

A light gray world map serves as the background. Scattered across the map are approximately 15 circles of various colors: red, green, purple, orange, pink, blue, and yellow. Some circles are solid, while others have a dashed outline. A large red circle with a dashed white border is centered in the upper half of the image, containing the text '蘑菇街'.

蘑菇街

基于规则引擎的闭环质量平台



目录 Content

01

测试模型分析

02

输入输出与结果校验

03

规则引擎实现

04

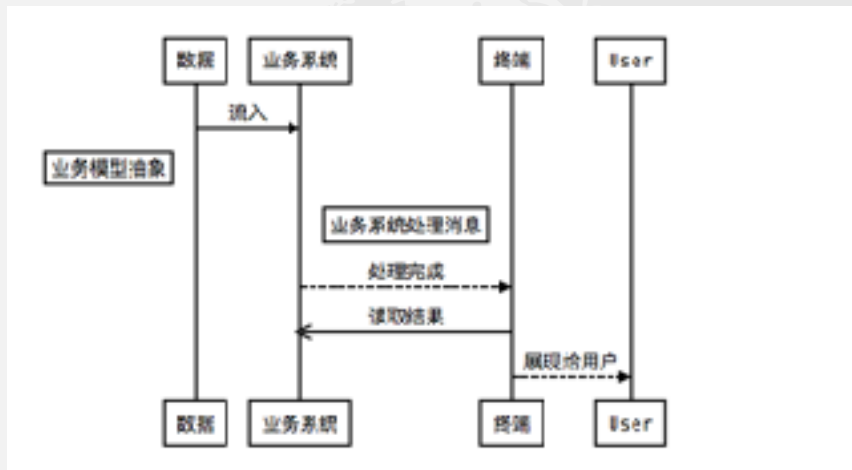
监控与报警

05

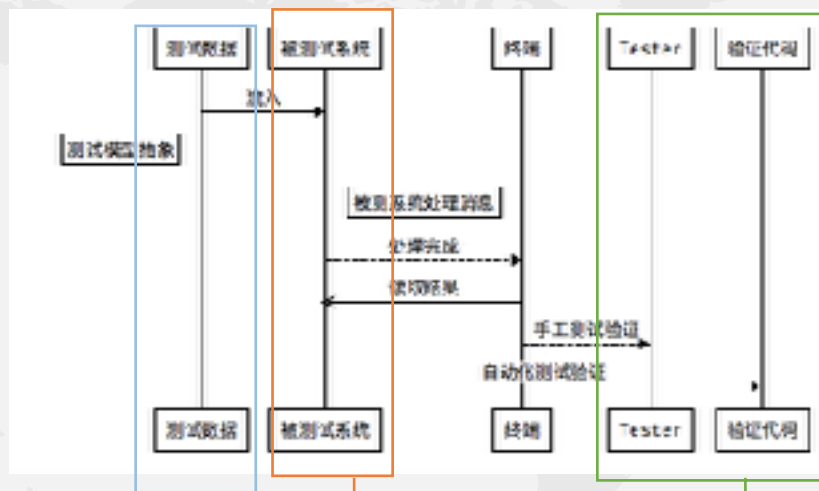
报表分析

业务模型 vs 测试模型

业务数据->业务系统/模块->业务结果展示



测试数据->被测试系统（业务逻辑）->结果输出/验证



1、根据业务逻辑编写测试用例

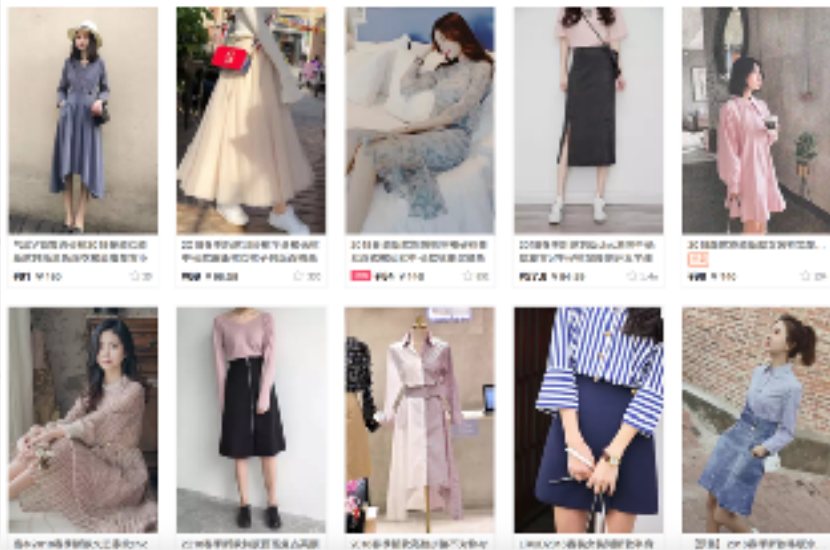
2、准备测试数据

3、结果验证

电商应用典型特点分析

业务特点分析：

- 1、后端数据流转复杂，涉及招商、促销、引擎等多个系统交互，会影响到价格、销量等敏感字段的正确性
- 2、线上数据多样性，包含了各种正常请求，XSS注入攻击，超长url等
- 3、人员熟练程度引发的运营排期、招商模版设置错误引发的问题





目录 Content

01

测试模型分析

02

输入输出与结果校验

03

规则引擎实现

04

监控与报警

05

报表分析

用户输入数据类测试数据源

- 1、搜索类：以线上数据为蓝本，每日清洗线上数据，并根据访问流量大小配比，每日自动更新约4000条新的请求数据
- 2、搜索类：自定义请求数据：用于特殊场景的验证及线上发布的快速回归



影响正确性的因素

- 1、引擎同步消息丢失，逻辑变更
- 2、应用层组装异常
- 3、异常输入，如用户构建的xss注入攻击

固定产品页/模块类测试数据源

- 1、固定页面透出类：定期刷新页面数据，及时感知到数据变更，保证能够第一时间发现页面的异常（如显示不完整，空窗）等
- 2、所有引用固定模块的页面数据检查，如大促类模块的自动扫描录入



所有引用某模块的页面列表

商品名称	品牌	规格	价格	库存
元气少女穿搭经	元气少女	1000g	100.00	1000
元气少女穿搭经	元气少女	1000g	100.00	1000
元气少女穿搭经	元气少女	1000g	100.00	1000
元气少女穿搭经	元气少女	1000g	100.00	1000
元气少女穿搭经	元气少女	1000g	100.00	1000
元气少女穿搭经	元气少女	1000g	100.00	1000
元气少女穿搭经	元气少女	1000g	100.00	1000
元气少女穿搭经	元气少女	1000g	100.00	1000
元气少女穿搭经	元气少女	1000g	100.00	1000
元气少女穿搭经	元气少女	1000g	100.00	1000



影响正确性的因素：

- 1、商品排期空窗
- 2、招商模版选择错误
- 3、引擎消息同步异常
- 4、其它因素

结果输出片段

特点：

- 1、不同的接口返回数据格式有差异
- 2、相同的接口不同版本返回结构与字段有差异



校验规则分类

1、通用型规则

- 1) title、image、url等关键字段非空
- 2) price与引擎、DUMP、招商报名的正确性比对
- 3) 销量、收藏数正确性检查

2、特定业务逻辑的校验

- 1) 搭配列表商品熟需大于10
- 2) 商品置顶功能生效
- 3) 秒杀商品存在秒杀价字段等

3、如何灵活的校验？

- 1) 能够适配不同结构的返回结果
- 2) 规则与请求能够匹配



目录 Content

01

测试模型分析

02

输入输出与结果校验

03

规则引擎实现

04

监控与报警

05

报表分析

规则引擎实现：校验规则与请求匹配

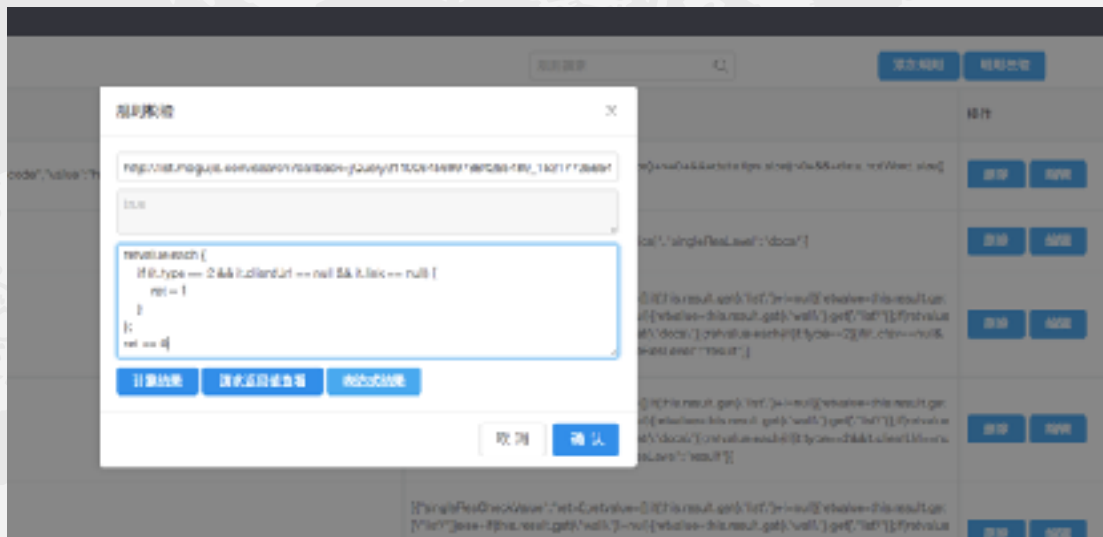
校验规则	入参示例	正则校验规则
返回数据	<pre>{ "parameter": "cKey", "value": "cse-csf", "parameter": "trid", "value": "20000811", "parameter": "trid", "value": "110000", "parameter": "screen_production", "value": "1" }</pre>	<pre>0?singleResCheckValue"first=0;resValue=[- +]";this.result.get("list")=null();[- + resValue=this.result.get("list")];(else-0)this.result.get("wall")=null();(else=ch.sresult.get("wall"),get("list")();(firstValue=null();[- + resValue=ch.sresult.get("wall"),get("does")();(else=ch.sresult.get("tridItem")="r8dq";["singleResLevel":"result"];</pre>
请求参数包括cKey=V Ublv的请求结果会匹配本条校验规则	<pre>{ "parameter": "cKey", "value": "V Ublv" }</pre>	<pre>0?singleResCheckValue"first=0;resValue=[- +]";this.result.get("list")=null();[- + resValue=this.result.get("list")];(else=ch.sresult.get("wall")=null();[- + resValue=this.result.get("wall"),get("list")];(else=ch.sresult.get("tridItem")="r8dq";["singleResLevel":"result"];</pre>
在等号左侧验证不为空	<pre>{ "parameter": "cKey", "value": "APVfZz", "parameter": "version", "value": "419430" }</pre>	<pre>0?singleResCheckValue"first=0;resValue=[- +]";this.result.get("list")=null();[- + resValue=this.result.get("list")];(else=ch.sresult.get("wall")=null();[- + resValue=this.result.get("wall"),get("list")];(else=ch.sresult.get("tridItem")="r8dq";["singleResLevel":"result"];</pre>
返回数据不为空	<pre>{ "parameter": "trid", "value": "10050000" }</pre>	<pre>0?singleResCheckValue"first=0;resValue=[- +]";this.result.get("list")=null();[- + resValue=this.result.get("list")];(else=ch.sresult.get("wall")=null();[- + resValue=this.result.get("wall"),get("list")];(else=ch.sresult.get("tridItem")="r8dq";["singleResLevel":"result"];</pre>
返回数据不为空	<pre>{ "parameter": "trid", "value": "10050000" }</pre>	<pre>0?singleResCheckValue"first=0;resValue=[- +]";this.result.get("list")=null();[- + resValue=this.result.get("list")];(else=ch.sresult.get("wall")=null();[- + resValue=this.result.get("wall"),get("list")];(else=ch.sresult.get("tridItem")="r8dq";["singleResLevel":"result"];</pre>

规则引擎实现：引擎校验逻辑

基于Groovy的表达式求值



规则验证小工具



规则配置示例

规则代码示例

```
ret = 0;
retvalue = [];
if (this.result.get("list") != null) {
    retvalue = this.result.get("list")
} else if (this.result.get("wall") != null) {
    retvalue = this.result.get("wall").get("list")
};
if (retvalue == null) {
    retvalue = this.result.get("wall").get("docs")
};
retvalue.each {
    if (it.type == 2 && it.clientUrl == null &&
it.link == null) {
        ret = 1
    }
};
ret == 0
```

规则配置列表

规则ID	规则名称	规则内容	操作
1	规则1	规则1内容	编辑 删除
2	规则2	规则2内容	编辑 删除
3	规则3	规则3内容	编辑 删除
4	规则4	规则4内容	编辑 删除
5	规则5	规则5内容	编辑 删除

某业务失败原因分维度统计

业务标示维度的错误统计

□	所属主体	所属地区	Shop	所属经营数	运营	支付方式	所属品牌
□	USA	2018-11-18-2019-11-15	shop1	100	NA	NA	美国品牌
□	USA	2018-11-18-2019-11-2	shop2	100	NA	NA	美国品牌
□	USA	2018-11-18-2019-11-3	shop3	100	NA	NA	美国品牌
□	USA	2018-11-18-2019-11-3	shop4	100	NA	NA	美国品牌
□	USA	2018-11-18-2019-11-3	shop5	100	NA	NA	美国品牌
□	USA	2018-11-18-2019-11-3	shop6	100	NA	NA	美国品牌
□	USA	2018-11-18-2019-11-3	shop7	100	NA	NA	美国品牌
□	USA	2018-11-18-2019-11-3	shop8	100	NA	NA	美国品牌
□	USA	2018-11-18-2019-11-3	shop9	100	NA	NA	美国品牌
□	USA	2018-11-18-2019-11-3	shop10	100	NA	NA	美国品牌

错误类型维度的统计

[illegible]



目录 Content

01

测试模型分析

02

输入输出与结果校验

03

规则引擎实现

04

监控与报警

05

报表分析

异常数据监控/报警

页面数据为空



价格/折扣为0



IM/短信报警

L [redacted] 以程序: 品牌店铺销量校对结果如下,发现店铺checkid: [redacted]
ShopName [redacted] 方城路店---StartTime 2018-10-14 00:00:00---EndTime:
2018-10-16 23:59:59疑似未正常卖出,请确认!

[redacted] 智家宝贝(昵称: [redacted]) - 快被针对链接口掉了,请检查,详情可以到页面看
看<https://antimogujia.com>

上下游链路监控/报警

业务端/引擎/dump异常报表

商品名称	活动时间	商品ID	商品名称	商品ID	商品名称	商品ID	商品名称	商品ID	商品名称	商品ID	商品名称	商品ID	商品名称	商品ID
商品名称	2018-10-10 14:11			2000	2000	2000	2000	2000	2000	2000	2000	2000	2000	2000
商品名称	2018-10-10 14:11			2000	2000	2000	2000	2000	2000	2000	2000	2000	2000	2000
商品名称	2018-10-10 14:11			2000	2000	2000	2000	2000	2000	2000	2000	2000	2000	2000

报警信息

-dump-engine 价格比对,活动阶段:日常,耗时1171s,共比对 1787个产品,其中35个失败,详情请查看:



目录 Content

01

测试模型分析

02

输入输出与结果校验

03

规则引擎实现

04

监控与报警

05

报表分析

各维度分析报表

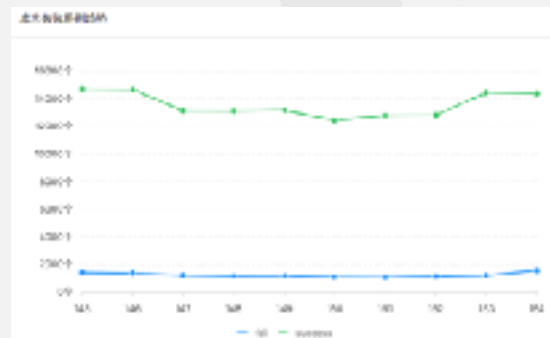
性能趋势



耗时排名与分析

排名	项目	成功率	执行用例数	最长时间
名: 参数: ("appFlat": "m", "activityLaunch": "B")				
2		100.00%	145	192ms
3		100.00%	85	85ms
4		100.00%	42	114ms
5		100.00%	105	169ms

线上回放数据走势分析（每日更新）





谢

谢

聆

听