大家好，我要介绍的主题是规则引擎

我们首先从一个简短的中台常规需求来聊起, 这是上个月产品经理提出了一个需求，需求描述是当用户在客诉系统创建客诉工单时，

创建客诉工单-投诉原因出现如下场景时,单据创建成功后需触发飞书提醒:

紧急产品事故-人身伤害

紧急产品事故-机器烧了/明火

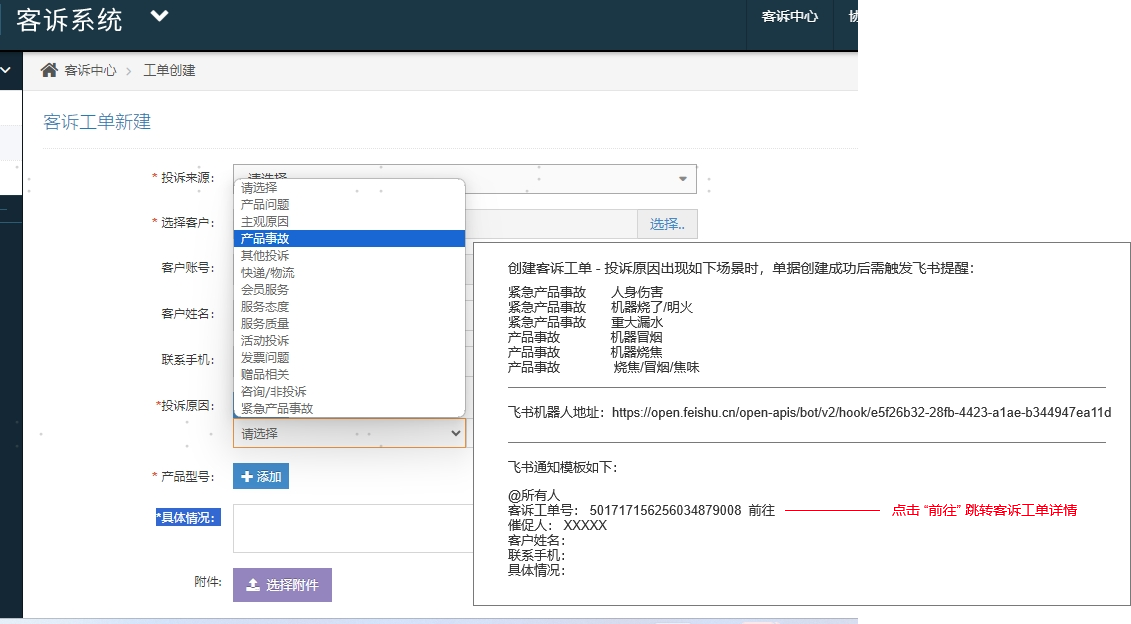
紧急产品事故-重大漏水

产品事故-机器冒烟

产品事故-机器烧焦

产品事故-烧焦/冒烟/焦味



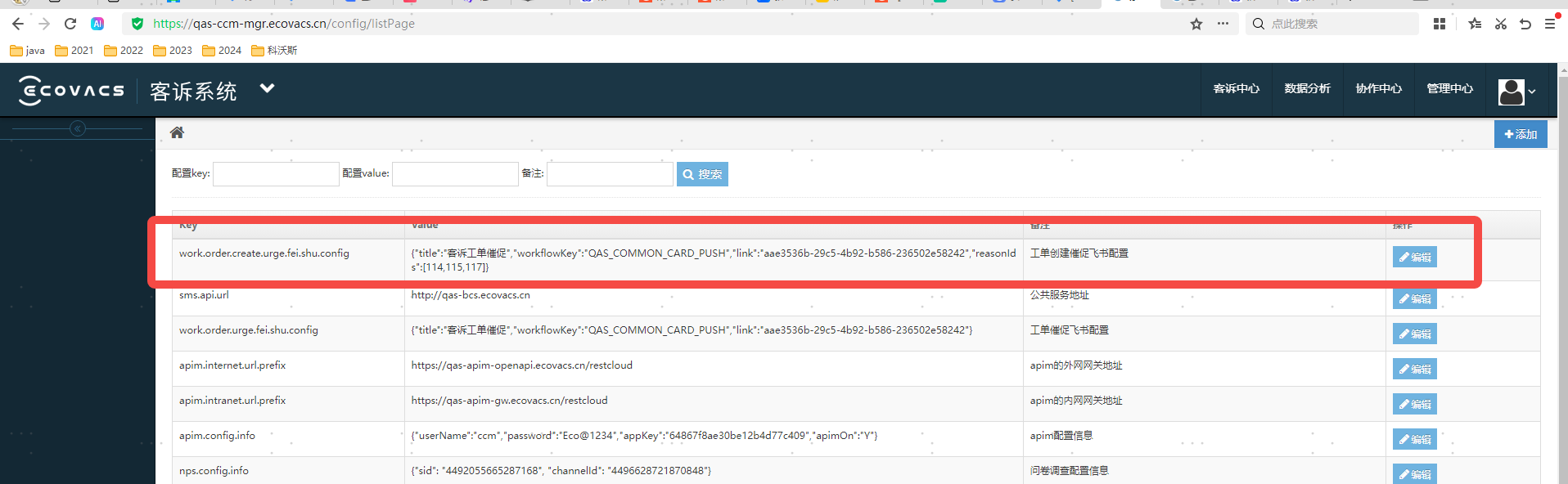


基于java编程语言



其中需要推送的投诉原因，我们维护成为了一个配置项，也就是说投诉原因发生变更的时候，我们只需要修改配置项，而不用修改代码，也不用测试和上线。

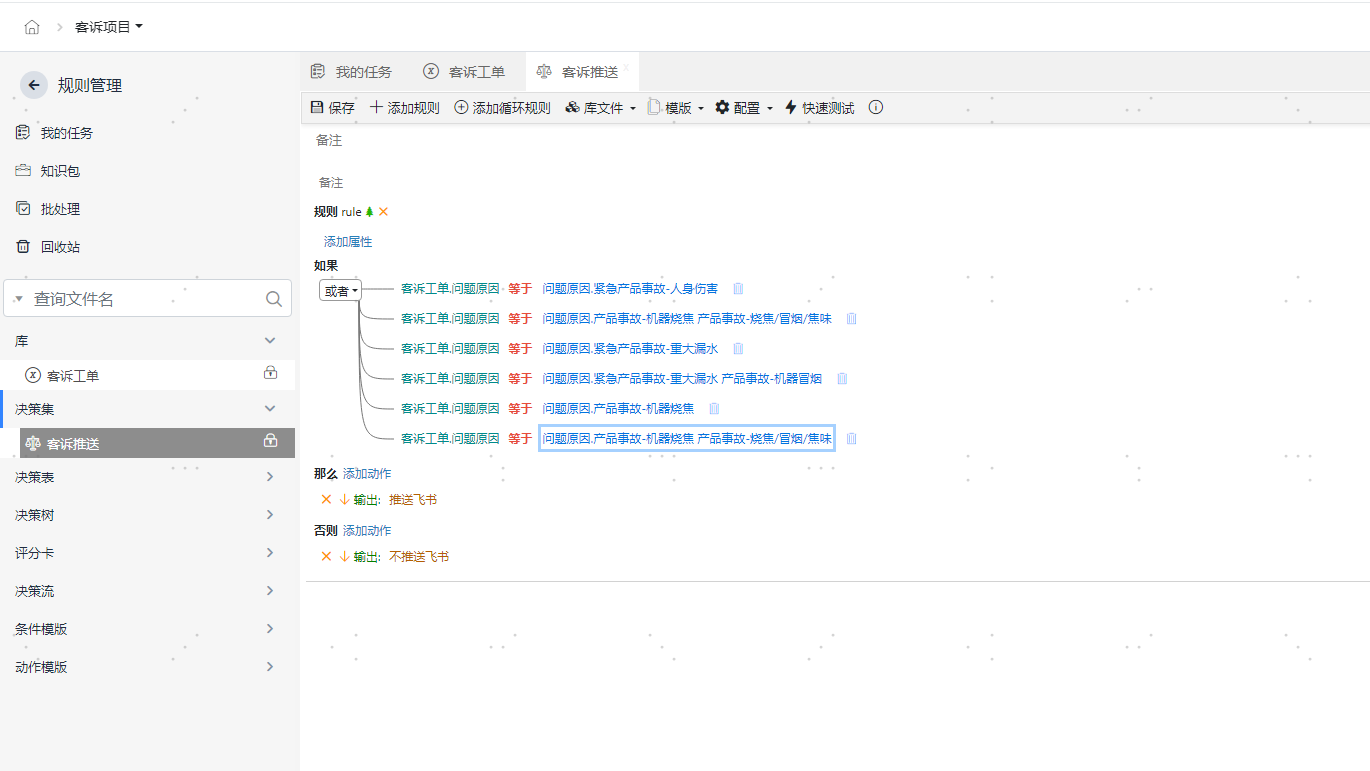
业务中台的每一个系统都专门维护了一个页面来维护配置项，配置项的作用就是把一些可能发生变化的数据进行单独处理。

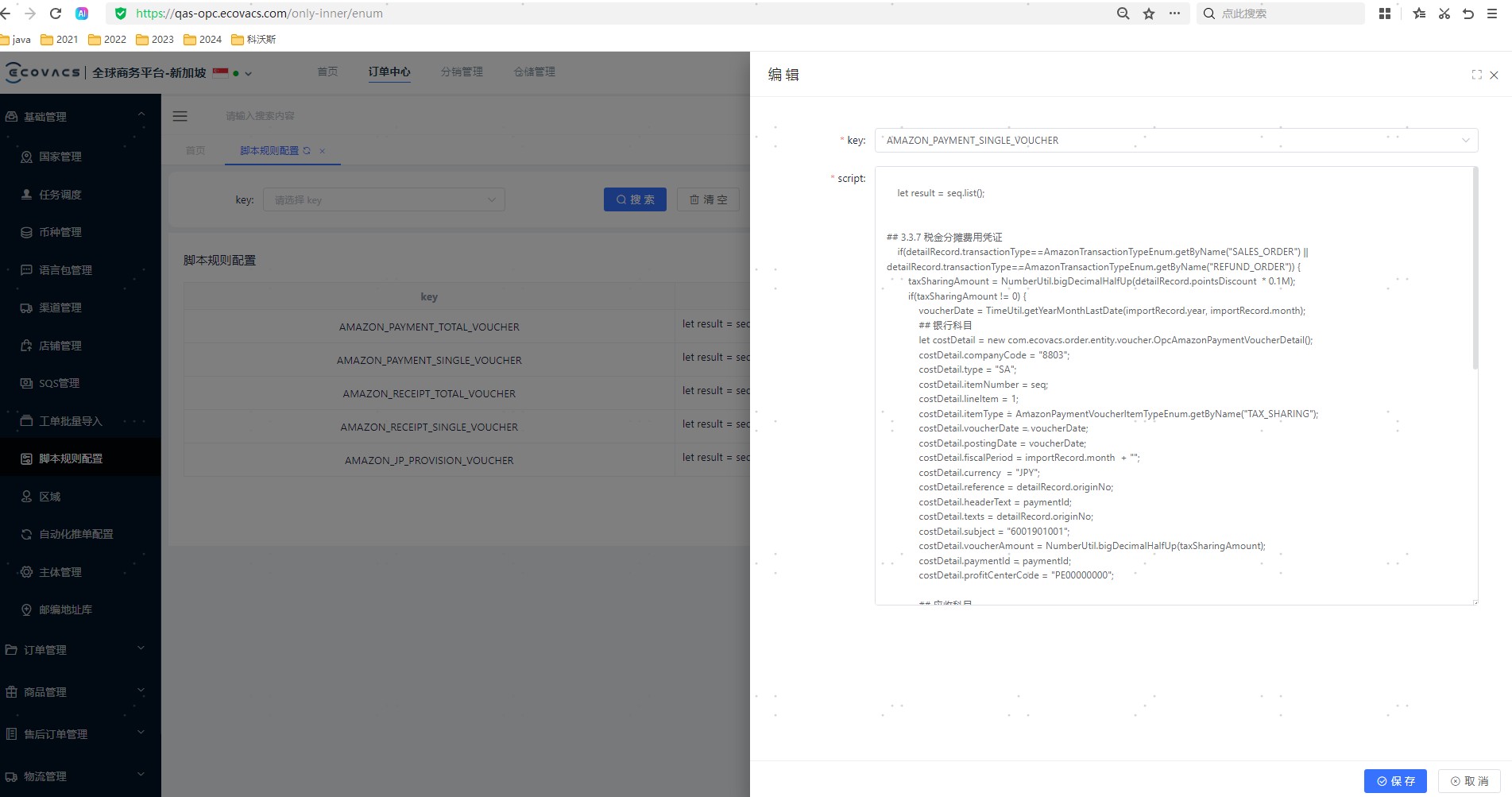


但是只有指标发生变化，我们的配置项才会发生作用，也就是说我们配置项的能力是对数值变化做出及时响应。所以说配置项的能力是有限的。

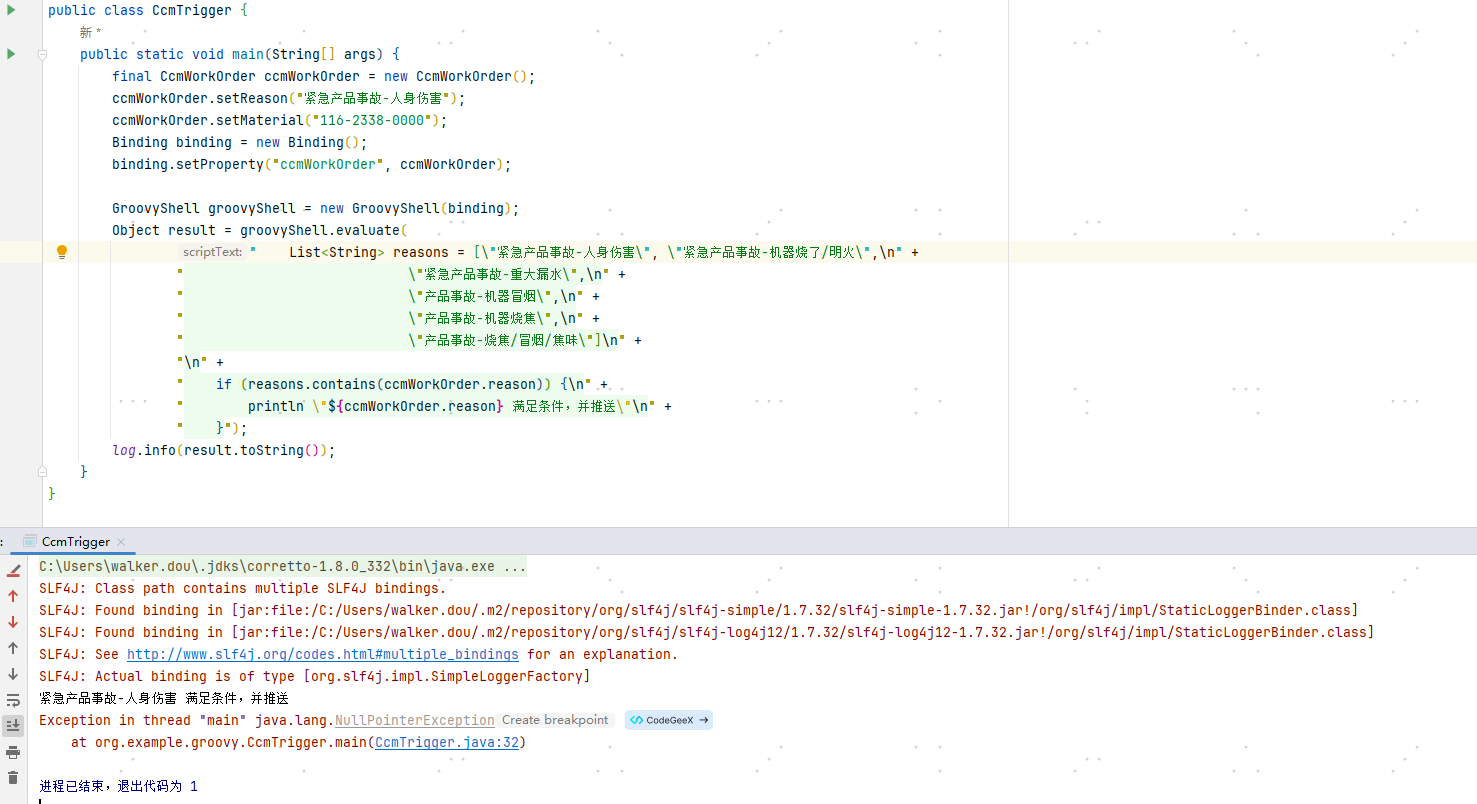
假设有一天，客诉工单推送飞书的条件发生了变化，不仅是需要投诉原因满足条件，还需要产品型号是T30,那我们之前维护的配置项就无法发挥作用了，我们又需要重复到‘开发-测试-发版’这一系列工作中了，开发需要时间，测试也需要时间，运维也需要时间。我们无法对业务变化做到及时响应，即使是这么小的一个业务变化。

我们开始思考是否有什么方式能对缩短响应时间，之前我们维护的是个配置项，我们改成维护一段脚本代码，我们在海外订单中心和解析平台也做过相似的事情，就是把容易发生变化的代码通过脚本语言来书写，当业务发生变化的时候直接维护脚本代码就可以，之前我们采用的aviator脚本语句来实现的。





这里我们选用groovy这种脚本语言，重新实现创建客诉工单推送飞书这段业务，



我们需要在java通过groovy引擎来调用groovy代码，groovy是一种脚本语言，特点就是很灵活，但是groovy过于灵活的语言特性，意味着一旦代码运行出错，很难发现问题，因为groovy中大量的语法糖。

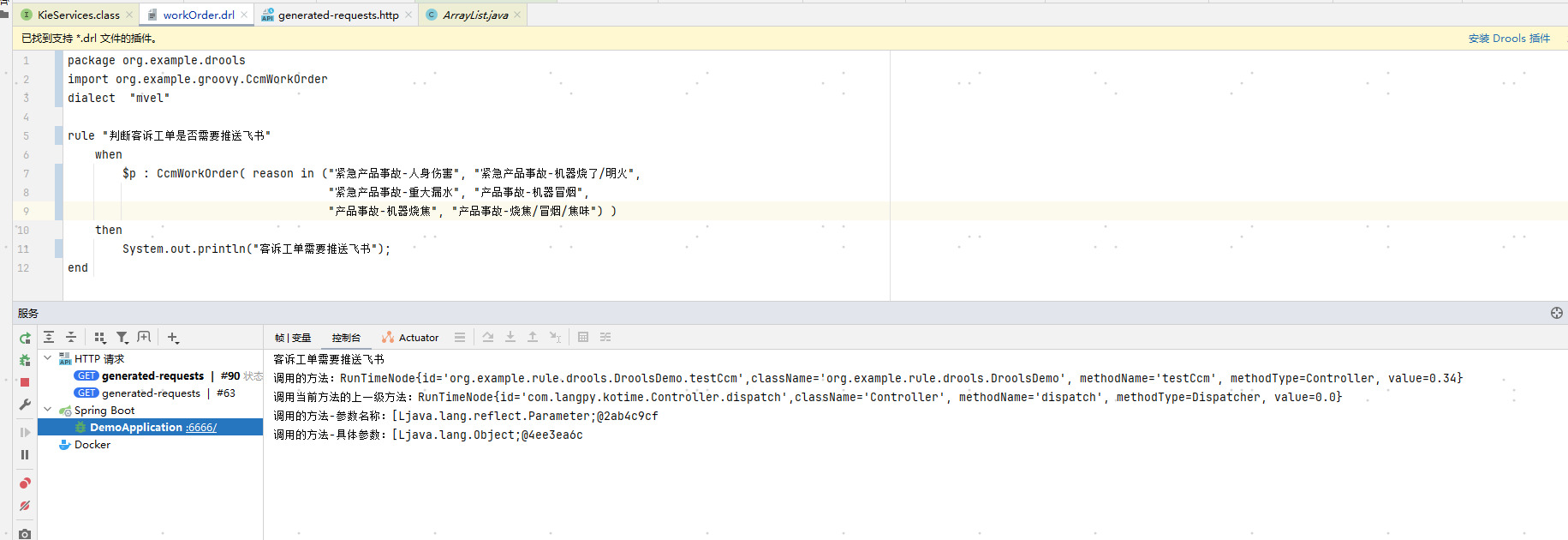
而且新学习一门编程语言都是需要很大成本的，无论是groovy还是lua，都会增加我们系统维护成本。所以我们想有没有更容易的语言，dsl或许是一种解决办法。Dsl(领域特定语言)

DSL是一种程序设计语言，因此它必须具备连贯的表达能力,但是只有在一个明确的小领域下，这种能力有限的语言才会有用。

我们继续采用drool这种规则引擎来优化我们的代码，

**Drools**（JBoss Rules ）具有一个易于访问企业策略、易于调整以及易于管理的开源业务规则引擎，符合业内标准，速度快、效率高。业务分析师或审核人员可以利用它轻松查看业务规则，从而检验是否已编码的规则执行了所需的业务规则。



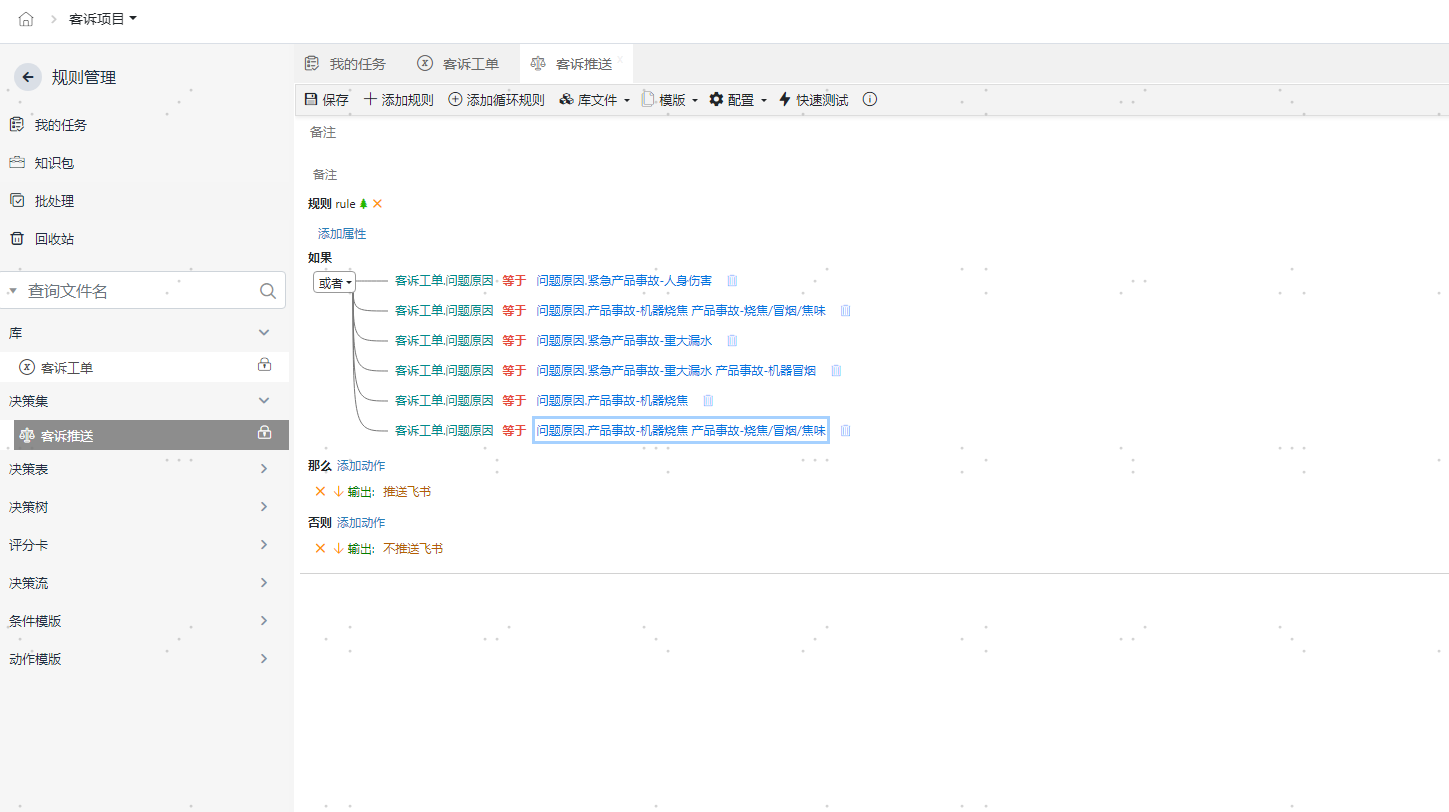


可以看出， drools语言要简单一些，而且在表现能力上drools也要比groovy强大许多,因为drools是专门用于规则引擎的dsl。但是dsl也需要学习成本，而且drools 我们很难对drools进行集中管理，即使集中管理，我们怎么对drools进行权限管理和监控那？因为drools仅仅是一个框架，并不是一个完备的系统。

在调研了常用的规则引擎框架之后，我们尝试去发现是否有一些功能相对丰富的规则引擎系统，这时urule进入了我们的视野。

URule是一款纯Java规则引擎，它可以运行在Windows、Linux、Unix等各种类型的操作系统之上；URule Pro的规则设计器采用业内首创的纯浏览器编辑模式，无须安装任何工具，打开浏览器即可完成复杂规则的设计与测试。

Urule







Urule规则的编写完全通过页面图形点击的方式构建，相对于groovy和drools开发难度大大降低，同时Urule作为一个系统还提供例如权限管理，快速测试，监控跟踪，版本迭代等等功能。