

关于 API 文档驱动研发模式的探索与实践

陈子煜

得物商家前端负责人

精彩继续！ 更多一线大厂前沿技术案例

📍 北京站

QCon

全球软件开发大会

时间：12月18-20日

地点：北京·四季酒店

扫码查看大会
详情>>



📍 北京站

GITC

全球大前端技术大会

时间：12月19-20日

地点：北京·四季酒店

扫码查看大会
详情>>



📍 北京站

ArchSummit

全球架构师峰会

时间：12月25-26日

地点：北京·海航万豪酒店

扫码查看大会
详情>>



自我介绍



- 2015毕业于Rutgers University
- 曾就职 Epic、字节跳动、蚂蚁集团
- 得物商家前端负责人， Mooncake平台项目PM

大纲

- 迭代中的效能瓶颈分析
- 文档驱动流程相关介绍
- 文档驱动下前端的研发流程
- 后续计划

大纲

- 迭代中的效能瓶颈分析
- 文档驱动流程相关介绍
- 文档驱动下前端的研发流程
- 后续计划

前端迭代流程



沟通成本

- 需求确认、技术方案设计需要反复沟通确认
- 研发过程中，方案的变更需要及时沟通

信息互换、变更触达

依赖

- 前端应用数据驱动，据”来渲染页面依赖服务端返回的“数
- 测试过程中依赖“造数工具”复现场景进行回归

数据依赖

流程痛点

前端位于链路的最上游，因此需要付出最多的**协作**成本

- 数据依赖服务端，在所有下游服务开发完成后，才能获取到数据
- 开发过程中，任意下游服务发生变动，都有可能影响到前端应用
- 联调过程中严重依赖服务端、测试造数来进行场景的调试
- 出现问题，测试永远先从前端应用开始排查
-



大纲

- 迭代中的效能瓶颈分析
- 文档驱动流程相关介绍
- 文档驱动下前端的研发流程
- 后续计划

API Mandate

能不能造个轮子解决协作问题？如果可以，基于什么来造？

Jeff Bezos' API mandate

1. *Data and Capabilities must be exposed through APIs*
2. *Team Communications must be through APIs*
3. *There can be no side channels/shortcuts*
4. *Technology choice is secondary*
5. *APIs must be externalizable*

API文档驱动



基于API文档在Mooncake平台完成协同，各职能团队独立完成工作

API文档驱动

生产端

通过idea插件、cli等工具快速生成API文档

消费端

基于API文档完成文档驱动研发流程

生态

向内部其他平台、工具提供接口文档信息



API文档信息

基于文档完成协作，需要解决哪些问题？

- 信息互换
- 变更触达
- 数据依赖



API文档信息

为什么写

- 该接口本次迭代相关的如PRD链接等需求信息
- 从研发协同平台同步和该需求相关的研发人员信息，接口发生变更时能够及时触达相关人

接口列表 分类列表 × 现货商品详... × : local

文档 编辑 运行 类型声明 Mock 测试用例

分类列表 ★ 5.0分

POST /mooncake/v2/openApi/category/cascadeList • 设计中

更新时间: 2022-11-16 17:25:43 修改人: 匿名 分类: 项目

网关配置: d1 t1 pre prd ... 迭代版本号: 5.6.0 迭代需求: Mooncake迭代 PRD文档: 品牌专供直发流程优化

请求参数

参数名	位置	类型	是否必须	示例	说明
projectId	query	number	否	5104	项目id
isExport	query	string	否	true	是否导出

API文档信息

写什么

接口的基础信息字段，包括但不限于：

- 出入参数
- 字段备注
- 文档备注
- 文档评论
-

The screenshot displays the Mooncake API documentation interface. The main content area shows a POST endpoint: `/mooncake/v2/openApi/category/cascadeList`. The endpoint is marked as "设计中" (Designing) and has a 5.0 rating. It includes a "相关责任人" (Responsible Person) dropdown menu with "后端: 陈子煜" (Backend: Chen Ziyu), "前端: 陈子煜" (Frontend: Chen Ziyu), and "测试: 陈子煜" (Test: Chen Ziyu). The interface also shows a "请求参数" (Request Parameters) table and a "请求Body" (Request Body) table.

参数名	位置	类型	是否必须	示例	说明
projectId	query	number	否	5104	项目id
isExport	query	string	否	true	是否导出

参数名	类型	是否必须	默认值	其他信息	备注
code	number	非必须			状态码
status	number	非必须			状态
+ data	object	非必须			
auto	boolean	非必须			
message	string	非必须			
traceld	string	非必须			

参数名	类型	是否必须	默认值	其他信息	备注
code	number	非必须			状态码
status	number	非必须			状态
+ data	object	非必须			
auto	boolean	非必须			
message	string	非必须			提示信息
traceld	string	非必须			请求追踪id

API文档信息

写什么

以迭代维度记录接口的历史版本信息，包括但不限于：

- 变更原因
- 变更时间
- 变更人
-



The screenshot shows a web interface for API management. On the left, there's a sidebar with 'mooncake' selected. The main area shows a search bar and a list of API endpoints. A modal window titled '历史变更管理' (History Change Management) is open, displaying a table of version changes.

迭代版本	开始时间	结束时间	迭代状态
+ 5.04	10月14日 00:00	2022-10-27	筹备
5.05	10月28日 00:00	2022-11-17	发布
编辑人	变更时间	变更说明	操作
陈光远	11月03日 20:10	增加了订单优惠信息	
陈光远	10月30日 17:17	修复订单金额计算问题	对比
陈光远	10月10日 17:34	增加了优惠券发放人	对比
+ 5.06	11月11日 00:00	2022-11-24	筹备

API文档信息

写什么

支持对历史版本diff，快速定位到变更的字段，主要应用于前后端信息交换的场景

The screenshot shows the 'mooncake' API documentation interface. The left sidebar contains a navigation menu with categories like '商家订单', '商家管控', 'mooncake', '项目', and '关联了用例的接口'. The main content area displays details for a specific endpoint: `POST /v2/openApi/category/cascadeList`. It includes a '版本说明' (Version Description) section with '当前版本' (Current Version) 5.06 and '历史版本' (Historical Versions) 5.05. Below this is the '请求参数' (Request Parameters) section, which contains a table of query parameters:

参数名	类型	是否必须	说明
projectId	number	否	项目id
isExport	string	否	是否导出

Below the request parameters is the '返回响应' (Return Response) section, which contains a table of response parameters:

参数名	类型	必须	默认值	说明	其他信息
code	number	否		状态码	
status	number	否		状态	
data	object	否			
list	object []	否			item 类型: object
value	string	是		分类id	
label	string	是		分类名	
children	object []	是		子节点	item 类型: object
value	string	是			
label	string	是			
auto	boolean	否			
message	string	否		提示信息	
traceld	string	否		请求追踪id	

API文档信息

要怎么用

基于单测用例即文档的理念，让用例当作文档的一部分。每一条用例即该接口的一个使用场景

平台同时也支持用例的执行、结果查看等

The screenshot shows the 'mooncake' API testing interface. The left sidebar lists API endpoints under categories like '商家订单' and '商家管控'. The main area displays a '测试用例列表' (Test Case List) table with columns for '用例名称', '优先级', '创建人', '创建时间', '用例状态', '运行结果', and '操作'. Three test cases are listed, all with a status of '已上线' and '成功'.

	<input type="checkbox"/>	用例名称	优先级	创建人	创建时间	用例状态	运行结果	操作
+	<input type="checkbox"/>	现货订单正向流程	P3	zhangchun	2022-08-02 10:02:46	已上线	● 成功	运行 运行结果
+	<input type="checkbox"/>	现货订单逆向流程	P3	zhounuo	2022-08-02 18:10:00	已上线	● 成功	运行 运行结果
+	<input type="checkbox"/>	现货商品正向流程	P3	zhounuo	2022-08-03 20:40:34	已上线	● 成功	运行 运行结果

API文档信息

要怎么用

用例详情描述了该场景的具体信息，包含但不限于：

- 用例负责人信息
- 用例的断言信息
- 接口的上下游调用链路信息
-

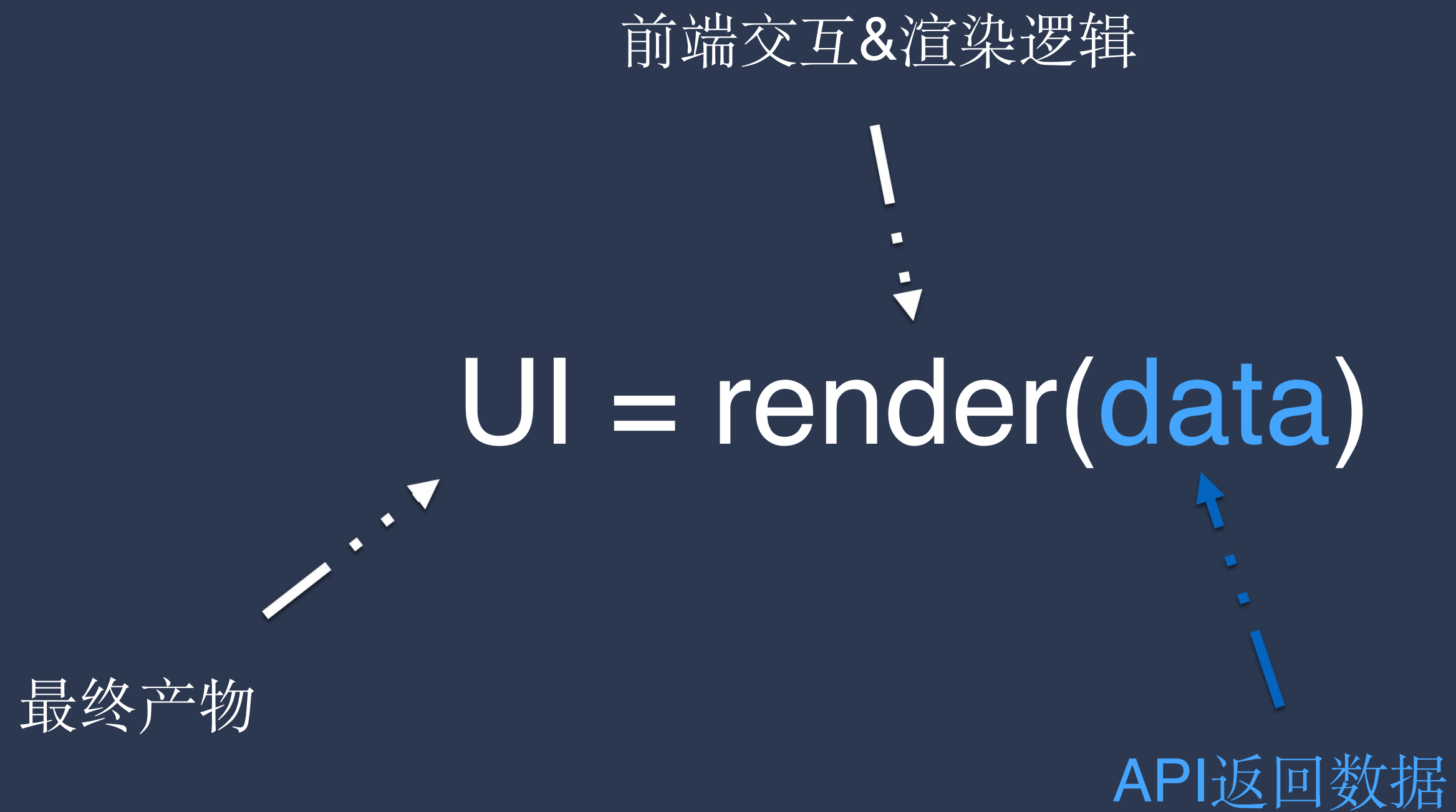
The screenshot displays an API documentation interface for a GET endpoint. The breadcrumb trail includes '接口列表', '现货商品详...', and '现货订单正...'. The endpoint is identified as 'GET /v1/order/coupon' with a status code of 200, a timeout of 10s, and a modification time of 2022-08-03 11:16:48. The '用例步骤' (Use Case Steps) section lists three steps: 'POST 新建订单', 'GET 查询优惠' (highlighted), and 'POST 提交订单'. The '参数设置' (Parameter Settings) section shows a table with columns for '字段' (Field), '值' (Value), and '描述' (Description), which is currently empty with a '暂无数据' (No data) message. The '断言设置' (Assertion Settings) section contains a table with columns for '断言名称' (Assertion Name), '断言表达式' (Assertion Expression), '断言方法' (Assertion Method), '期望值' (Expected Value), and '期望值类型' (Expected Value Type). The table lists three assertions: '断言code码' (assertEqual, 200, int), '校验商卡选项不为空' (assertIsNotEmpty, string), and '校验白名单用户' (assertNotInclude, 收藏, string). A page number '1' is visible at the bottom right.

断言名称	断言表达式	断言方法	期望值	期望值类型
断言code码	\$.code	assertEqual	200	int
校验商卡选项不为空	\$.data.list	assertIsNotEmpty		string
校验白名单用户	\$.data.list[*].tabName	assertNotInclude	收藏	string

大纲

- 迭代中的效能瓶颈分析
- 文档驱动流程相关介绍
- 文档驱动下前端的研发流程
- 后续计划

前端构成



前端研发流程

现有流程



前端开发、调试都依赖于服务端数据



- 前端迭代效率低
- 联调效率低，压力大
- 交付质量差，跟测效率低

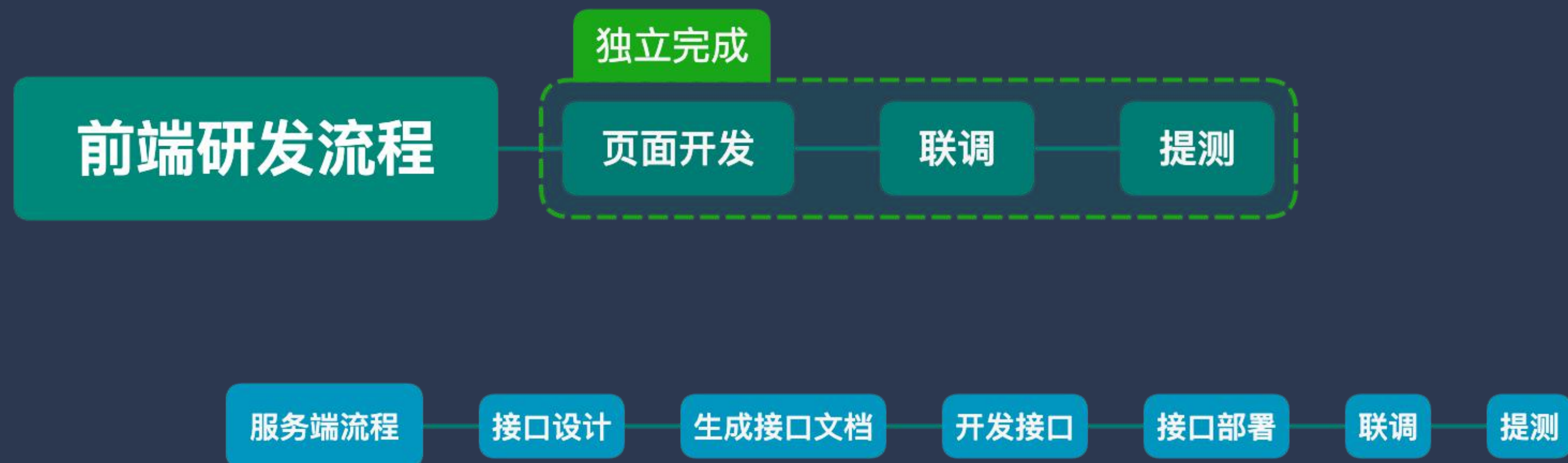
前端研发流程

期望流程

在进入实际code之后，前后端完全解藕，能够独立的完成需求的开发



解决对于服务端数据依赖问题



文档生态 – API Mock

通过mooncake对接口进行mock，分为入参校验、数据返回两种mock形式



入参校验mock

POST /h5/fes/swiz/get-swiz-batch-export

* 场景名称: Mock数据 (二选一): 静态JSON数据 动态JS 开启参数校验

参数名	比较	类型	参数值	操作
orderId <input type="text" value=""/>	等于 <input type="text" value=""/>	string <input type="text" value=""/>	<input type="text" value="请输入"/>	删除
+ 添加一行数据				

文档生态 – API Mock

数据Mock

核心在于降低造数的成本，Mooncake基于不同场景提供了多种造数方式：

- 根据定义自动生成 – 新接口
- 抓包 – 现有接口
- 手动编辑

The screenshot displays the '实时抓包' (Real-time Packet Capture) interface. On the left, there is a search bar and a list of captured APIs with checkboxes and colored buttons (GET, POST) indicating their methods. On the right, there is a text input for the scenario name and a code editor showing a JSON response structure. At the bottom right, there are buttons for '取消' (Cancel) and '生成Mock场景' (Generate Mock Scenario).

```
1 {
2   "url": "/api/v1/h5/biz/brand/order/checkMerchantMeetBrandPermission",
3   "methodType": "get",
4   "response": {
5     "code": 200,
6     "msg": "success",
7     "data": [
8       {
9         "orderPermissionDesc": "品牌直发",
10        "isPermission": 1
11      }
12    ],
13    "status": 200
14  }
15 }
```


文档生态 – API Mock

Mock场景

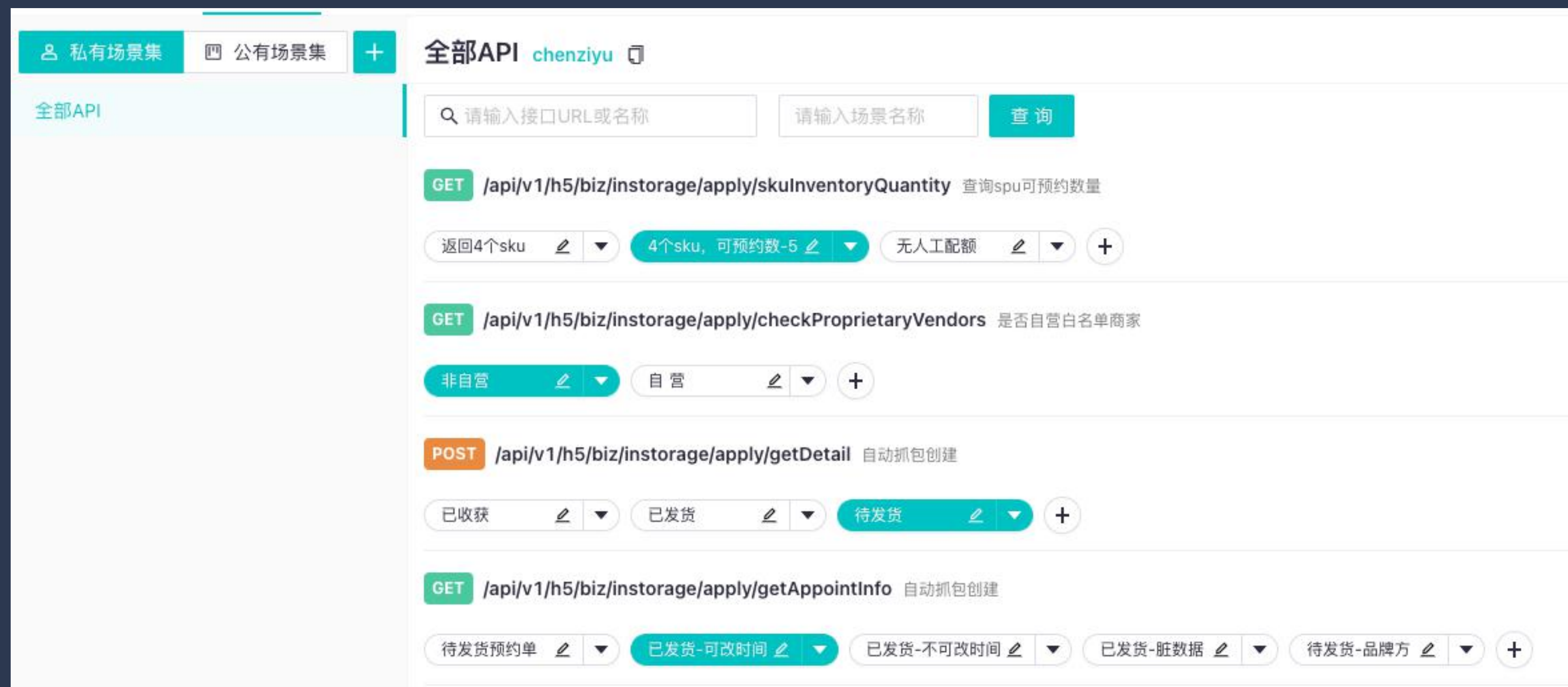
通过各接口多场景的排列组合，解决调试过程中各分支链路的自测问题

Mock场景集

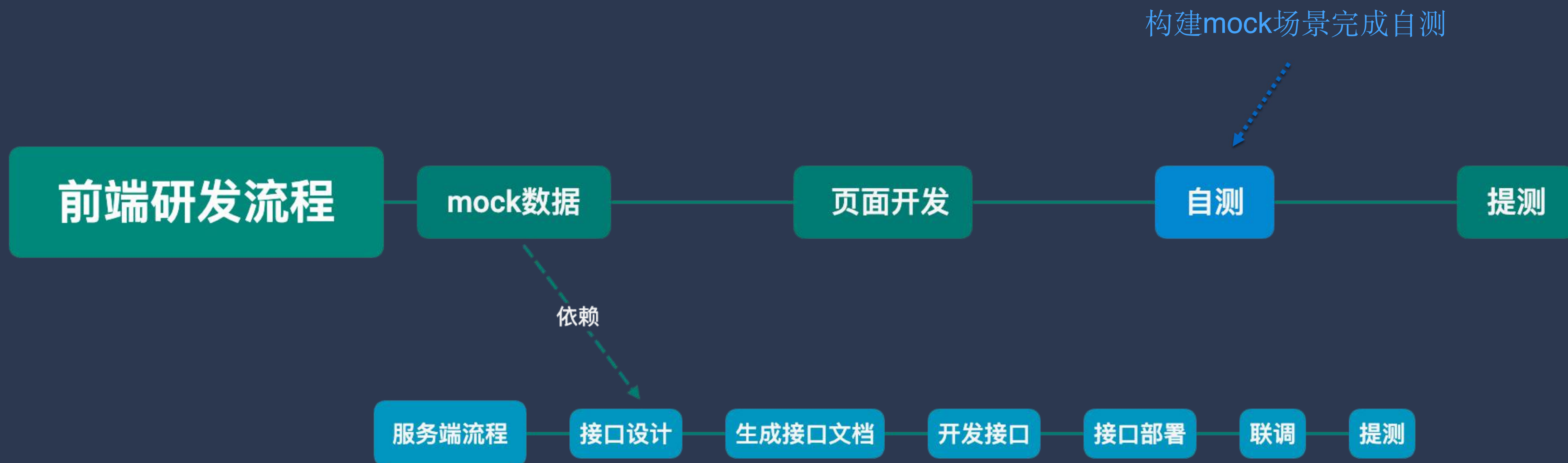
解决同一项目多人协作时的数据冲突问题



前端自测代替联调



前端如何文档驱动



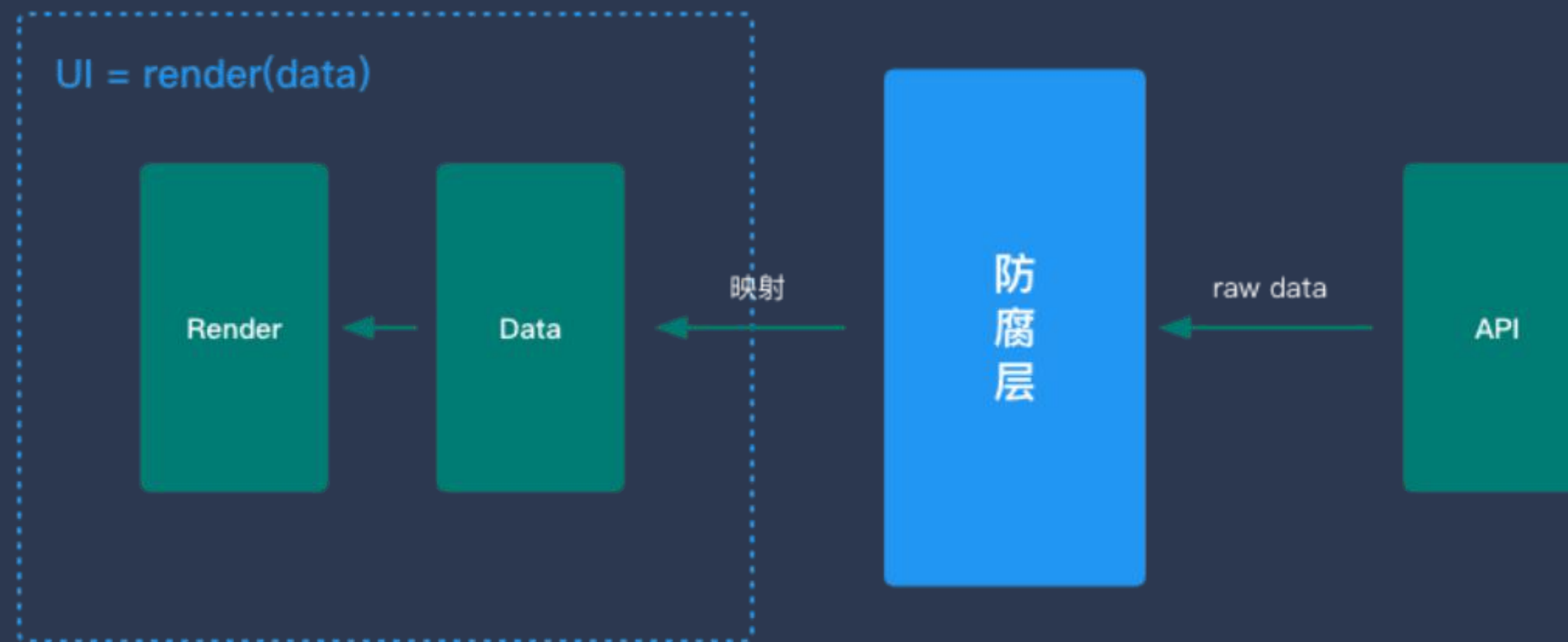
Mock数据的生产依然依赖于服务端接口定义

前端如何文档驱动

防腐层(ACL)

Implement a façade or adapter layer between different subsystems that don't share the same semantics. This layer translates requests that one subsystem makes to the other subsystem. Use this pattern to ensure that an application's design is not limited by dependencies on outside subsystems. This pattern was first described by Eric Evans in *Domain-Driven Design*.

- Azure Docs



前端如何文档驱动

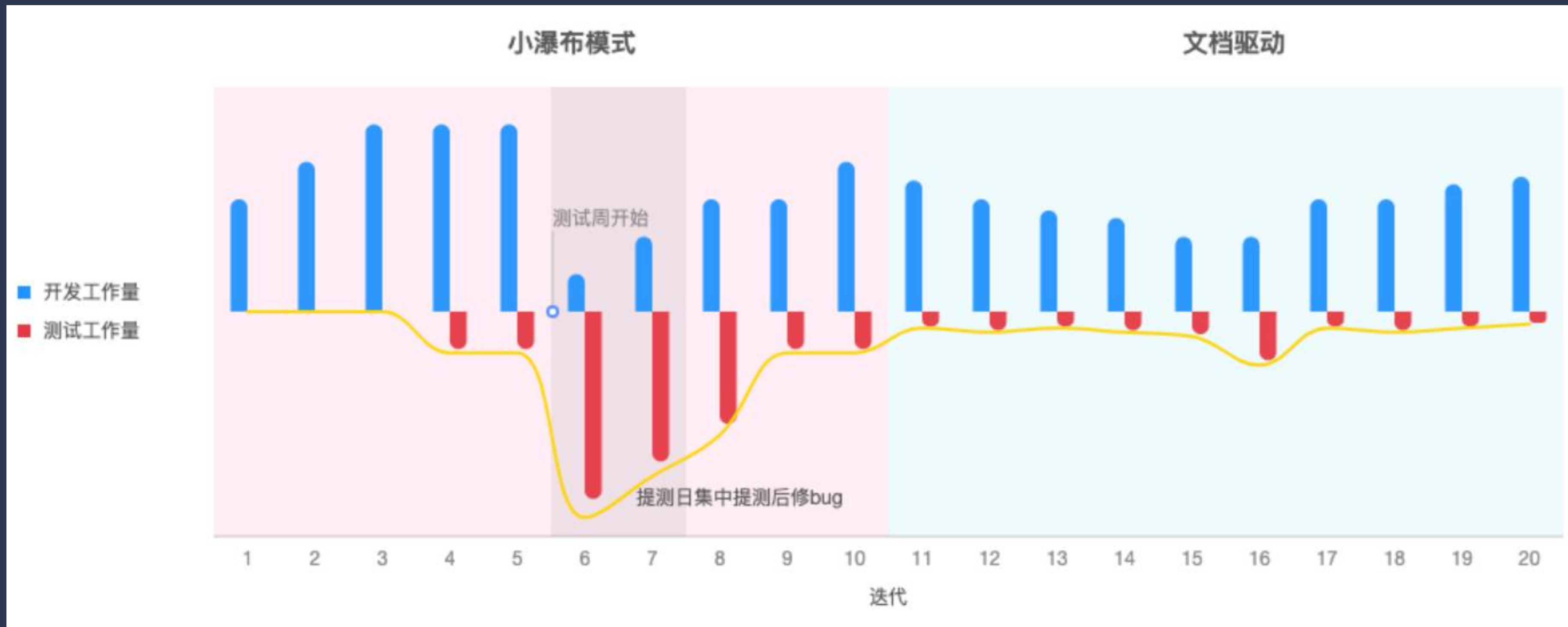


- 利用mooncake通知维护防腐层
- 利用mock数据自测代替联调



前端独立完成迭代流程

前端研发流程变化



- ~~联调风险~~
- ~~时间碎片化~~
- ~~接口依赖~~

落地成果

迭代效率

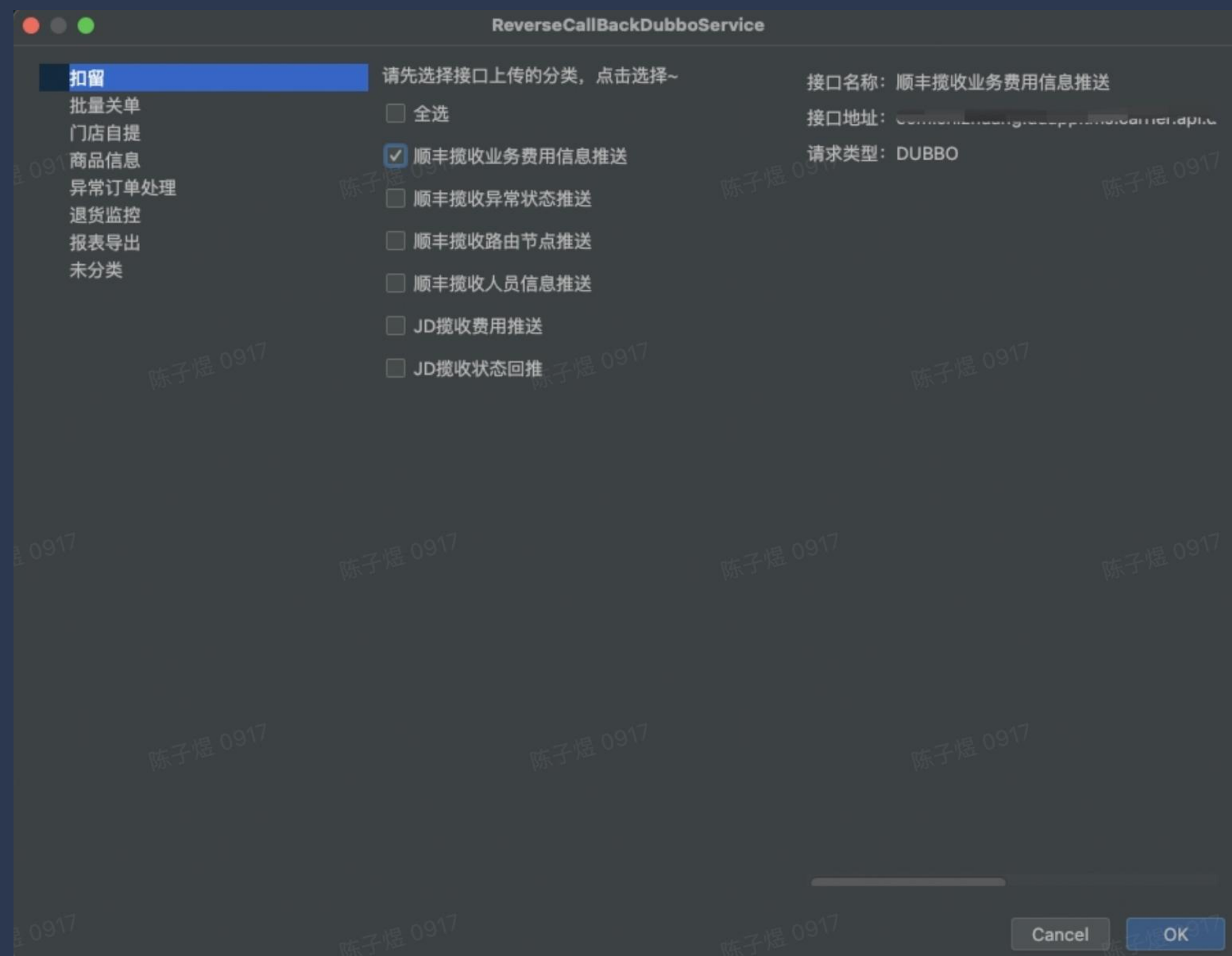
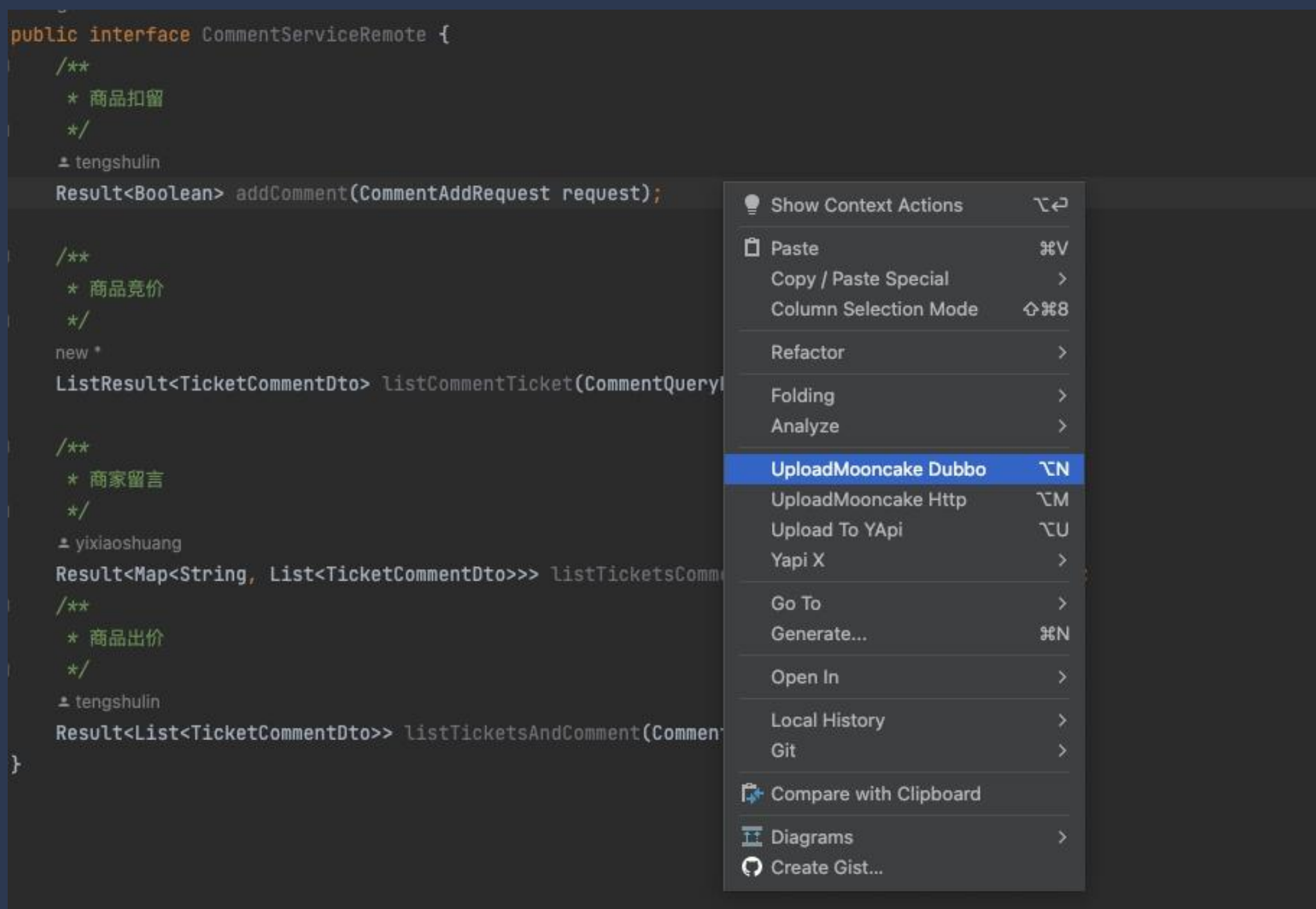


缺陷引入率



不止于前端 - 文档生产

基于IDEA插件快速上传接口



不止于前端 - 服务端文档驱动

Mooncake文档代替飞书文档

审核导出优化

请求URL: /api/v1/h5/order-cx/fakeDelivery/createExportTask

接口:
com.shizhuang.duapp.trade.order.cx.api.definition.backend.FakeDeliveryApi#createExportTask

// TODO 评估一下导出的上限, 做好限制

1. 根据筛选条件导出

- 入参新增导出条件 (黄色背景为新加入参字段):

```
1 {
2   "fakeProcessType": 2,
3   "startDeliveryTime": "2022-07-21",
4   "endDeliveryTime": "2022-07-26",
5   "subOrderNo": "111",
6   "sellerId": 222,
7   "isDeductDeposit": true,
8   "fakeDeliveryStatus": 3
9 }
```

- 校验endDeliveryTime大于startDeliveryTime,
- createTime控制在半年内

2. 导出增加 履约模式、商家名称字段 (公司名称)、订单状态 字段



trade-hogwarts-interfac...

接口列表 报告导出任务...

文档 编辑

报告导出任务创建 ☆--

DUBBO com.shizhuang.hogwarts.api.report.ReportApi.createExportTask 开发中

更新时间: 2022-08-24 13:49:05 修改人: 胡志敏 分类: 报告导出 服务分组: com.shizhuang.hogwarts.api.report

迭代版本号: 5.1.0 迭代需求: 品牌直发-虚假发货交易后台审核优化 PRD文档: 品牌直发-虚假发货交易后台审核优化

备注

5.1.0
目前导出订单不支持商家发货时间以及订单列表状态筛选的, 后导出全量的, EXCEL非常大, 电脑卡死严重。
目前已经关闭的订单无法在页面中搜索和筛选找到, 出现问题订单时查找比较艰难。

5.0.0
新增导出功能

请求参数

参数名	位置	类型	是否必须	示例	说明
Content-Type	header		否		

请求Body

```
1- {
2-   "com.shizhuang.hogwarts.api.report.ReportApi.createExportTask": {
3-     "fakeProcessType": "Integer",
4-     "startDeliveryTime": "String",
5-     "endDeliveryTime": "String",
6-     "subOrderNo": "Long",
7-     "sellerId": "Integer",
8-     "isDeductDeposit": "Boolean",
9-     "fakeDeliveryStatus": "Integer"
10-  }
11- }
```


不止于前端 - 服务端文档驱动流程

提供Dubbo/GRPC Mock能力，解决服务端跨团队协作效率问题



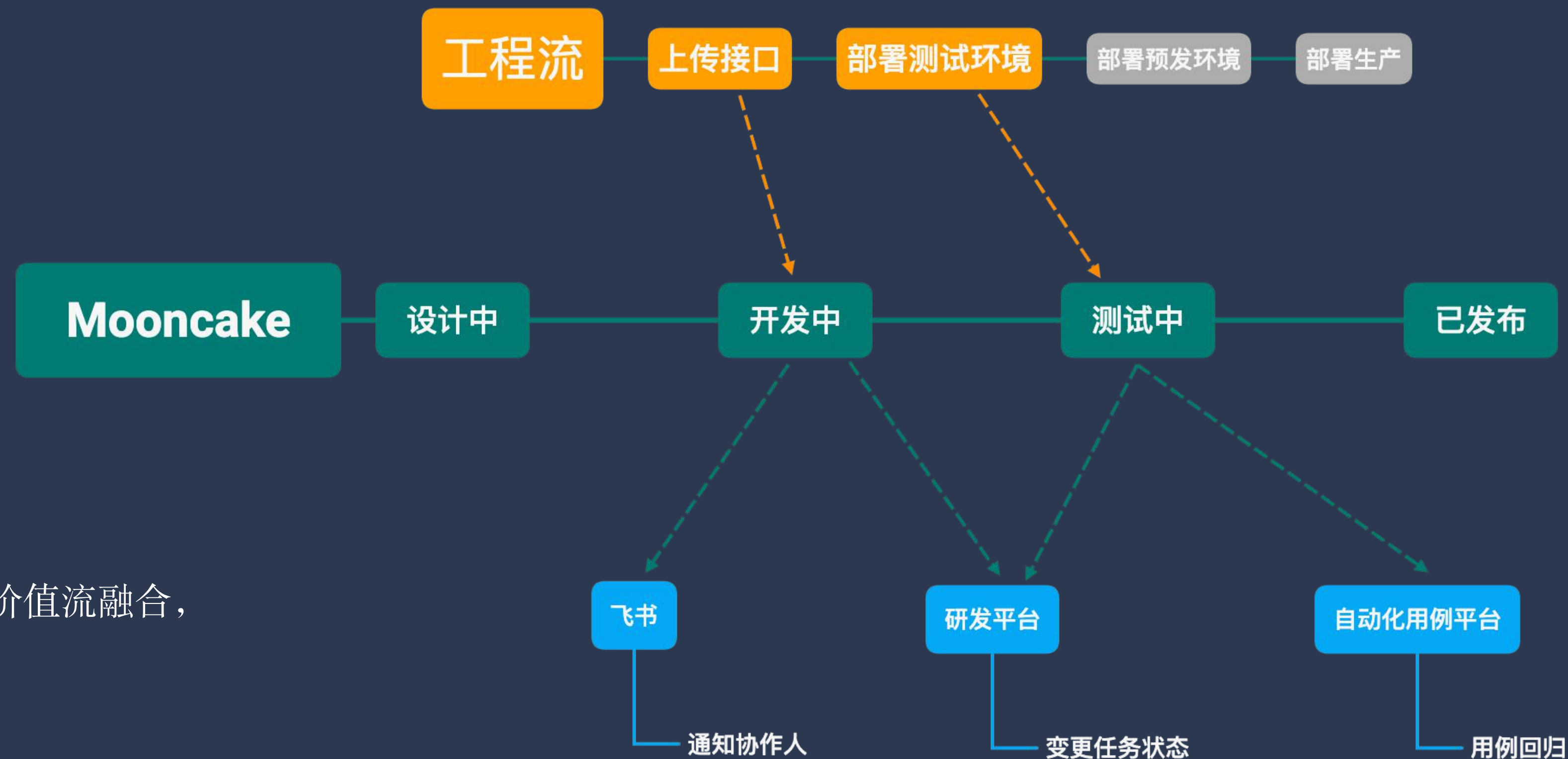
类postman线上调试工具，调试结果线上化

不止于前端 - 服务端文档驱动流程

服务端研发流程



不止于前端 — 流程融合

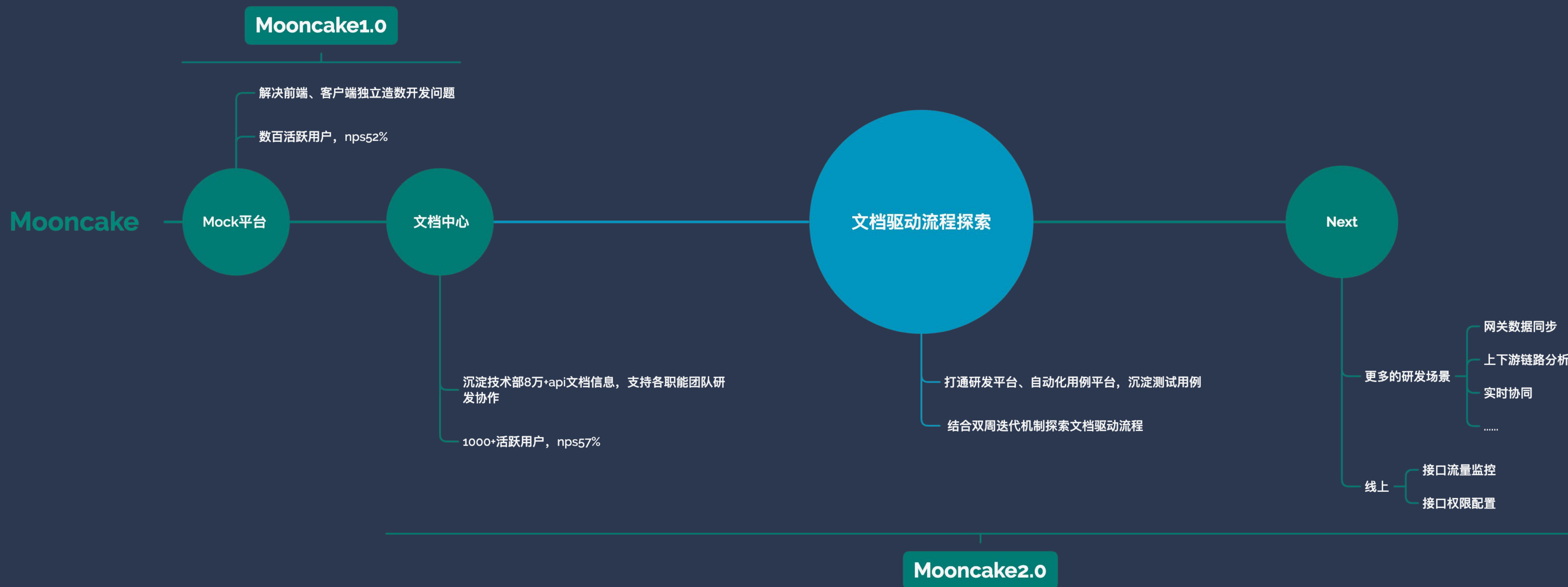


通过mooncake将研发工程流和业务价值流融合，
让研发聚焦代码流程的推进

大纲

- 迭代中的效能瓶颈分析
- 文档驱动流程相关介绍
- 文档驱动下前端的研发流程
- 后续计划

后续规划



THANKS

—
软件正在重新定义世界

Software Is Redefining The World