> ids) {
    AjaxData ajaxData = new AjaxData(false);
    usersDao.resetUsersPassword(ids, sysconfigService.getSysconfig().getPasswordDefault());

    UserIdAndSessionQo userIdAndSessionQo = new UserIdAndSessionQo();
    userIdAndSessionQo.setUserIds(ids);
    userIdAndSessionQo.setClientType(1);
    List<UserIdAndSession> userIdAndSessions = userIdAndSessionDao.queryList(userIdAndSessionQo);
    if(null != userIdAndSessions){
        for(int i=0; i<userIdAndSessions.size(); i++){
            String sessionId = userIdAndSessions.get(i).getSessionId();
            try {
                dataManager.remove(sessionId, DATA_TYPE.SESSION);
            } catch (DataLoadException e) {
                logger.info("connect to cache error.",e);
                return ajaxData;
            }
        }
    }

    workStateDao.updateByIds(ids);

    //...
}

public AjaxData updatePassword(Integer userId, String password, String newpassword) {
    AjaxData ajaxData = new AjaxData(false);
    Users user = usersDao.get(userId);
    if (user.getStrPassword().equals(password)) {
        if (!password.equals(newpassword)) {
            // 更新密码、修改密码时间
            user.setStrPassword(newpassword);
            user.setDtUpdatePwdTime(new Date());
            usersDao.update(user);

            List<Integer> ids = new ArrayList<Integer>(1);
            ids.add(userId);
            workStateDao.updateByIds(ids);
        } else {
            ajaxData.setFailureMsg("新密码与当前密码相同");
        }
    } else {
        ajaxData.setFailureMsg("当前密码不正确");
    }
    return ajaxData;
}
```

VehicleAction

```
https://192.0.0.241/APP-Financial/branches/base_develop/iVMS-8000-CVTMS/Dev_v1.0.3/web/finance-cvtms-web/src/main/java/com/hikvisi

public String toAddVehiclePage() {
    if (vehicleInfo.getNorgId() != null) {
        vehicleInfo.setNanalogNum(4);
        vehicleInfo.setNalarmInNum(4);
        vehicleInfo.setNalarmOutNum(1);
        vehicleInfo.setNtalkNum(1);
        vehicleInfo.setNdefaultFenceId(null);
        vehicleInfo.setNdefaultLineId(null);
        vehicleInfo.setNdefaultStaffgroupId(null);
        vehicleInfo.setNvagServerId(null);
        vehicleInfo.setNstreamServerId(null);
        vehicleInfo.setNdvrServerId(null);
        vehicleInfo.setNvrmServerId(null);
        operPage = "/modules/vehicles/dialogs/vehicle-win.jsp";
        return DISPATCHER;
    } else {
        msg = "未获取到车队id";
        success = false;
        return AJAX;
    }
}

public String getPolygonLine(){
    if (id != null) {
        vehicleInfo = vehicleService.getVehicleInfo(Integer.valueOf(id.toString()));
        success = true;
    }
    lines = vehicleService.getLinesOrFences(Constants.PolygonType.LINE);
    fences = vehicleService.getLinesOrFences(Constants.PolygonType.FENCE);
    staffgroups = vehicleService.getStaffGroups();
    vagServers = serverService.getServerBasicInfoByType(ServerType.MT_SERVICE_VAG);
    streamServers = serverService.getServerBasicInfoByType(ServerType.MT_SERVICE_STREAM);
    dvrServers = serverService.getServerBasicInfoByType(ServerType.MT_SERVICE_DVR);
    vrmServers = serverService.getServerBasicInfoByType(ServerType.MT_SERVICE_VRM);
    ajaxData.put("vehicleInfo", vehicleInfo);
    ajaxData.put("lines", lines);
    ajaxData.put("fences", fences);
    ajaxData.put("staffgroups", staffgroups);
    ajaxData.put("vagServers", vagServers);
    ajaxData.put("streamServers", streamServers);
    ajaxData.put("dvrServers", dvrServers);
    ajaxData.put("vrmServers", vrmServers);
    return AJAX;
}

/**
 * 跳转到车辆编辑页面
 * @author jinmeiling 2015年7月19日 下午6:28:41
 * @return
 */
public String toEditVehiclePage() {
    if (id != null) {
        vehicleInfo = vehicleService.getVehicleInfo(Integer.valueOf(id.toString()));
        operPage = "/modules/vehicles/dialogs/vehicle-win.jsp";
        return DISPATCHER;
    } else {
        msg = "未获取到车辆id";
        success = false;
        return AJAX;
    }
}
```

VehiclechannelDaoImpl

```
https://192.0.0.241/APP-Financial/branches/base_develop/iVMS-8000-CVTMS/Dev_v1.0.3/web/finance-cvtms-web/src/main/java/com/hikvisi

public List<RegionResDTO> queryRegionResList(ReqServerDataQo qo) {
    //...

    if(qo.getMoudleType() != null && StringUtils.isNotEmpty(qo.getServerIp())){
        sql += " join servervehiclemapping sv on sv.nvehicleId = v.id "
                + " join server vag on ( vag.id = sv.nserverId and vag.ntype = :serverType ) "
                + " join pcserverip ip on ( ip.npcserverId = vag.npcId and ip.ntype = "+NetType.NET_IN +" and ip.strIp = :ip ) "
                + " where vc.nstate= "+Constants.ENABLED+" ";
        if(qo.getSubType()!=null){
            sql +=" and vc.ntype = :ntype";
        }
        return this.createSQLQuery(sql).setParameter("ntype", qo.getSubType()).setParameter("serverType", qo.getMoudleType()).setP
    }else if(qo.getMoudleType() == ServerType.MT_SERVICE_EVENT){
        sql +=" where 1=1 ";
        if(StringUtils.isNotEmpty(qo.getDataIndex())){
            sql += " and vc.strInterAreaCode = :strInterAreaCode";
        }
        return this.createSQLQuery(sql).setParameter("strInterAreaCode", qo.getDataIndex()).setResultTransformer(Transformers.alia
    }

    sql +=" where 1=1 ";
    if(qo.getDataId()!=null){
        sql += " and vc.id = :vcid";
        return this.createSQLQuery(sql).setParameter("vcid", qo.getDataId()).setResultTransformer(Transformers.aliasToBean(RegionR
    }
    if(qo.getRelativeDevId()!=null){
        sql += " and vc.nvehicleId = :vcnvehicleId";
        return this.createSQLQuery(sql).setParameter("vcnvehicleId", qo.getRelativeDevId()).setResultTransformer(Transformers.alia
    }

    //...
}
```

IDeviceService

```
https://192.0.0.241/APP-Financial/branches/base_develop/iVMS-8000-CVTMS/Dev_v1.0.3/web/finance-cvtms-web/src/main/java/com/hikvisi

/**
 * 修改密码
 * @author wangxinglei 2017年2月23日 下午2:55:09
 * @param strPassword
 * @param strNewPassword
 * @param id
 * @param passwordLevel 密码安全等级
 */
public void updateDevicePwd(String strPassword, String strNewPassword, Integer id, Integer passwordLevel);
```

DeviceServiceImpl

```
https://192.0.0.241/APP-Financial/branches/base_develop/iVMS-8000-CVTMS/Dev_v1.0.3/web/finance-cvtms-web/src/main/java/com/hikvisi

@Override
public void updateDevicePwd(String strPassword, String strNewPassword,Integer id, Integer passwordLevel) {
    Device device = deviceDao.get(id);

    // 1、验证密码
    strPassword = SercurityUtil.encrypt(PasswordEncryptUtil.decodeBase64(strPassword));
    if (!strPassword.equals(device.getStrPassword())) {
        throw new ExpectedException("", "原始密码输入有误");
    }
```

```
    // 2、修改密码
    strNewPassword = SercurityUtil.encrypt(PasswordEncryptUtil.decodeBase64(strNewPassword));
    device.setStrPassword(strNewPassword);
    device.setPasswordLevel(passwordLevel);
    deviceDao.update(device);

    OperationLogUtils.setContent(GlobalMessageUtil.getMessage("log.mfvcDevice.changePwd", new String[]{device.getStrName()}));
}

@Override
public void updateDevicePwdReset(String strPassword, String strNewPassword, Integer id, Integer passwordLevel) {
    Device device = deviceDao.get(id);

    // 1、验证用户密码
    Integer userid = SessionUtil.getUserSession().getUserId();
    String passUser = usersDao.get(userid).getStrPassword();
    if (!strPassword.equals(passUser)) {
        throw new ExpectedException("", "登陆密码输入有误");
    }
    // 2、修改密码
    strNewPassword = SercurityUtil.encrypt(PasswordEncryptUtil.decodeBase64(strNewPassword));
    device.setStrPassword(strNewPassword);
    device.setPasswordLevel(passwordLevel);
    deviceDao.update(device);

    OperationLogUtils.setContent(GlobalMessageUtil.getMessage("log.mfvcDevice.changePwd", new String[]{device.getStrName()}));
}
```

web.xml

```
https://192.0.0.241/APP-Financial/branches/base_develop/iVMS-8000-CVTMS/Dev_v1.0.3/web/finance-cvtms-web/src/main/webapp/WEB-INF/w

 <servlet>
    <servlet-name>kaptcha</servlet-name>
    <servlet-class>com.google.code.kaptcha.servlet.KaptchaServlet</servlet-class>
    <init-param>
        <param-name>kaptcha.image.width</param-name>
        <param-value>200</param-value>
    </init-param>
    <init-param>
        <param-name>kaptcha.image.height</param-name>
        <param-value>85</param-value>
    </init-param>
    <init-param>
        <param-name>kaptcha.border.color</param-name>
        <param-value>white</param-value>
    </init-param>
    <init-param>
        <param-name>kaptcha.textproducer.char.length</param-name>
        <param-value>4</param-value>
    </init-param>
    <init-param>
        <param-name>kaptcha.textproducer.char.string</param-name>
        <param-value>23456789QWERTYUIPASDFGHJKZXCVBNMqwertyuipasdfghjkzxcvbnm</param-value>
    </init-param>
    <init-param>
        <param-name>kaptcha.textproducer.font.size</param-name>
        <param-value>60</param-value>
    </init-param>
    <init-param>
        <param-name>kaptcha.textproducer.char.space</param-name>
        <param-value>6</param-value>
    </init-param>
    <init-param>
        <param-name>kaptcha.textproducer.font.color</param-name>
        <param-value>black</param-value>
```

```
    </init-param>
    <init-param>
        <param-name>kaptcha.textproducer.font.names</param-name>
        <param-value>宋体,楷体,微软雅黑</param-value>
    </init-param>
    <init-param>
        <param-name>kaptcha.background.clear.from</param-name>
        <param-value>white</param-value>
    </init-param>
    <init-param>
        <param-name>kaptcha.background.clear.to</param-name>
        <param-value>white</param-value>
    </init-param>
</servlet>
<servlet-mapping>
    <servlet-name>kaptcha</servlet-name>
    <url-pattern>/randomCode.jpg</url-pattern>
</servlet-mapping>
```