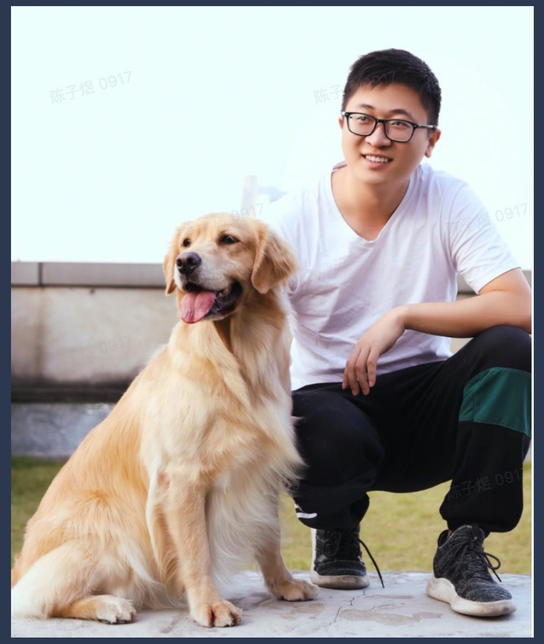
关于 API 文档驱动研发模式的探索与实践

陈子煜

得物商家前端负责人

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

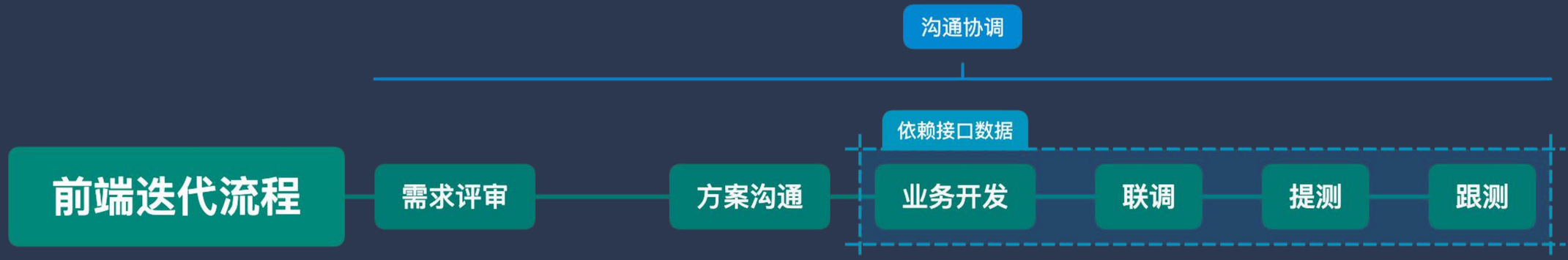
大纲

- 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

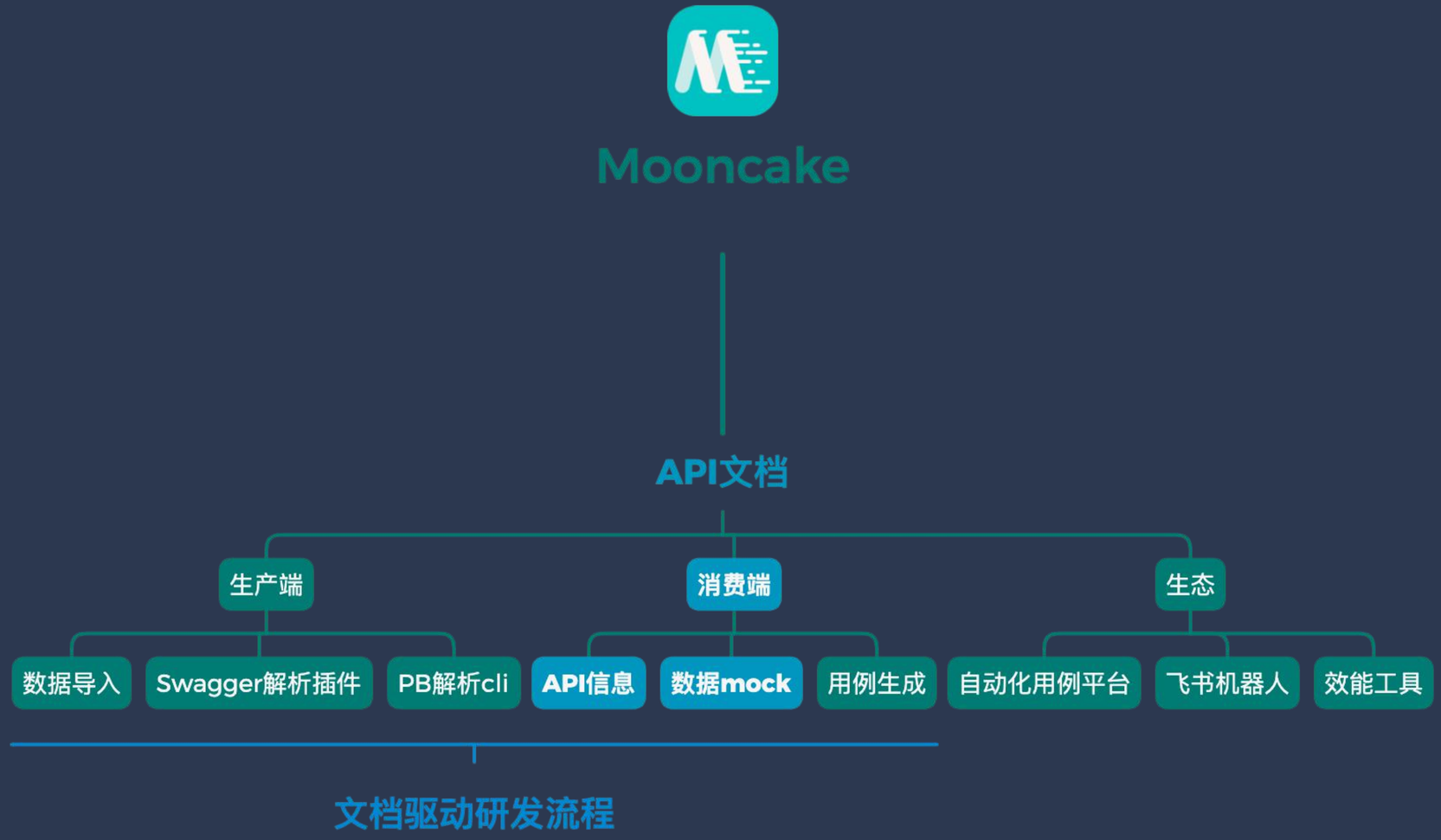
通过idea插件、cli等工具快速生成API文档

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

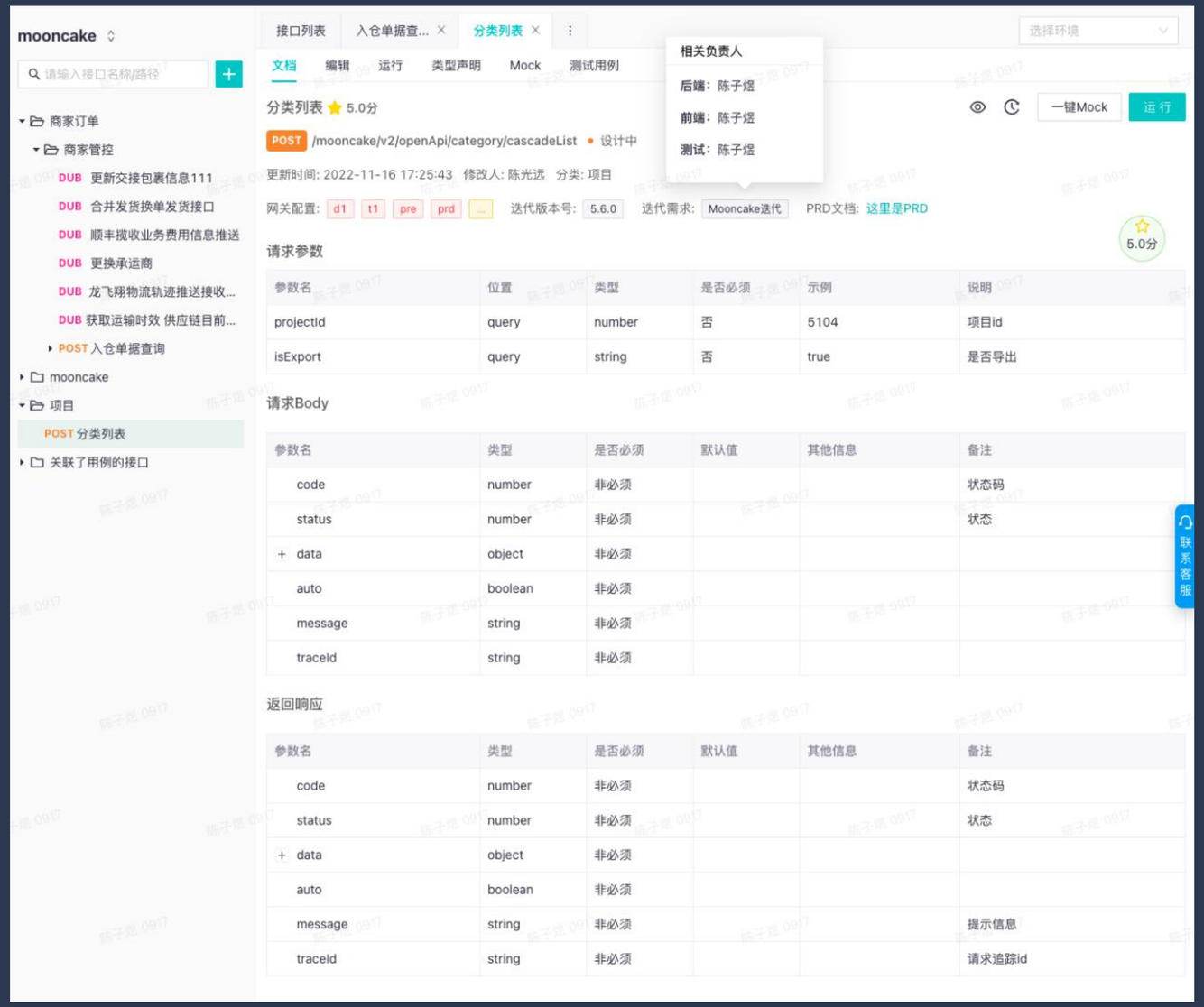
- 该接口本次迭代相关的如PRD链接等需求信息
- 从研发协同平台同步和该需求相关的研发人员信息，接口发生变更时能够及时触达相关人

API文档信息

写什么

接口的基础信息字段，包含但不限于：

- 出入参数
- 字段备注
- 文档备注
- 文档评论
- ……



API文档信息

写什么

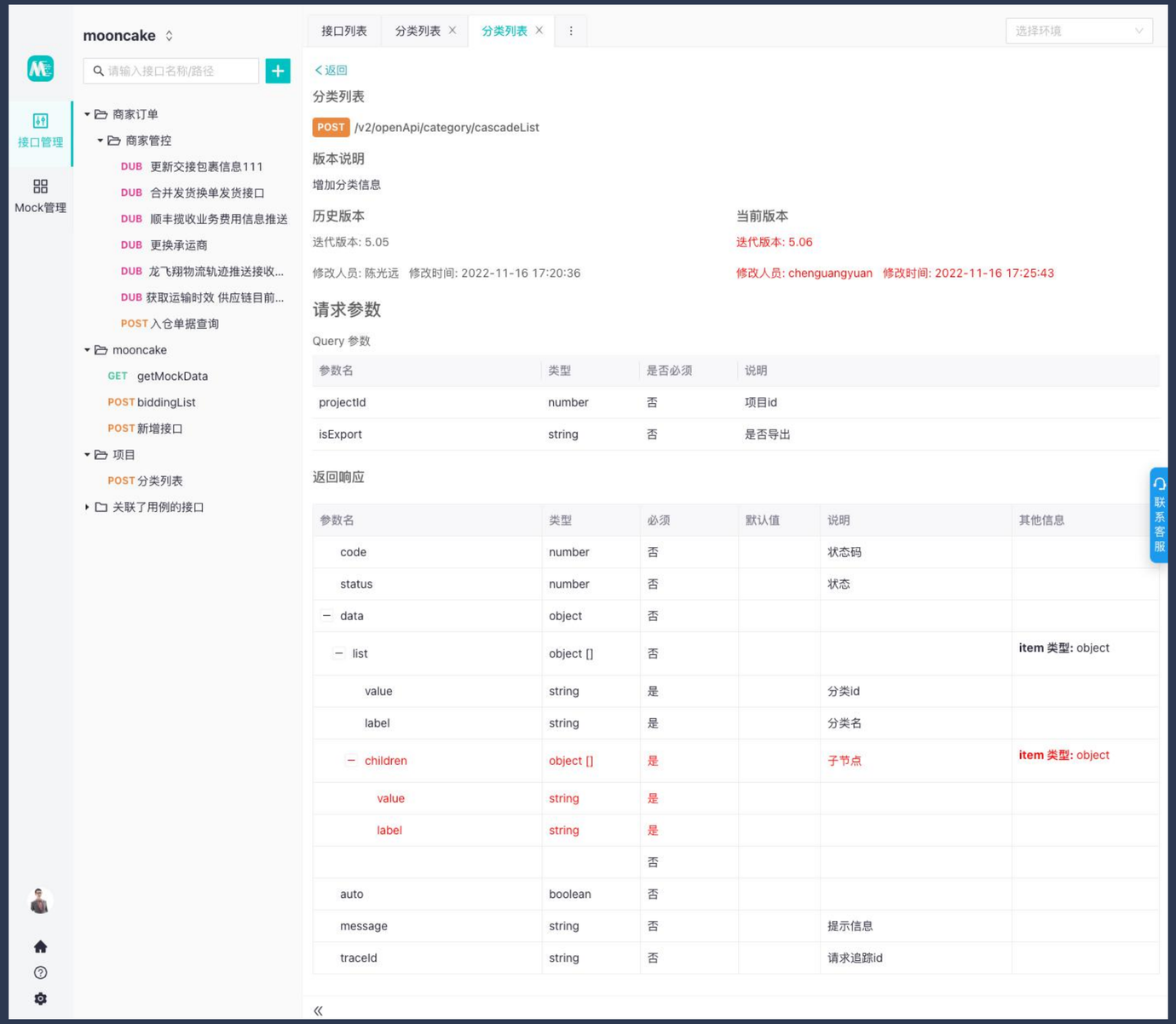
以迭代维度记录接口的历史版本信息，包括但不限于：

- 变更原因
- 变更时间
- 变更人
- ……

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



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

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
- ……

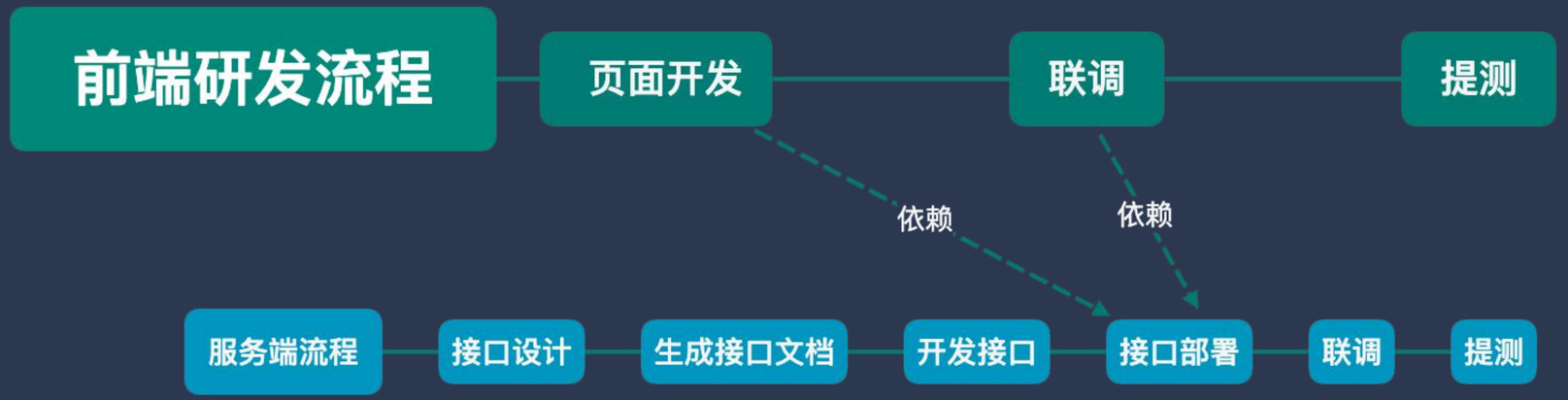
大纲

- 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

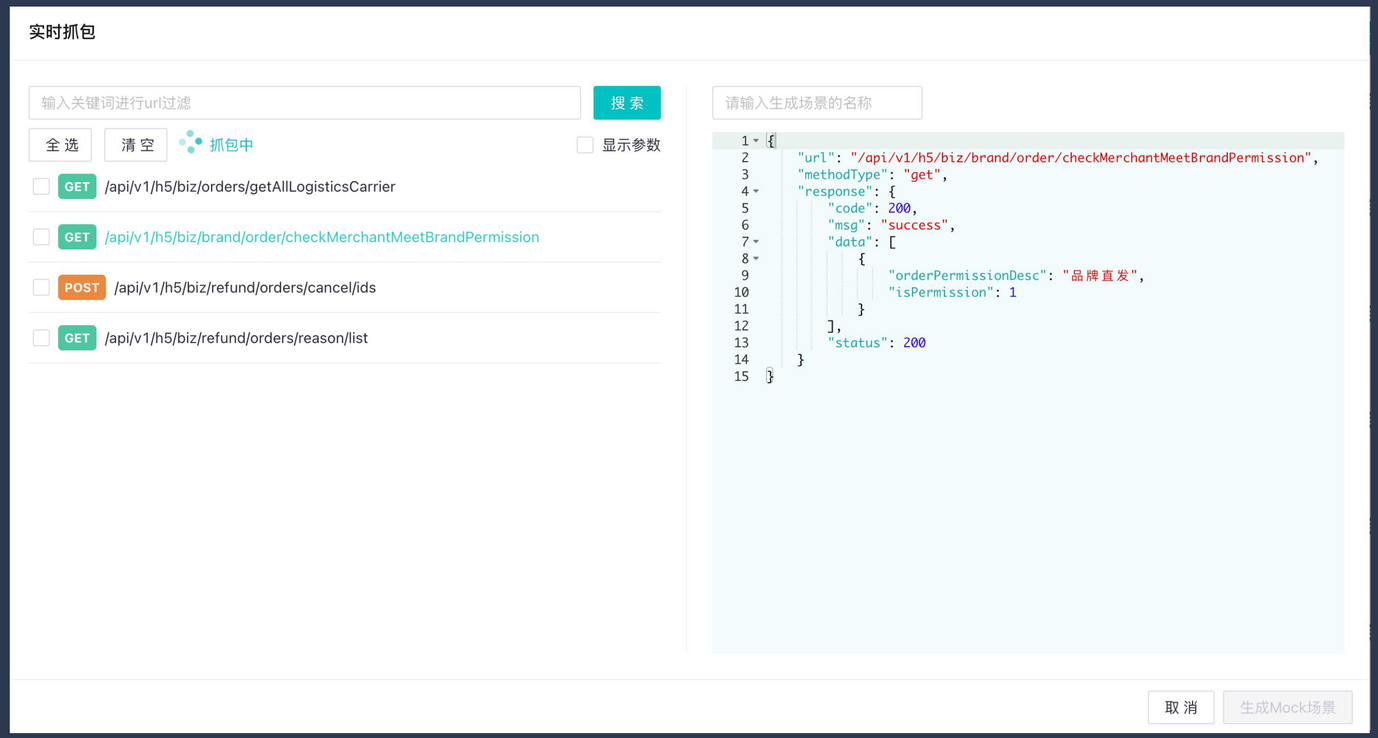
入参校验mock

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



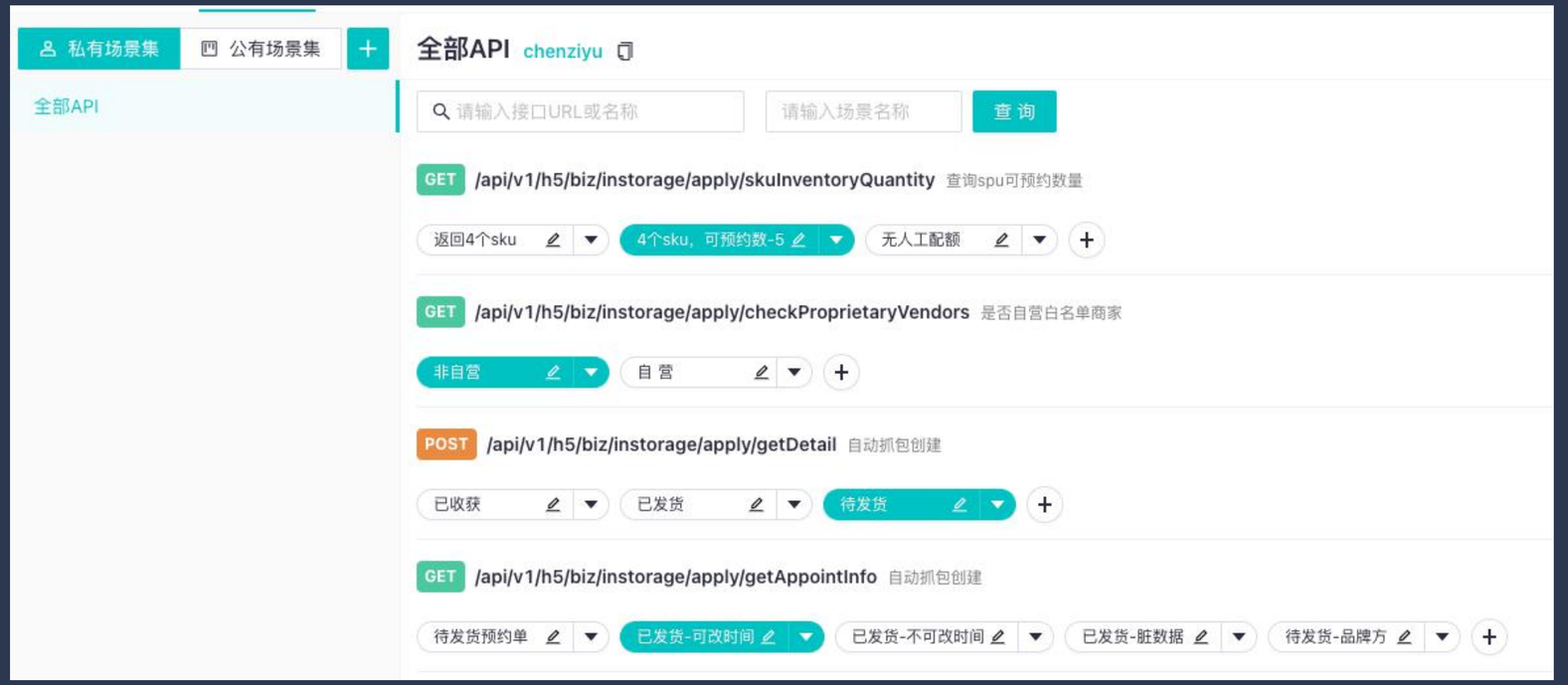
文档生态 – API Mock

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

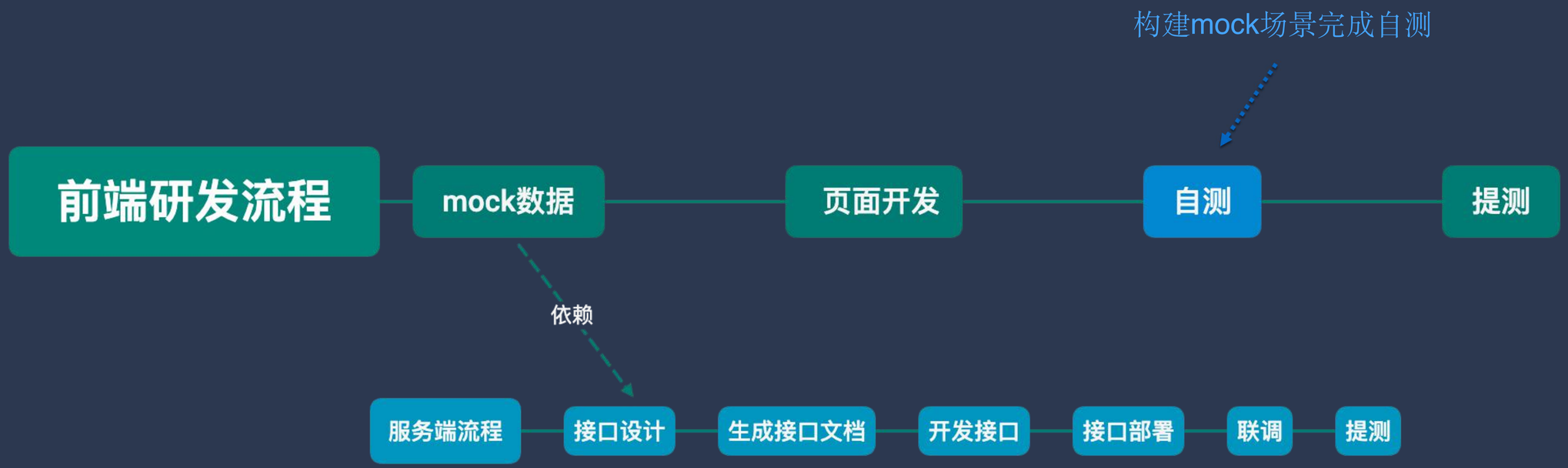
Mock场景集

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



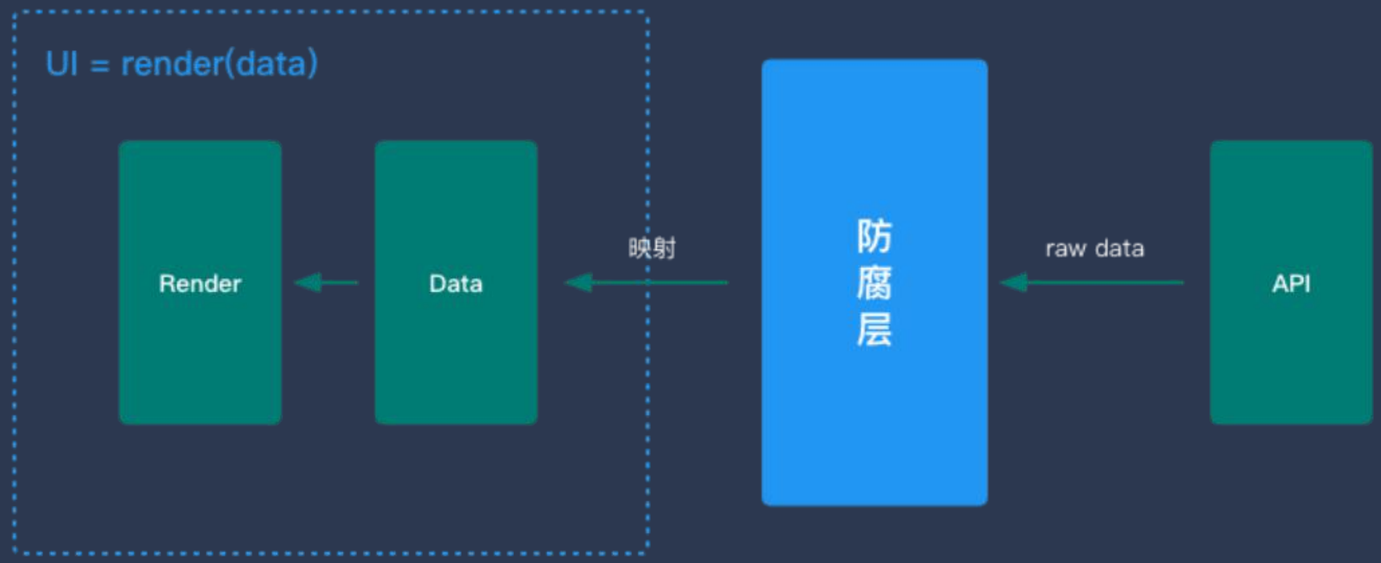
Mock数据的生产依然依赖于服务端接口定义

前端如何文档驱动

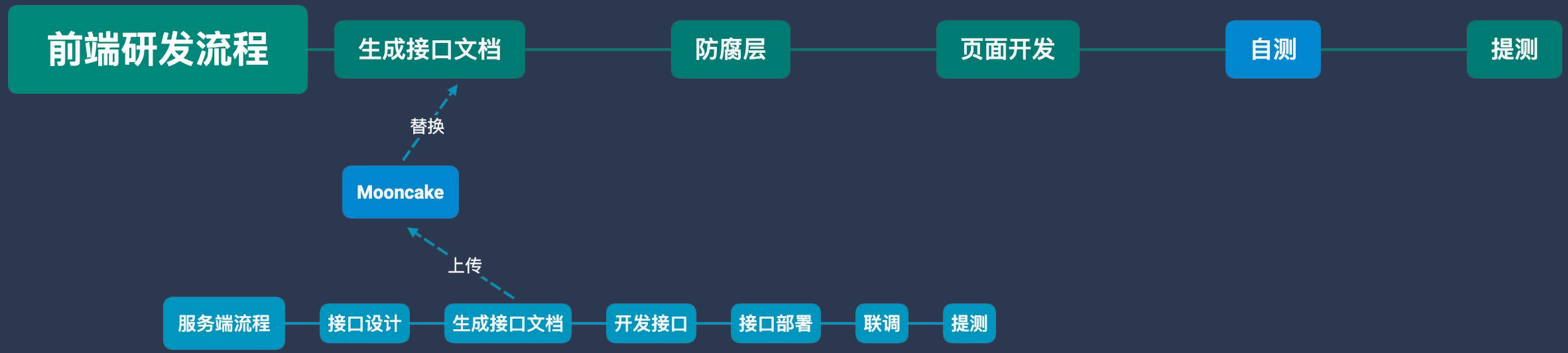
防腐层(ACL)

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



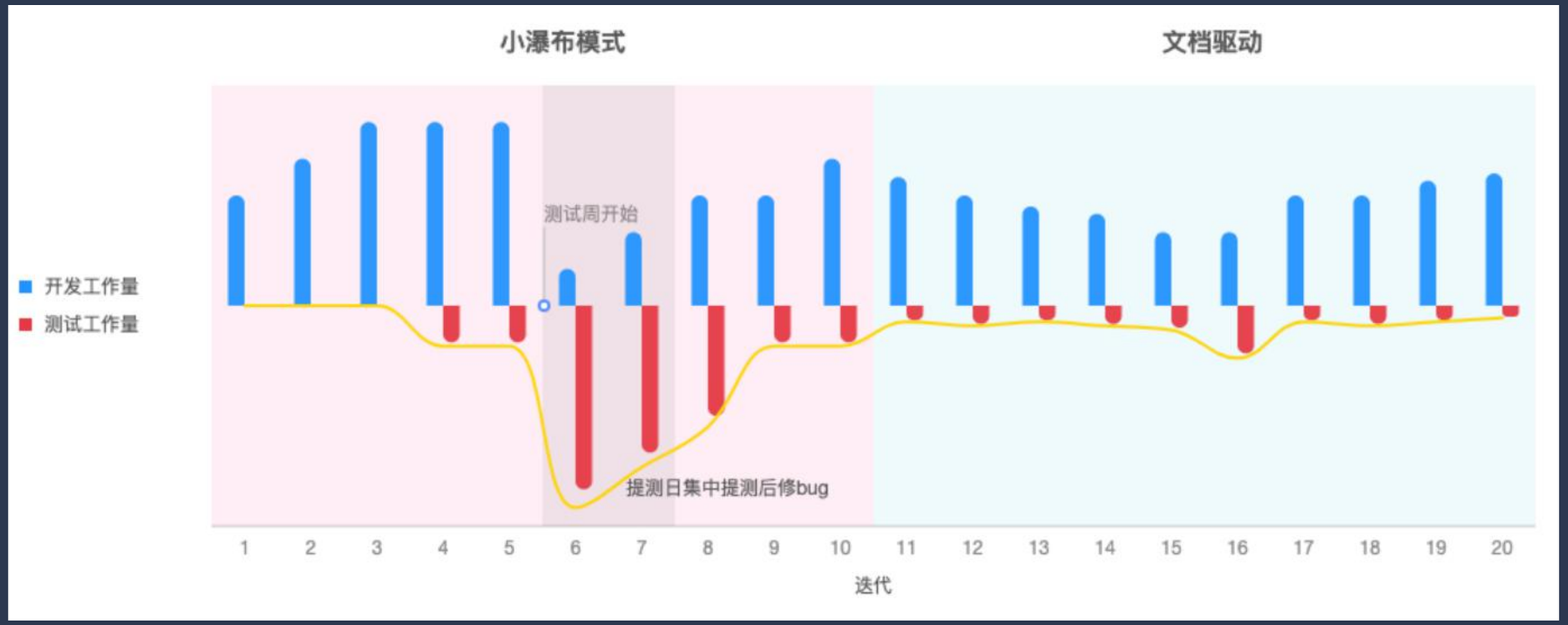
前端如何文档驱动



- 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

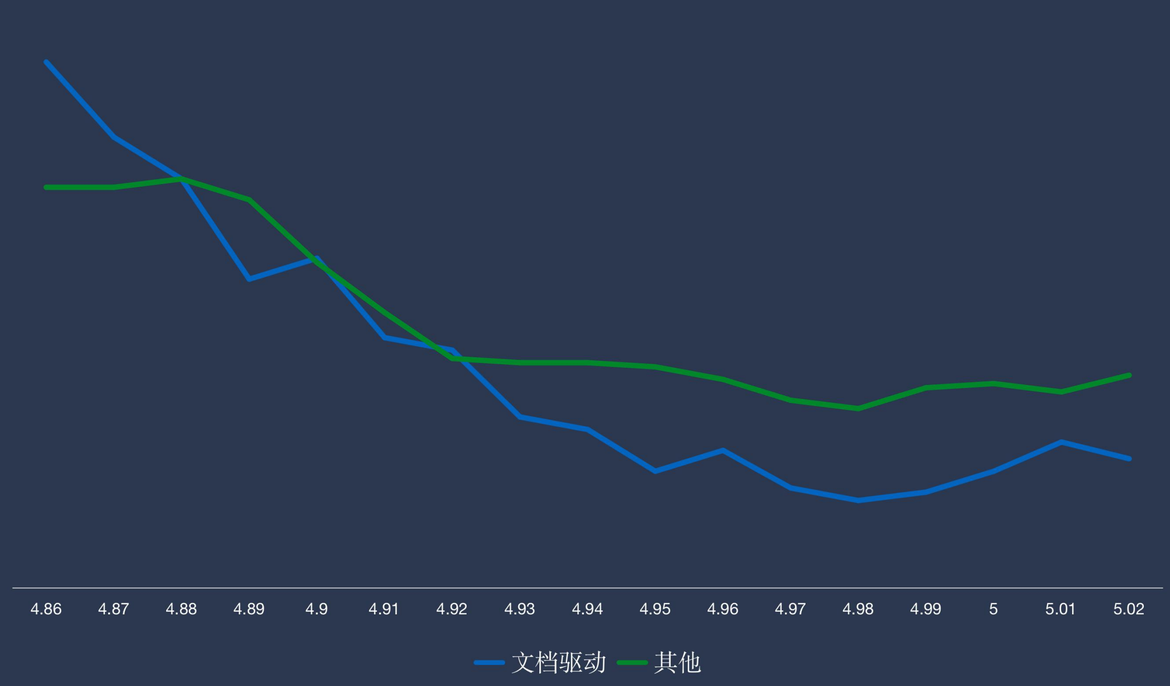


- ~~联调风险~~
- ~~时间碎片化~~
- ~~接口依赖~~

落地成果

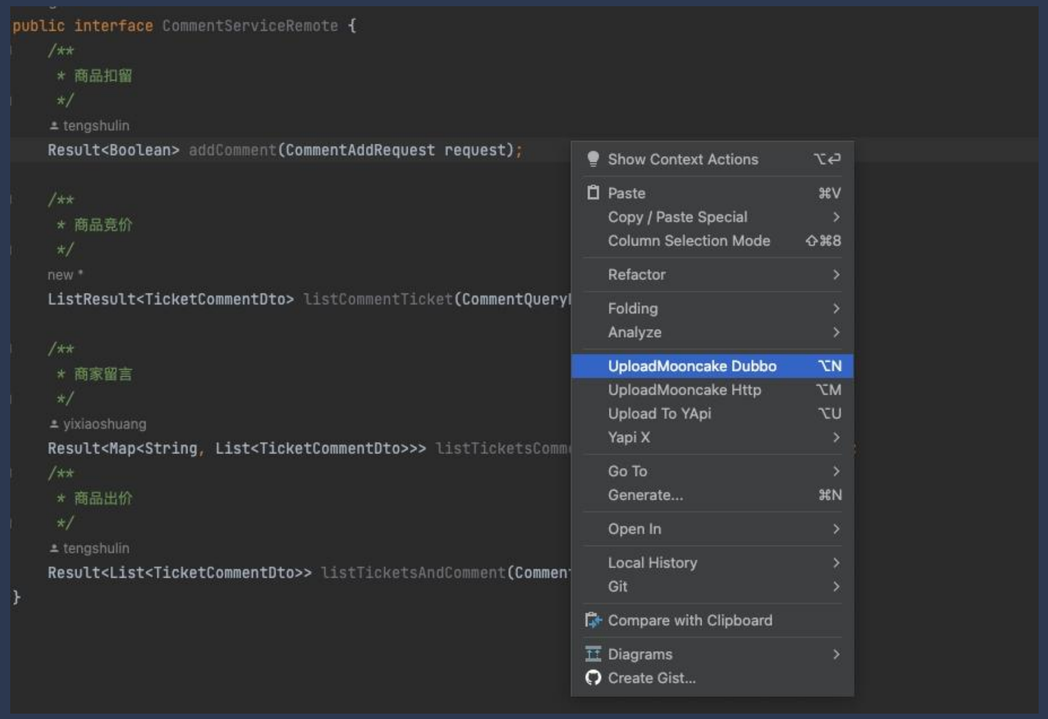
迭代效率

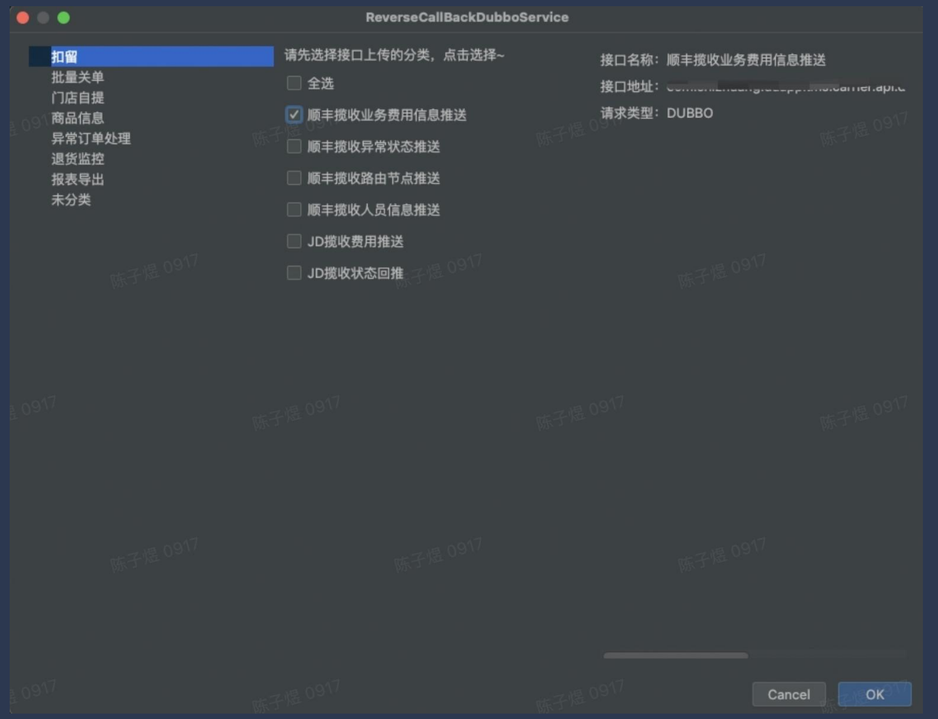
缺陷引入率



不止于前端 – 文档生产

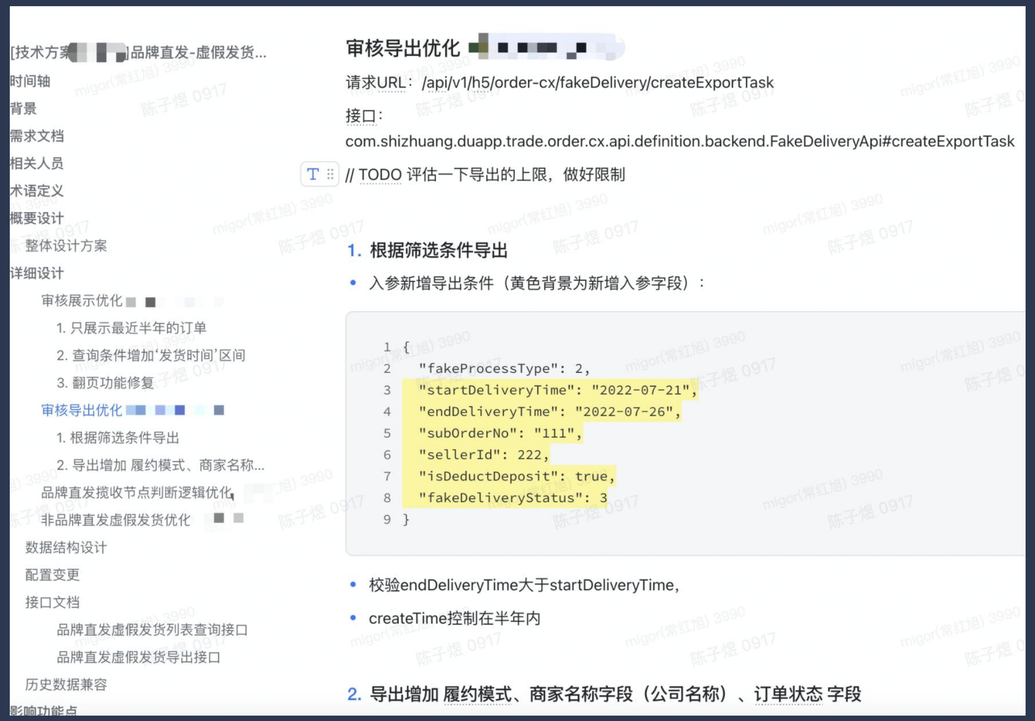
基于IDEA插件快速上传接口

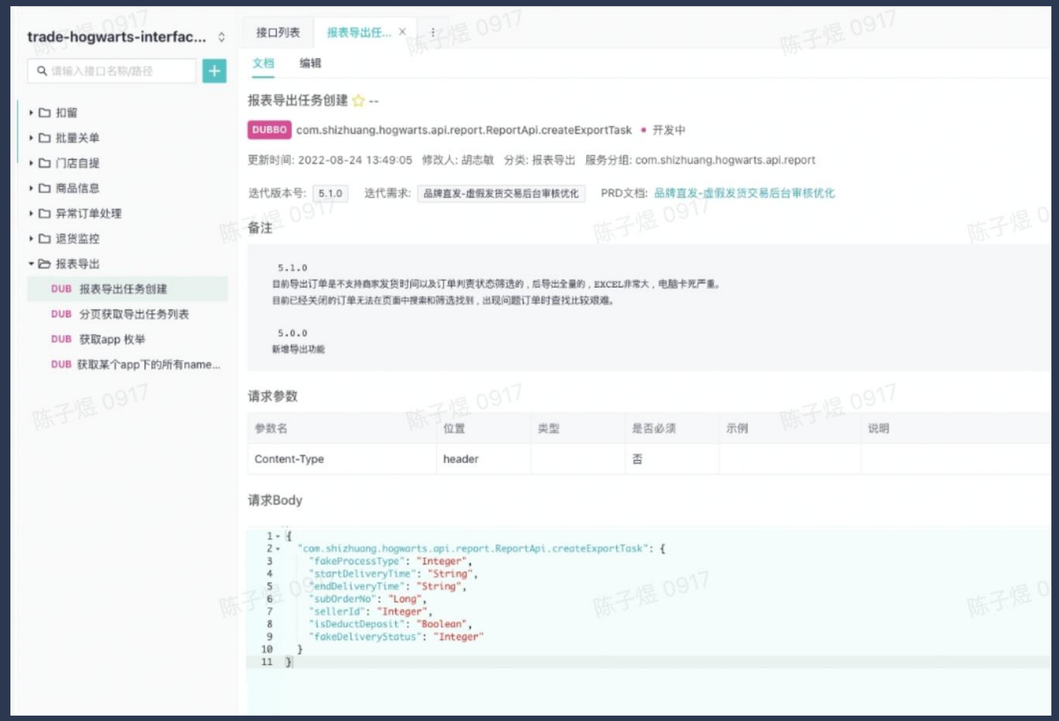




不止于前端 – 服务端文档驱动

Mooncake文档代替飞书文档



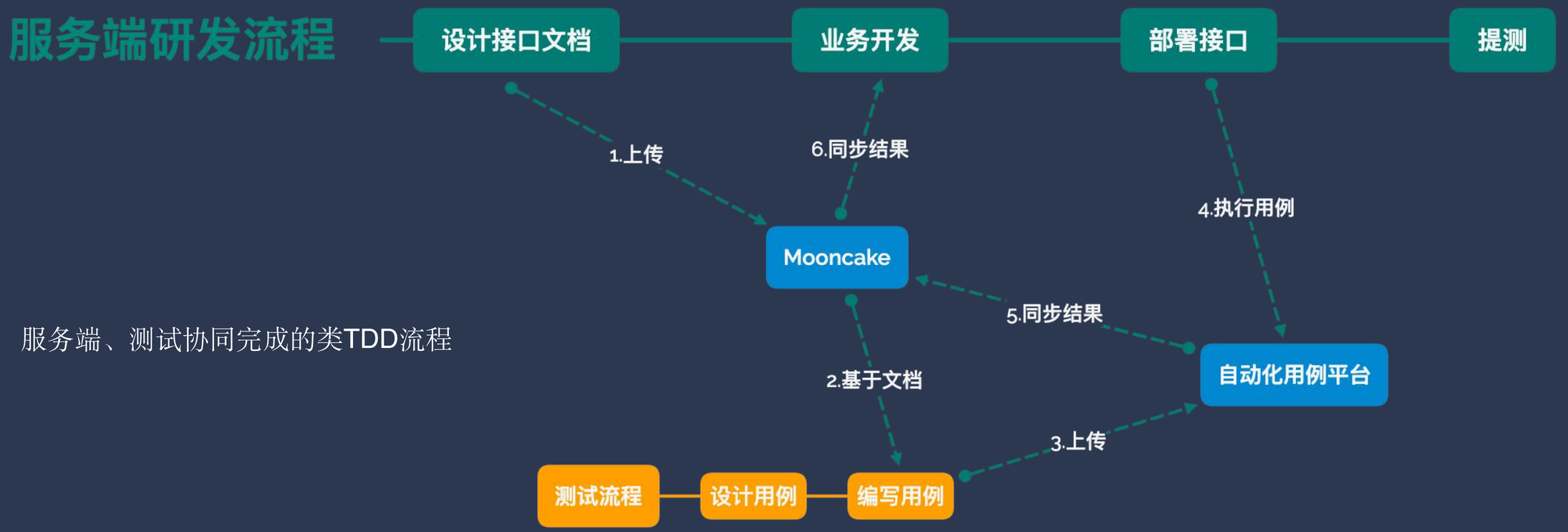


不止于前端 – 服务端文档驱动流程

提供Dubbo/GRPC Mock能力，解决服务端跨团队协作效率问题

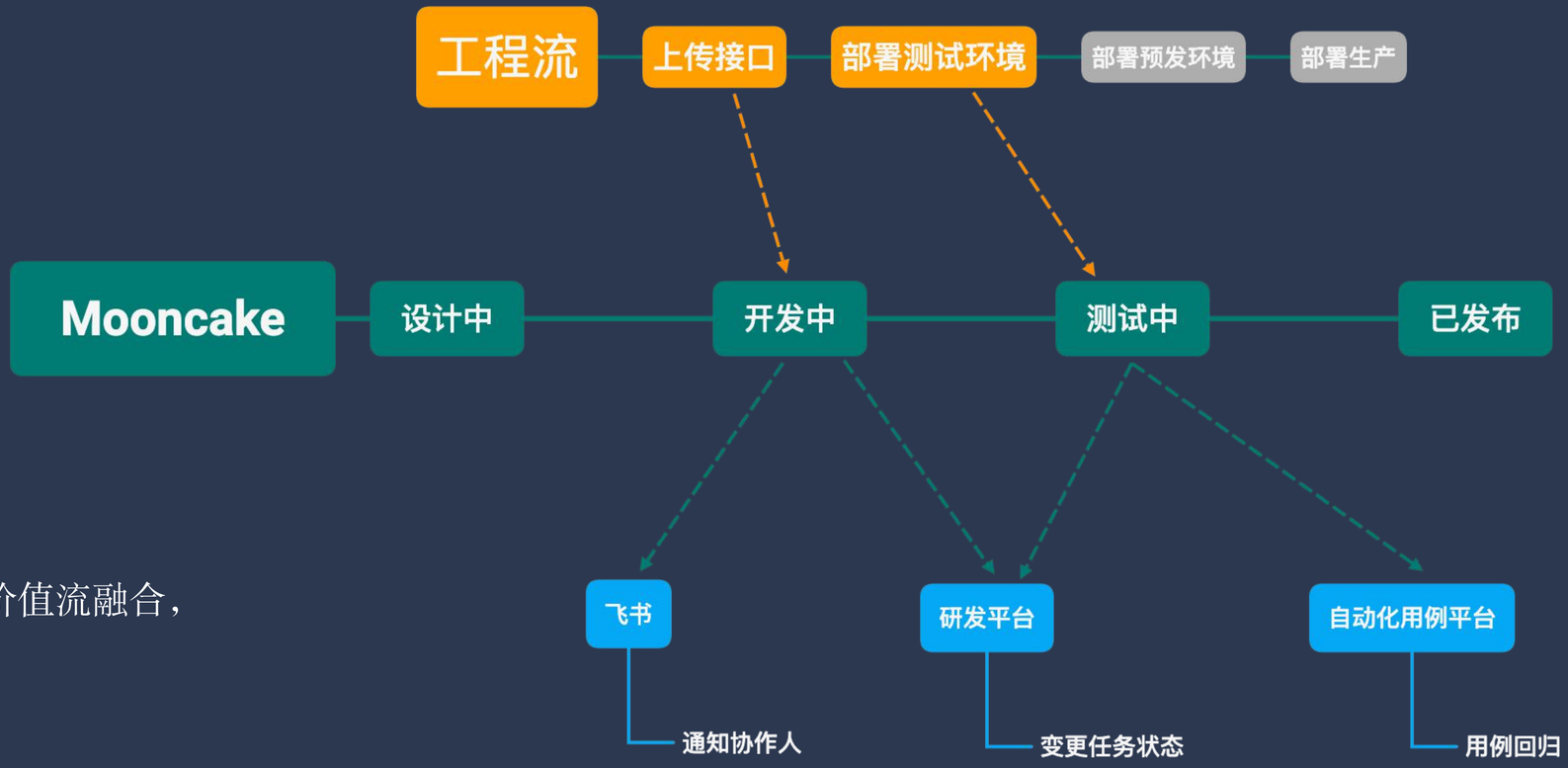
类postman线上调试工具，调试结果线上化

不止于前端 – 服务端文档驱动流程



服务端、测试协同完成的类TDD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


Mooncake1.0
解决前端、客户端独立造数开发问题
数百活跃用户，nps52%
Mooncake
Mock平台
文档中心
文档驱动流程探索
Next
沉淀技术部8万+api文档信息，支持各职能团队研发协作
1000+活跃用户，nps57%
打通研发平台、自动化用例平台，沉淀测试用例
结合双周迭代机制探索文档驱动流程
更多的研发场景
网关数据同步
上下游链路分析
实时协同
.....
线上
接口流量监控
接口权限配置
Mooncake2.0

THANKS

软件正在重新定义世界

Software Is Redefining The World