许可证构件的重构

一、 所选构件

https://github.com/SummerWish/TahitiQuotaLimiter

二、 构件选择理由

1. 接口简洁，使用 tryAcquire()方法进行申请许可服务，直接通过返回式 true 还是 false 判断是否超过预设阈值；
2. 能够进行重置和修改配额的功能；
3. 能够进行批量请求，例如一次性请求 3 个则可以使用 tryAcquire(3)的方式，该功能虽然我们的项目里没用到，但是也使得该构件更加灵活。

三、 重构实现

1. 在原有代码的基础上进行更改，判断 tryAcquire()方法的返回值并进入不同的状态。部分代码如下所示：

```
public void OnSend(HashMap<String, String> args) {
    /*
     * 1.判断用户状态
     * 2.发送消息
     * 3.更新用户状态
     */

    String message;
    String msgToSend;

    incLocalReceiveMsgNum();

    if (getStatus() == Status.LOGIN || getStatus() == Status.IGNORE) {
        if (! getThroughputLimiter().tryAcquire())
            setStatus(Status.IGNORE);
        else {
            setStatus(Status.LOGIN);
            if (! getCapacityLimiter().tryAcquire())
                setStatus(Status.RELOGIN);
        }
    }

    switch (getStatus()) {
        case LOGOUT:
            incLocalIgnoreMsgNum();
            break;
        case LOGIN:
```

2. 在进行对消息的许可限制时，我们需要从服务端的配置文件里读入预设阈值，这里又使用到了 TahitiConfigManager，代码如下：

```
ConfigManager configManager = new ConfigManager(new JsonAdapter(), "./ServerConfig.json");
ConfigBean config = null;
try {
    config = configManager.loadToBean(ConfigBean.class);
} catch (IOException e) {
    e.printStackTrace();
}
this.capacityLimiter = new CapacityLimiter(config.getMAX_NUMBER_PER_SESSION());
this.throughputLimiter = new ThroughputLimiter(config.getMAX_NUMBER_PER_SECOND());
```

通过读入配置文件的 MAX_NUMBER_PER_SESSION 和 MAX_NUMBER_PER_SECOND 进行设置阈值。

四、 构件改进意见

1. 最开始该构件没有重置的方法，当一个 CapacityLimiter 的配额用完之后，只能重新实例化一个新的 CapacityLimiter，所以我在 github 上提了 Issue，之后加上了重置和动态设置阈值的方法，这是一个改进。
2. 另一个意见是 ThroughputLimiter 目前只有以秒为单位的流量限制功能，虽然已经满足了该项目的需求，但是考虑的更宽广的适用性，可以加上对不同时间单位的支持，如每天多少。