

研发效能度量的误区、 体系化实践和效能提升案例

张乐

京东DevOps与研发效能技术总监

100 讲师简介



// DevOps 资深实践者

- DevOps与研发效能资深技术专家（百度、京东）
- 曾任职全球五百强外企、国内多家一线互联网公司

DevOps 技术布道师

- DevOps道法术器 – 企业级DevOps立体化实施框架
- EXIN DevOps **全系列** 国际认证官方授权讲师
- DevOpsDays 中国核心组织者、出品人、金牌讲师
- 凤凰项目 DevOps 沙盘 官方授权教练
- TOP100 年度案例峰会联席主席，DevOps专题出品人

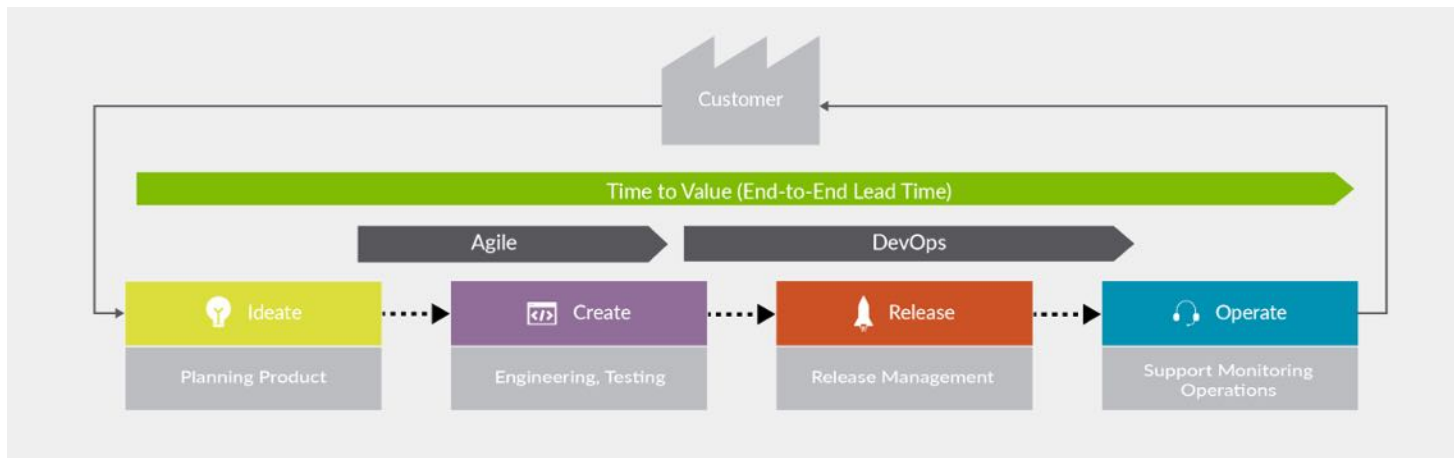
公众号



TOP1

- 1 业界研发效能的情况
- 2 研发效能度量的误区
- 3 研发效能度量体系模型
- 4 研发效能度量指导改进
- 5 经验分享和避坑指南

软件研发效能是企业的核心竞争力



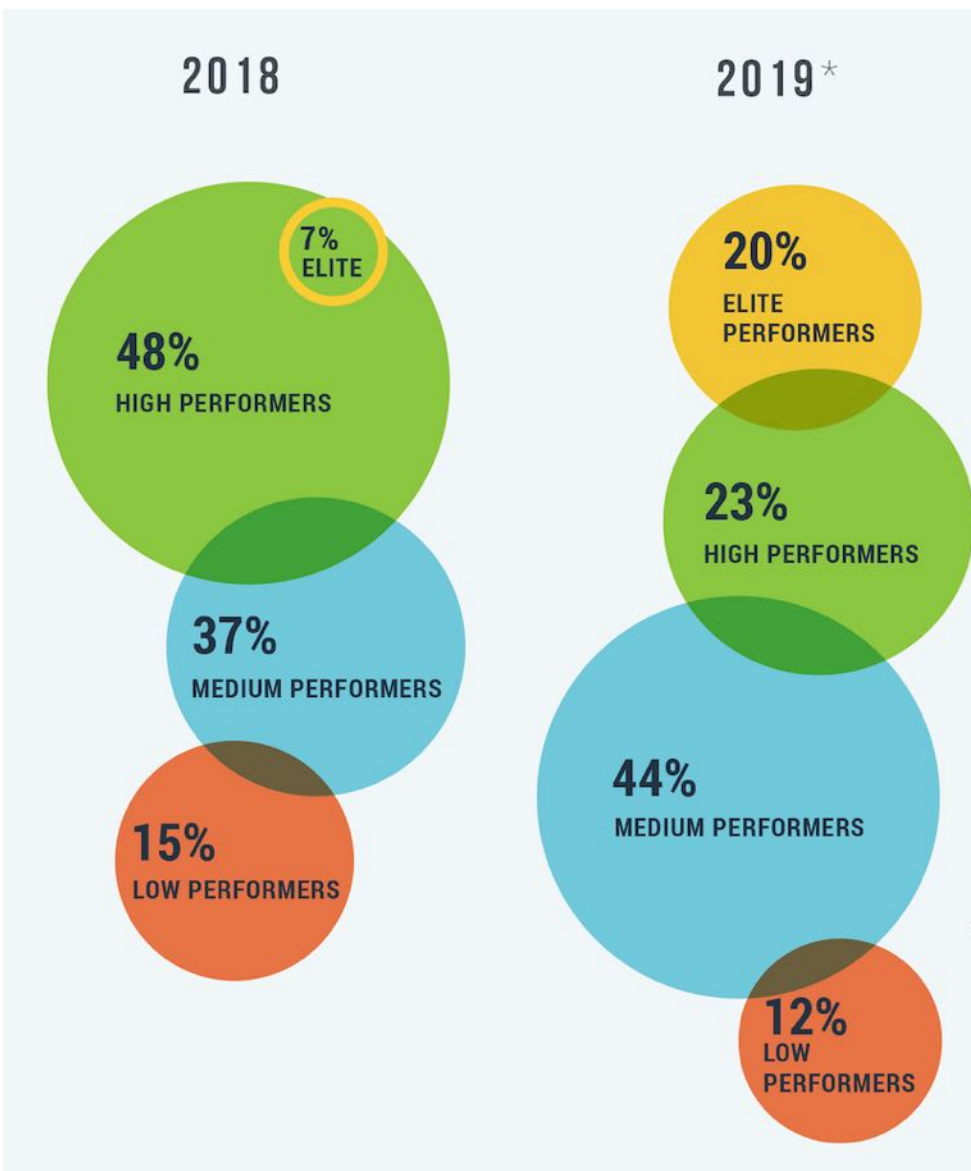
- 研发效能在业界**属于什么水平**？与竞争对手差距多大？
- 敏捷、精益、DevOps的转型有没有效果？是否可以**量化评估**？
- 是什么阻碍了软件**交付效率**？是什么影响了**交付质量**？
- 研发效能的改进，应该从何处入手，如何**针对性提升**？

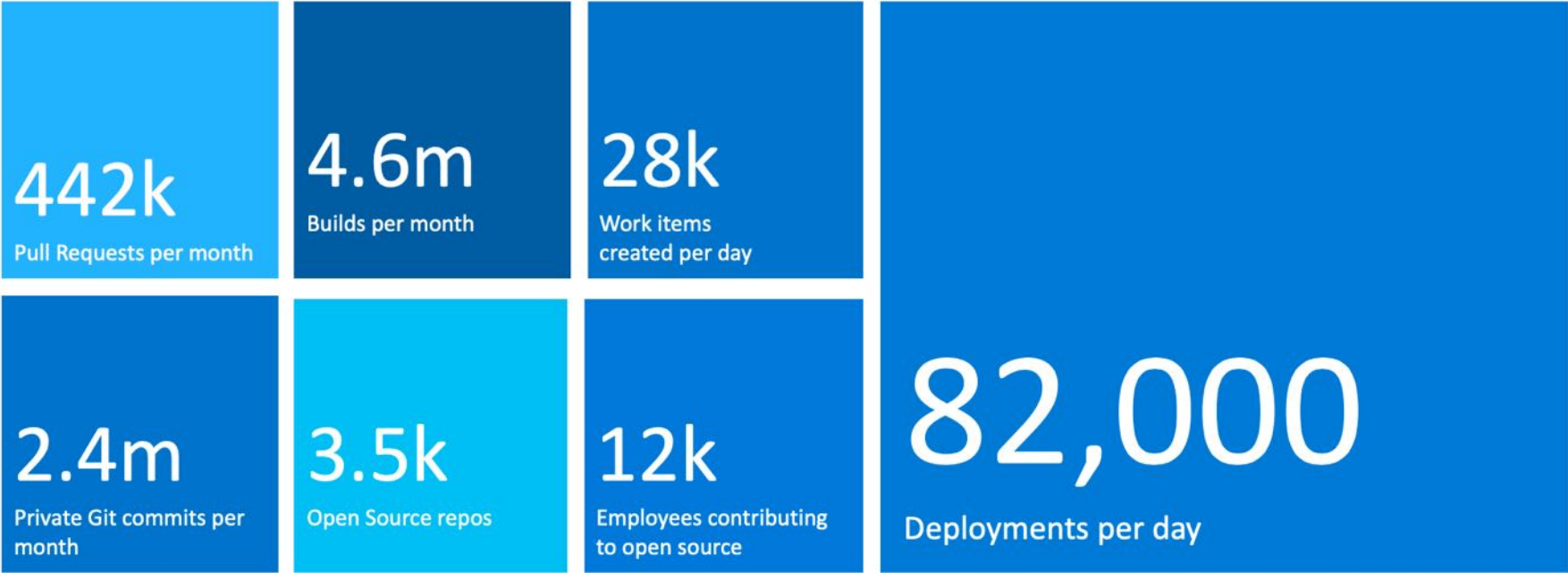
没有度量就无法管理

“You can’t manage what
you don’t **measure**”
Peter Drucker

软件交付效能的度量指标*	精英	高效能	中等效能	低效能
部署频率 对于您负责的主要应用或服务，您的组织将代码部署到生产环境或者发布给最终用户是什么频度？	按需（每日多次部署）	介于每天1次和每周1次之间	介于每周1次和每月1次之间	介于每月1次和每六个月一次之间
变更前置时间 对于您负责的主要应用或服务，变更的前置时间是多少（例如：从代码提交到代码成功运行在生产环境需要多长时间）？	少于1天	介于1天和1周之间	介于1周和1月之间	介于1月和6个月之间
服务恢复时间 对于您负责的主要应用或服务，发生服务故障或者发现影响客户的缺陷（例如：计划外中断或者服务受损），恢复服务通常需要多长时间？	少于1小时	少于1小时	少于1天 ^a	介于1周和1个月之间
变更失败率 对于您负责的主要应用或服务，对产品或发布给用户的更改中有多大比例会导致服务降级（例如：导致服务损坏或者服务中断），需要事后补救（例如：需要热修复，回滚，前向修复，补丁）？	0-15% ^{b,c}	0-15% ^{b,d}	0-15% ^{c,d}	46-60%

你的软件研发效能怎么样？



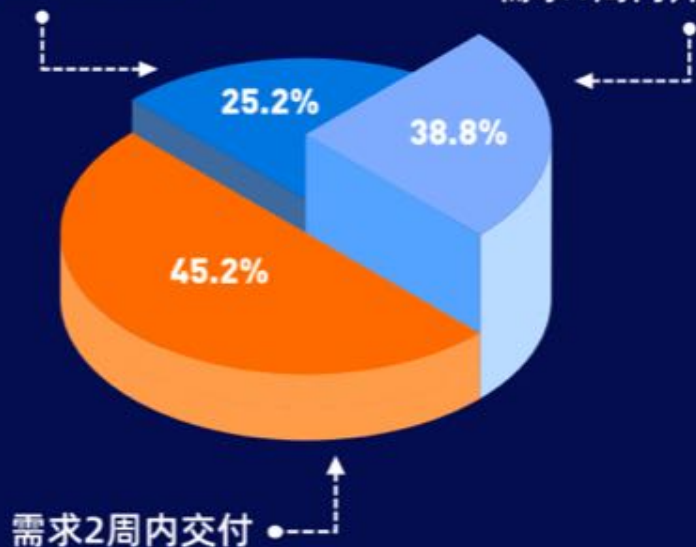


Data: Internal Microsoft engineering system activity, March 2019

效能数字化 直奔211

变更1小时内完成发布

需求1周内开发完成



需求2周内交付

2020，更多团队已达“211”交付速度

Aone（云效）精益看板实践、自动化发布流水线、AI研发助手、无人值守发布等工具，让我们离“211”交付愿景更近一步。



疫情期间新增代码行
2.9 亿

3天 开发出健康码引擎，
20天 健康码在全国24省200多座城市落地。
2小时 上线无接触配送项目，
3小时 上线火神山无接触收银项目。
5000行 代码完成全序列基因检测算法，
CT影像分析从5-15分钟降为 20s，识别准确率达 96%。
阿里速度背后，是扎扎实实的数据技术。

平均每个工作日提交 **6 万次**

每次代码变动 **107.26 行**

持续集成构建超 **50 万次**

周代码扫描超 **30 万次**

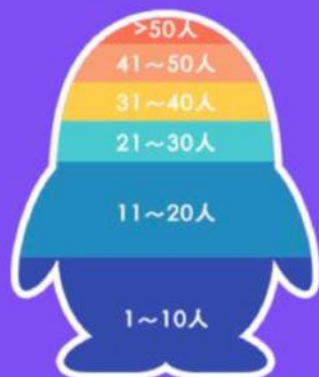


研发效能度量数据 – 腾讯

保持小团队，保持敏捷

60%

项目团队规模控制在
20人以下



47.8%

迭代周期在1周内



用户为本，快速响应

在腾讯

平均每天完成需求

3802 ↑



28%

需求在1天内得到响应



腾讯74%的技术Leader仍在持续输出代码，2019年全年，平均每人输出代码3万行，参与140次代码评审；38%的12级及以上技术专家潜心编码，平均每人输出代码3.3万行，参与155次代码评审。

研发效能度量具体怎么做？



- 1 业界研发效能的情况
- 2 研发效能度量的误区
- 3 研发效能度量体系模型
- 4 研发效能度量指导改进
- 5 经验分享和避坑指南

案例故事：某大型跨国公司大规模敏捷转型

- 大型跨国公司，某领域曾经的市场领导者，市值一度达到2500亿美元
- 高管们意识到敏捷对于他们适应快速变化的市场的极为重要
- 高层对大规模敏捷转型给予了极大的支持，从上而下，投入巨大
- 基层开发人员对任何敏捷实践都没有异议，而且自我感觉良好
- 定义了敏捷团队的行为，与当时的最佳Scrum实践对比，进行**度量评估**
- 曾被视为敏捷开发方法可以在大规模企业落地的证据之一

As an agile citizen, I can assess a team's behavior and compare it to current Scrum best practices, so that I can consider changes that might increase productivity.

— An Agile Citizen

敏捷团队的度量评估

问题1：迭代

当团队对迭代进行承诺时，需要知道迭代的长度，以便按更好的节奏交付价值。

验收测试（不加总）

- 迭代长度4~6周
- 迭代长度4周之P
- 过去三个迭代，
- 过去三个迭代，
- 过去三个迭代，
- 过去三个迭代，

问题5：产品待办列表

列表按价值排序，因

验收测试（加总）：

- 团队工作于多个
- 团队工作于单个
- PO定期与团队讨论
- 三个月之后才做
- 团队了解每个故
- PO根据净现值排
- PO优先安排低成本的原型以便尽快验证价值，得2分

问题4：产品负责人

产品负责人帮助团队理解和排序价值，因此获得长期收益。



会，得2分
团队对故事进行估算，得1分
图，得1分

件今早交给用户并快速调整。

2分

完成，得2分
告，得1分

2分

研发效能度量的原则

01

结果指标 > 过程指标

以终为始，通过结果指标评估结果，通过过程指标指导改进。

比如：需求交付周期是结果指标，敏捷活动成熟度是过程指标

02

全局指标 > 局部指标

过度局部优化可能导致全局劣化，只聚焦在易于度量的局部指标上，会以牺牲组织更好地提升全局目标为代价

03

定量指标 > 定性指标

尽量使用量化指标客观评价，并通过系统自动采集，降低对团队的干扰。但也不排除部分综合评价的定性指标

04

团队指标 > 个人指标

指标设定需要促进团队协作，共同努力达到组织目标。不能因相互冲突的指标而破坏团队配合，制造出更多的部门墙

指导性，可牵引行动

指标设定为目标服务，指标的数值和趋势可以牵引团队改进。比如适当设定缺陷类指标可以促进质量内建能力的建设

05

全面性，可相互制约

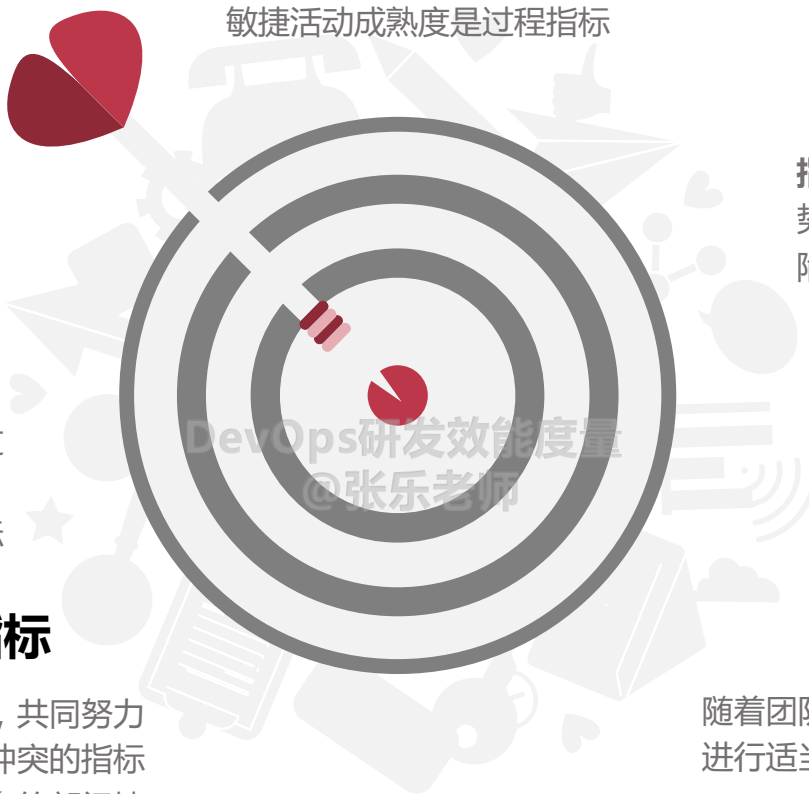
需求周期 vs 缺陷数量
需求吞吐量 vs 需求规模
研发周期 vs 技术债务

06

动态性，按阶段调整

随着团队能力的提升，指标也需要随之进行适当调整，从而促进团队持续改进

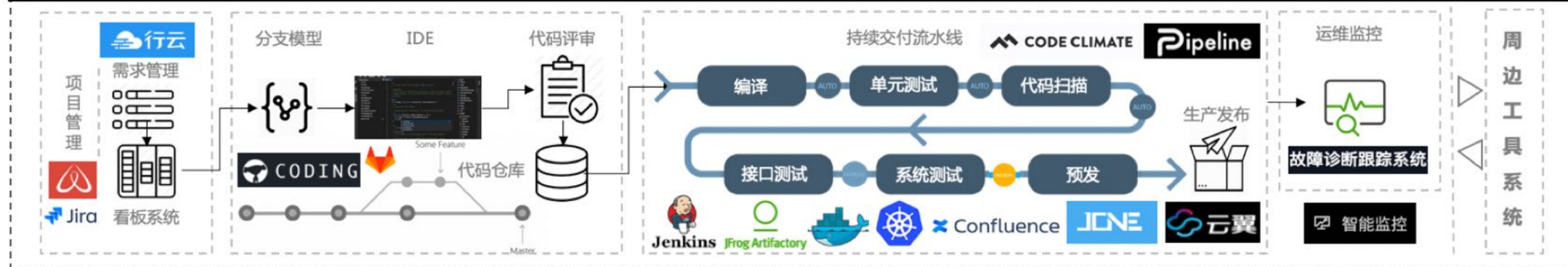
07



- 1 业界研发效能的情况
- 2 研发效能度量的误区
- 3 研发效能度量体系模型
- 4 研发效能度量指导改进
- 5 经验分享和避坑指南

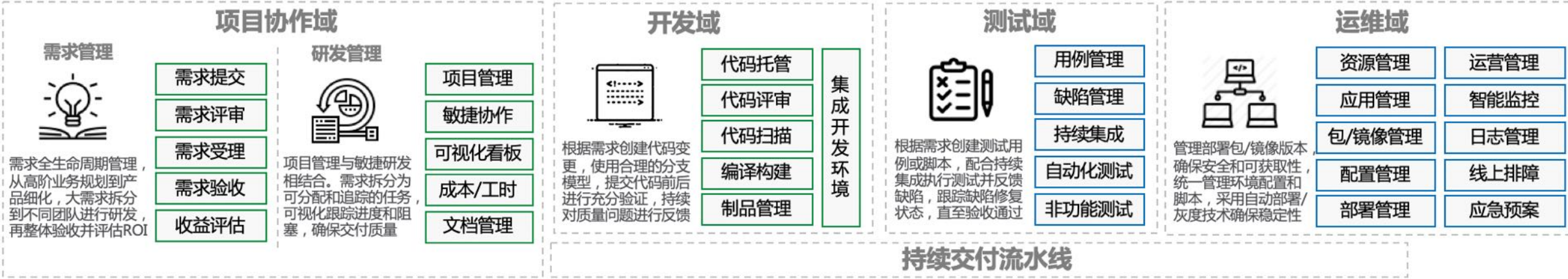


DevOps 工具链网络



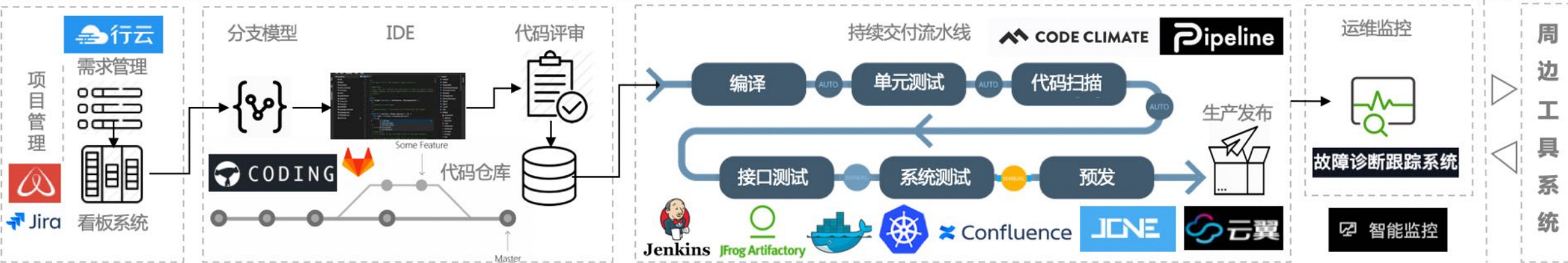
功能板块

研发效能度量

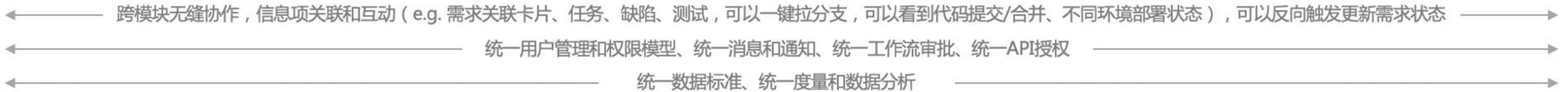


通用能力整合 + 业务定制（配置、插件、集成）

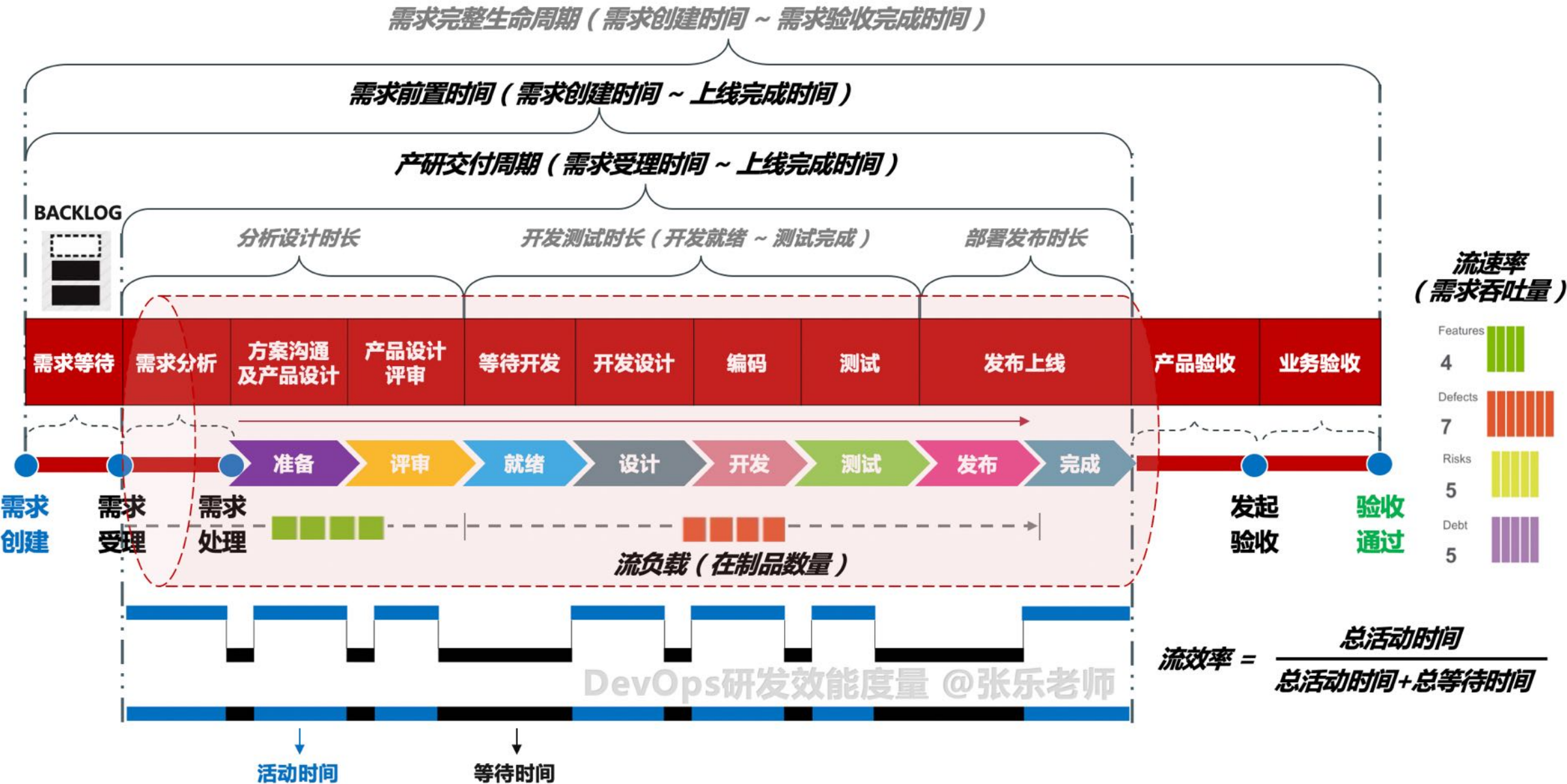
效能组件



平台框架（基础UI、基础服务、基础数据）

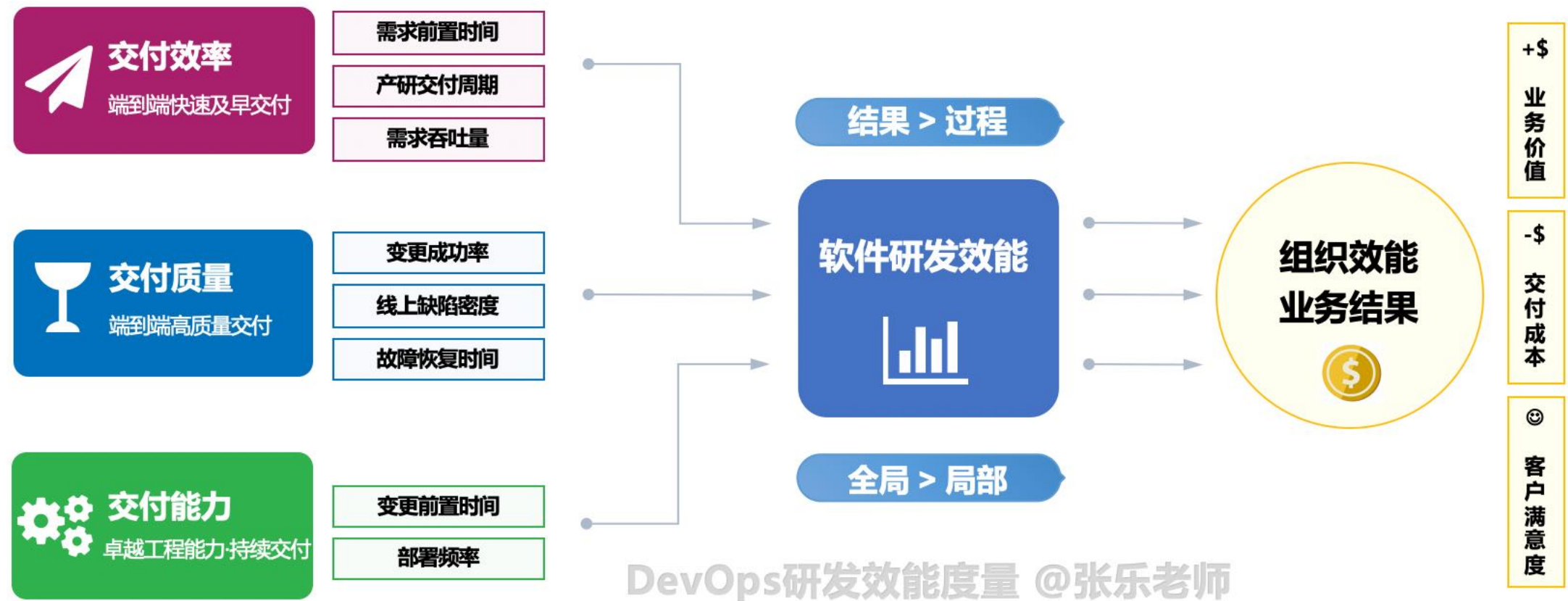


价值流交付模型



研发效能度量分析模型

在端到端交付价值流中，以结果指标为牵引，通过全局研发效能的提升促进组织效能提升



基于价值流的度量

需求

开发

测试

发布

运营



交付效率

端到端快速及早交付



交付质量

端到端高质量交付



交付能力

卓越工程能力 持续交付

需求前置时间

产研交付周期

需求各阶段平均停留时长

需求等待时长 • 需求分析时长 • 产品设计时长 • 研发排期时长 • 开发时长 • 测试时长 • 部署发布时长 • 提交验收时长 • 验收处理时长

流速率（需求吞吐量）

流效率 = 总活动时间 / (总活动时间 + 总等待时间)

需求数量

提出、受理、完成、验收通过

代码库数量

代码提交量

测试用例数量

资源利用率

需求按时交付率

代码提交/合并频率

缺陷解决时长

应用性能

需求规模（颗粒度）

圈复杂度/重复度

需求评审通过率

需求变更率

需求价值达成率

需求代码关联率

代码评审覆盖率

代码评审通过率

单元测试覆盖率

代码扫描问题数

代码技术债率

代码提测成功率

自动化测试覆盖率

自动化测试通过率

线下缺陷数量

状态、类型、严重程度、注入阶段

线下缺陷密度

缺陷解决率

缺陷逃逸率

部署次数

上线成功率

部署回滚率

系统可用性

线上缺陷密度

故障恢复时间 MTTR

故障检测时间 MTDD

故障间隔时间 MTBF

计划外工作占比

流负载（在制品数量）

构建频率

构建时长

构建成功率

自动化测试
执行时长

持续集成
红灯修复时长

部署时长

部署频率

变更前置时间（代码合入到部署）

图
例

结果指标

过程指标

业务
结果

+\$

业务
价值

-\$

交付
成本

😊

客户
满意度

团队 · 周期 · 版本

项目度量

项目成本

项目进度

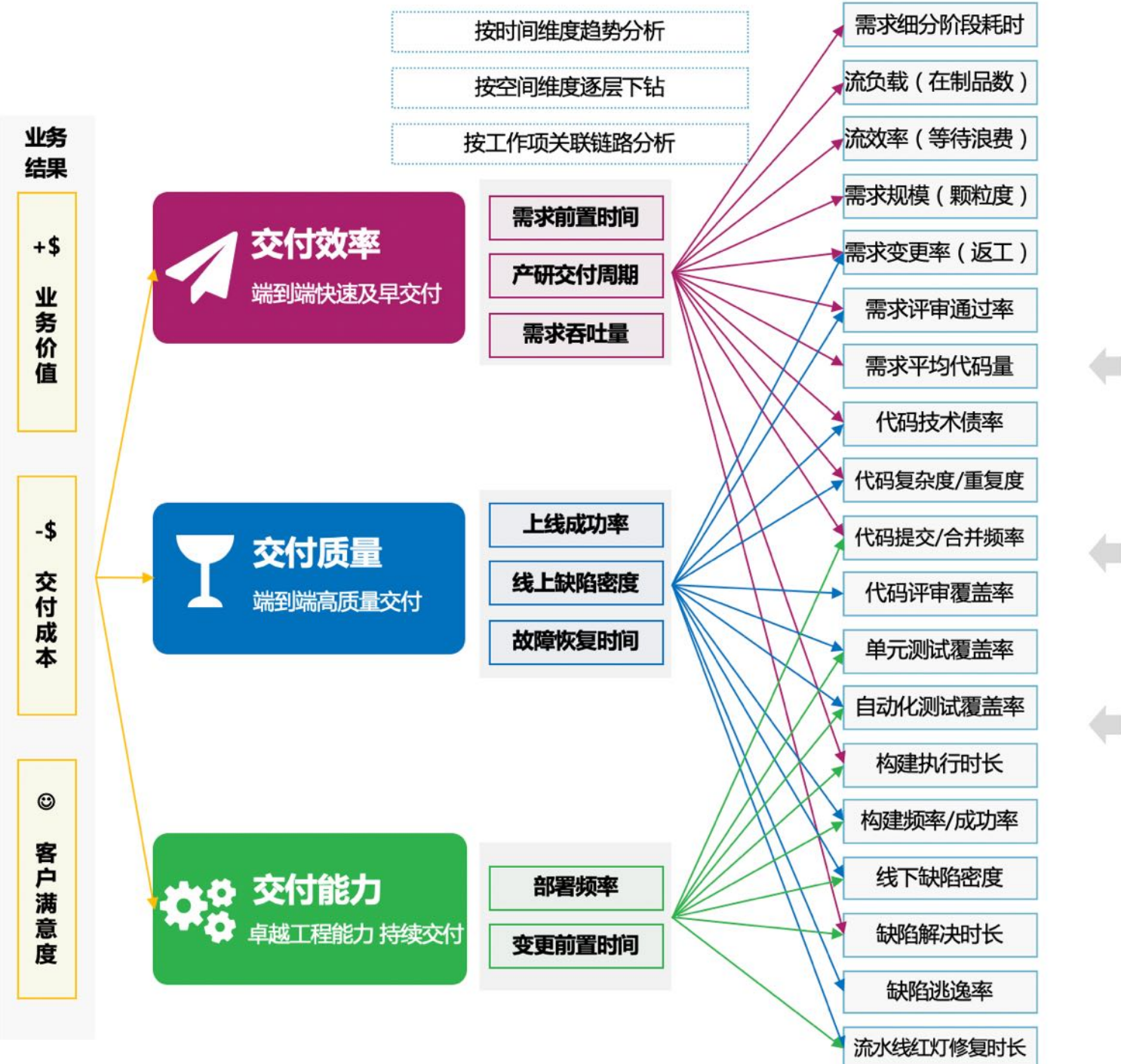
项目工时

项目质量

项目ROI达成率

项目满意度

DevOps研发效能度量 @张乐老师



- 1 业界研发效能的情况
- 2 研发效能度量的误区
- 3 研发效能度量体系模型
- 4 研发效能度量指导改进
- 5 经验分享和避坑指南

研发效能度量分析实例

需求前置时间

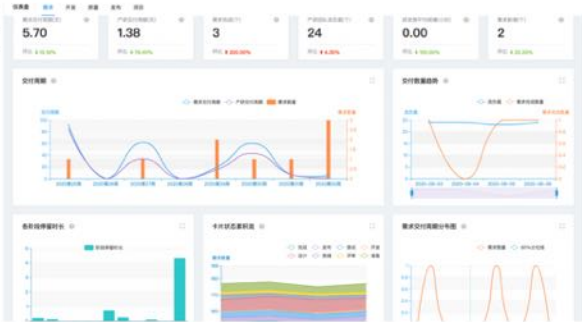
趋势分析

周/月/季度趋势
同比环比分析
效能改进决策点

下钻分析

关联分析

需求规模、需求吞吐量
线上缺陷密度（交付质量）
关联代码质量（技术债务）



按部门下钻

按组织结构逐层下钻

按团队下钻

按产品/特性团队下钻

按阶段下钻

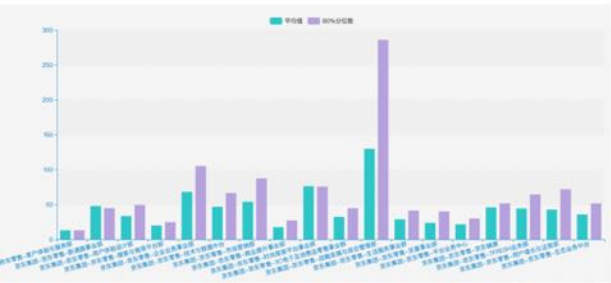
按阶段寻找低效瓶颈点

流效率分析

活动时间与等待时间占比

流负载分析

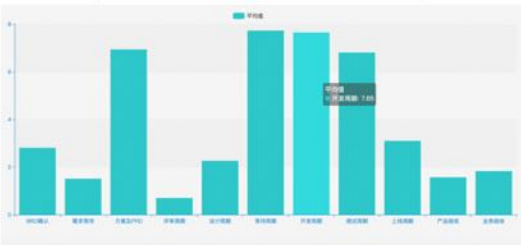
在制品数量和队列长度



累积流图



周期分解



明细分析



- 全链路仪表盘
- 自定义仪表盘
- 单点分析
- 需求单点分析
- 研发部门视角
- 受理部门视角
- 代码库单点分析
- 关联分析
- 全链路关联分析
- 问题诊断分析
- 设置
- 监控预警

单点分析 | 需求单点分析

指标项: 产研实施周期 数据源: 需求及卡片完成 按需求完成时间

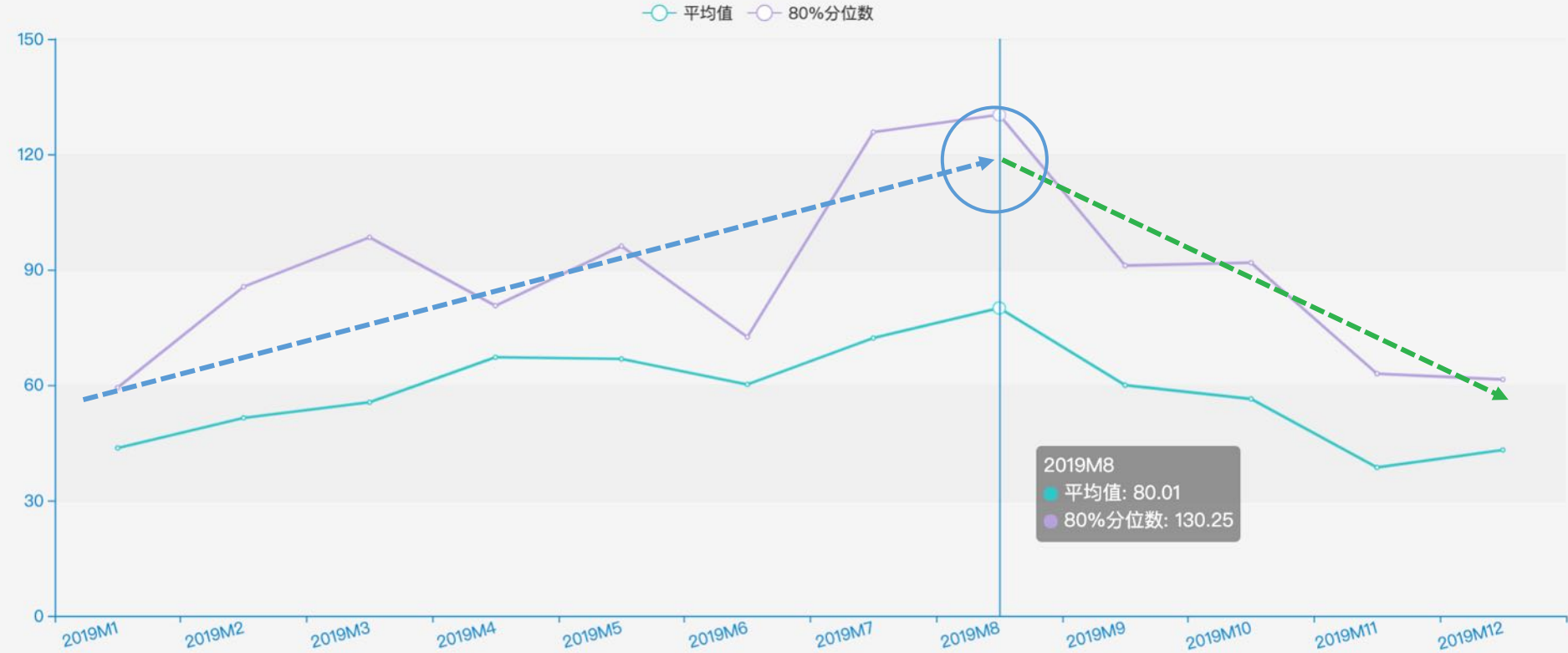
时间范围:

统计维度(X轴): 月趋势 需求来源: 全部 ☐ 过滤团队空间

查询 返回上级部门 导出 卡片维度导出

指标说明:

产研实施周期=卡片完成时间-需求受理时间, 需求受理时间及卡片完成时间取最晚的时间。单位(天)



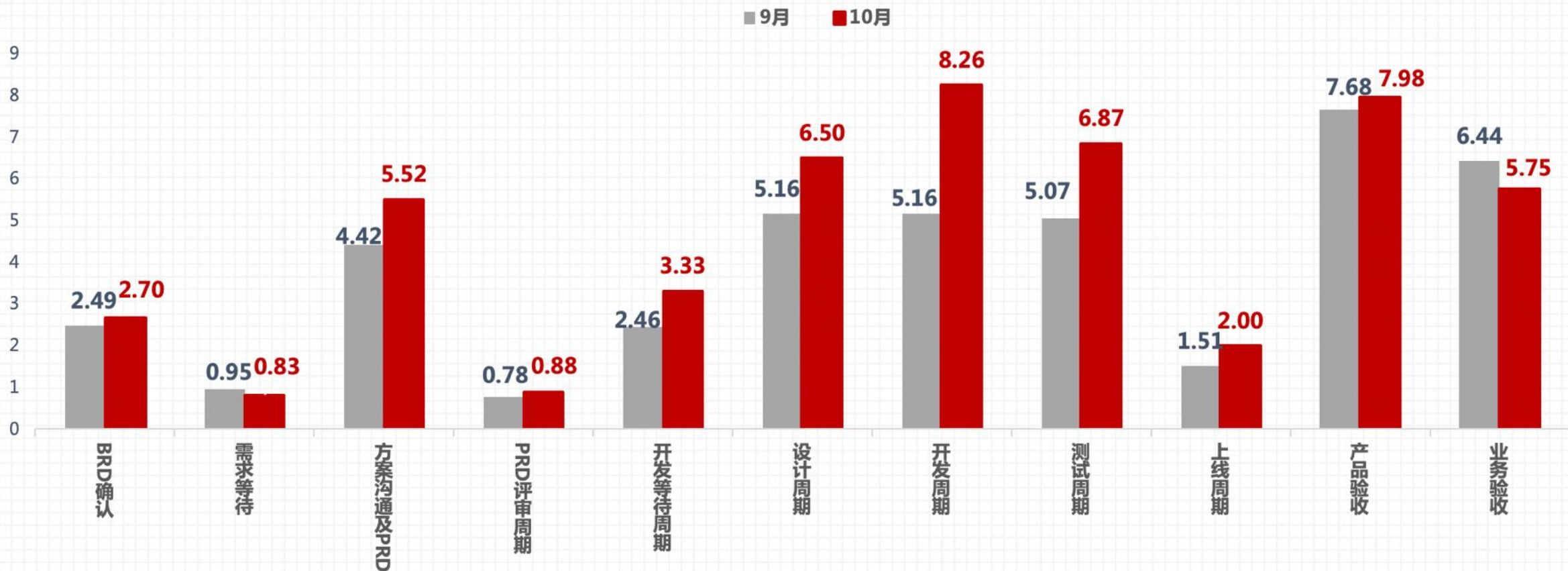
趋势分析 - 案例

- 需求交付周期从**75天** (2019年Q4数据) 缩短到**46天** (2020年Q3数据), 周期降幅**39%**, 同期需求交付数量稳步上升, Q2/Q3/Q4分别为16,527/23,001/25,398个。
- 2020年**业务满意度**得分: **3.73** (Q1)、**4.34** (Q2) 和**4.52** (Q3), 业务满意度稳步提升。

19Q4-20Q3需求交付周期和交付数量变化



需求各阶段停留时长 (天)

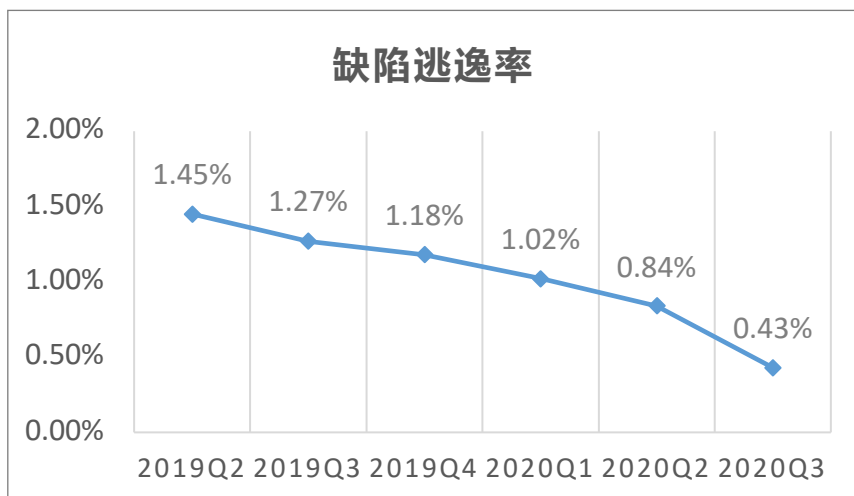


趋势分析 - 案例

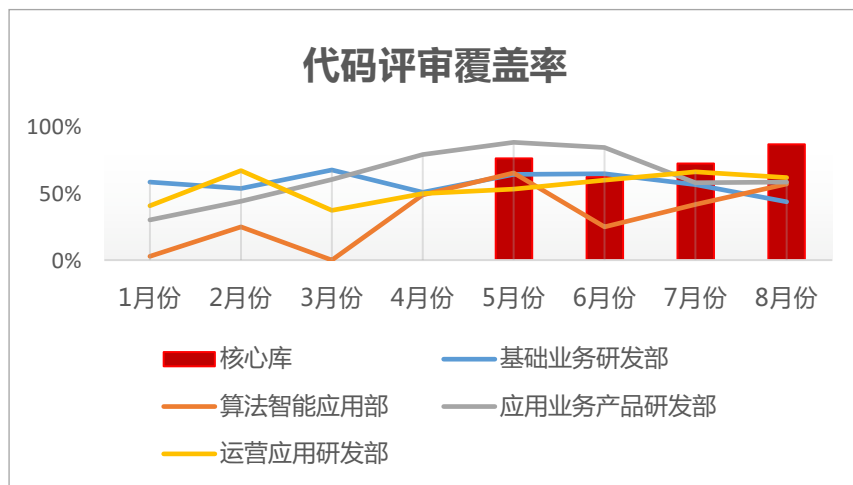


质效合一的工程能力建设：全链路研发交付工具链，研发过程质量持续提升

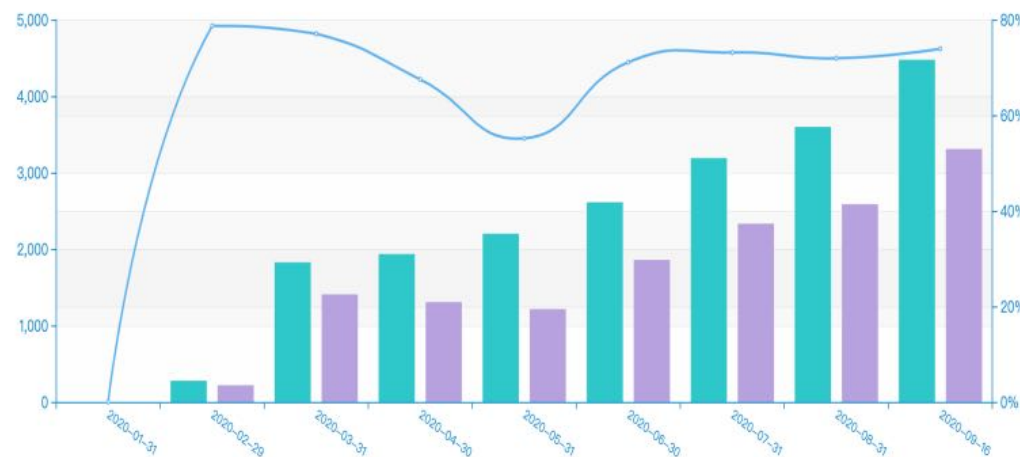
- 平台缺陷逃逸率降幅200%，Q3达到历史新低



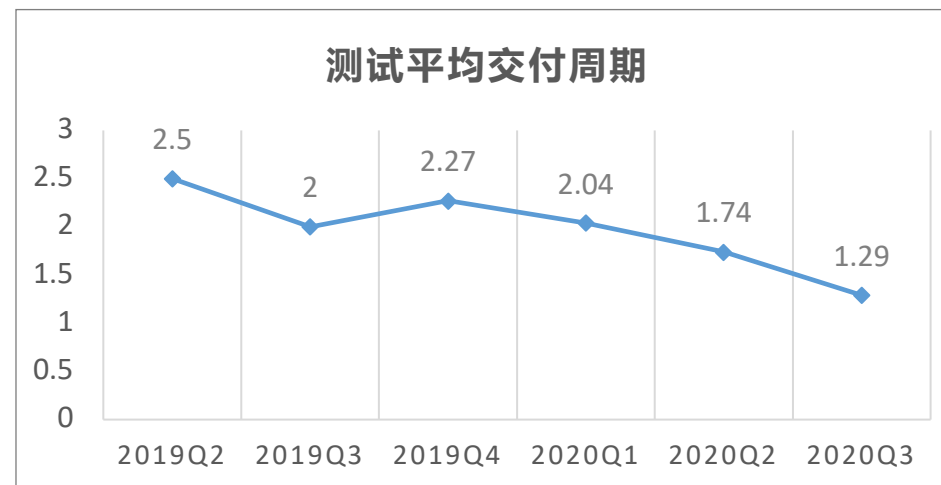
- 研发核心代码评审覆盖率稳定80%



- 摸索符合公司实际的工程实践方法，核心单元测试数量突破5000，通过率稳定80%



- 测试平台化能力建设，自动化助力测试交付周期缩短100%



需求前置时间

趋势分析

周/月/季度趋势
同比环比分析
效能改进决策点

下钻分析

关联分析

需求规模、需求吞吐量
线上缺陷密度（交付质量）
关联代码质量（技术债务）

按部门下钻

按组织结构逐层下钻

按团队下钻

按产品/特性团队下钻

按阶段下钻

按阶段寻找低效瓶颈点

流效率分析

活动时间与等待时间占比

流负载分析

在制品数量和队列长度

累积流图

周期分解

明细分析

需求 Requirement

数据源: 行云

需求吞吐能力统计



7093	3844
需求数	已完成
362	2885
待验收	处理中

平均交付周期统计

研发部门-卡片完成



完成卡片平均耗时

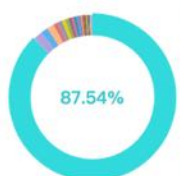
复合阶段平均耗时

工时统计

213904.5H	114714.3H
计划工时	实际工时

工时投入分布

回到上一级



6309

卡片阶段耗时

卡片吞吐量

需求分析时长
开发处理时长
测试发布时长

more >

测试 QA

数据源: 穹天

平均测试交付统计

more >

1.26天	5911
提测单执行时长	提测单数

自动化测试统计

more >

28.41%	99.63%
用例覆盖率	用例通过率

其它统计

more >

100%
提测成功率

1 %
缺陷逃逸率

缺陷 Issue

数据源: jira

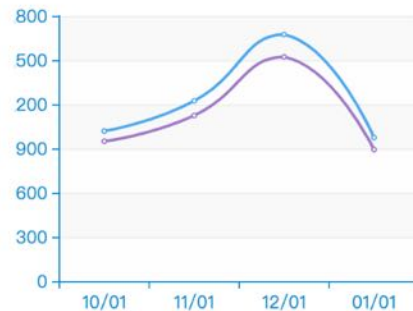
缺陷统计

more >

4888	92%	104.81H
新增缺陷	解决率	平均解决时长

新增Bug/Fix统计

新增Bug FixBug

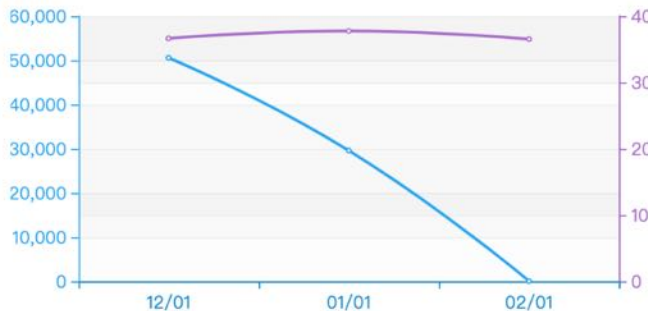


代码 Code

数据源: Coding

代码提交/评审覆盖率

代码提交 评审覆盖率



代码评审相关统计



发布 Release

数据源: J-one

发布信息统计

more >

46/天	14.44H
发布频率	前置时间

发布/成功统计

新发布 发布成功



运维 Operations

数据源: 西格玛

崩溃率统计

6‰	3‰
IOS崩溃率	Android崩溃率

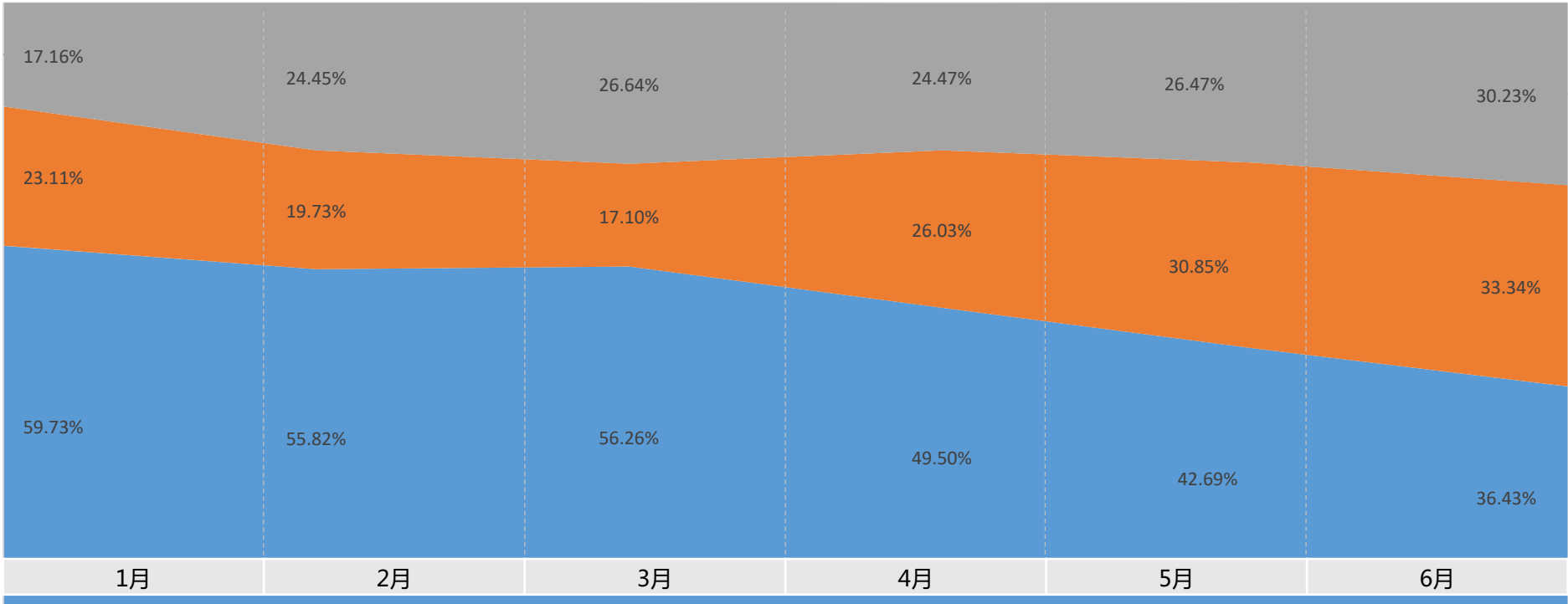
6.3‰

线上缺陷密度

不仅要关注创造价值的工作，还要关注保护价值的工作



各类别需求分布



■ 业务需求 ■ 产品需求 ■ 研发需求

月份	1月	2月	3月	4月	5月	6月
业务需求	59.73%	55.82%	56.26%	49.50%	42.69%	36.43%
产品需求	23.11%	19.73%	17.10%	26.03%	30.85%	33.34%
研发需求	17.16%	24.45%	26.64%	24.47%	26.47%	30.23%

需求前置时间

趋势分析

周/月/季度趋势
同比环比分析
效能改进决策点

下钻分析

关联分析

需求规模、需求吞吐量
线上缺陷密度（交付质量）
关联代码质量（技术债务）

按部门下钻

按组织结构逐层下钻

按团队下钻

按产品/特性团队下钻

按阶段下钻

按阶段寻找低效瓶颈点

流效率分析

活动时间与等待时间占比

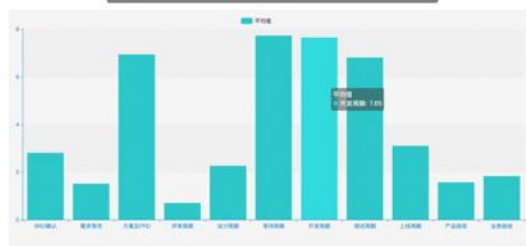
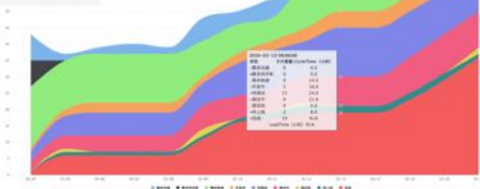
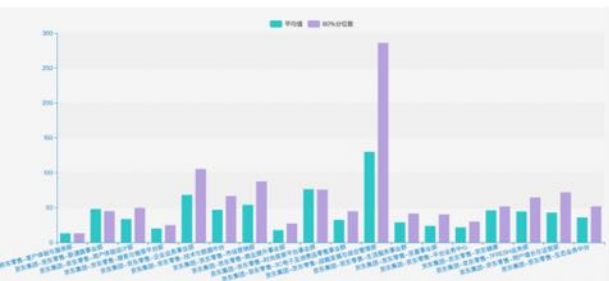
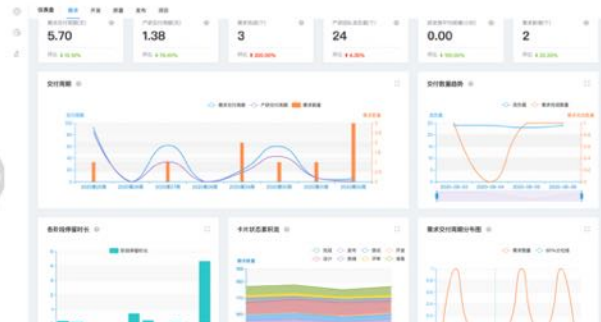
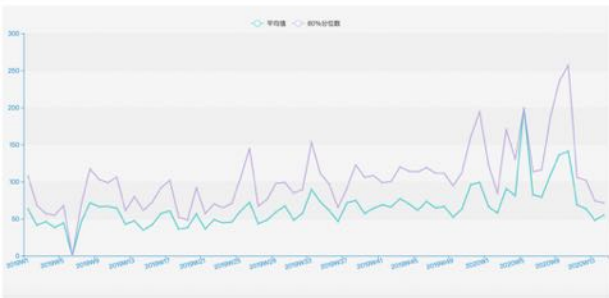
流负载分析

在制品数量和队列长度

累积流图

周期分解

明细分析



- 全链路仪表盘
- 自定义仪表盘
- 单点分析
- 需求单点分析
- 研发部门视角
- 受理部门视角
- 代码库单点分析
- 关联分析
- 全链路关联分析
- 问题诊断分析
- 设置
- 监控预警

单点分析 | 需求单点分析

指标项: 产研交付周期

数据源: 需求及卡片完成

按需求完成时间

时间范围:

统计维度(X轴): 同级部门平均

需求来源: 全部

☐ 过滤团队空间

查询

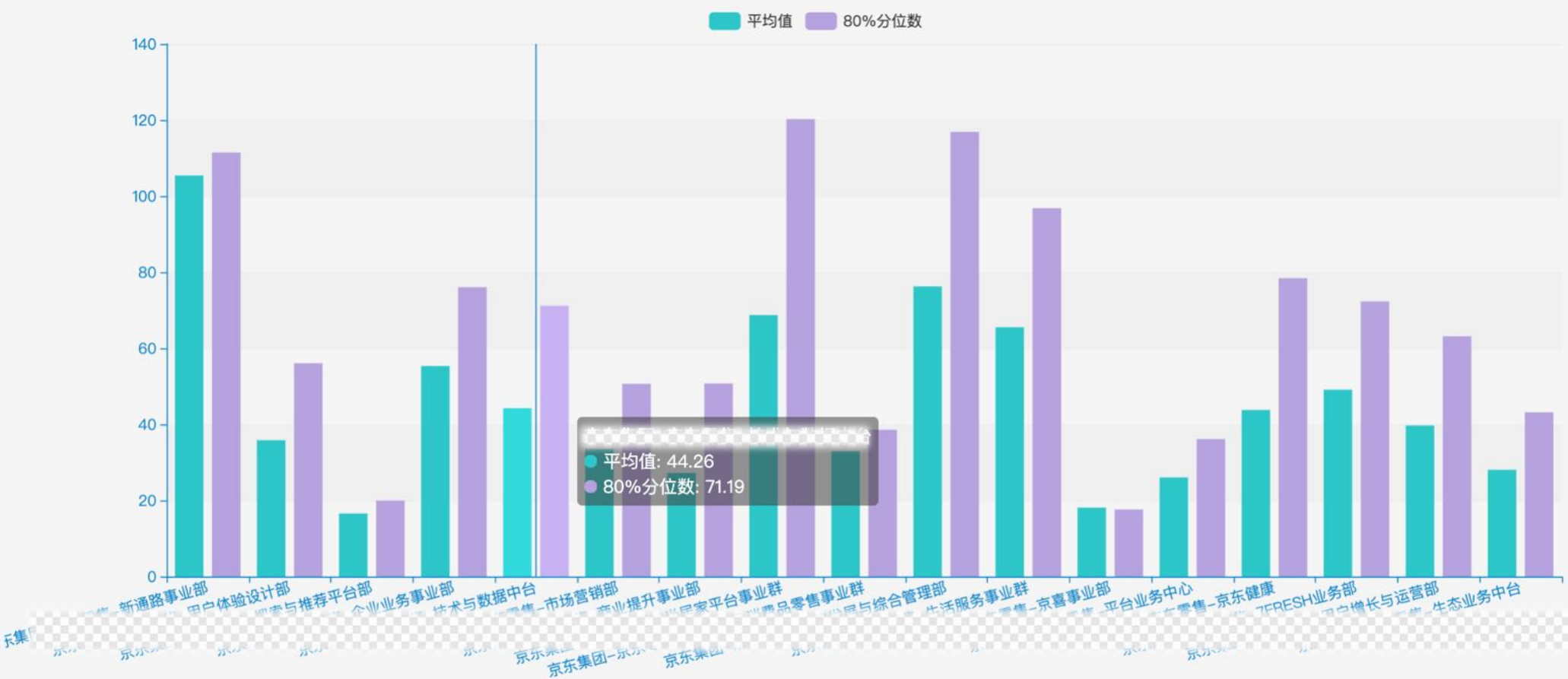
返回上级部门

导出

卡片维度导出

指标说明:

产研交付周期=卡片完成时间-需求提出时间, 需求提出时间及卡片完成时间取最晚的时间。单位(天)



需求前置时间

趋势分析

周/月/季度趋势
同比环比分析
效能改进决策点

下钻分析

关联分析

需求规模、需求吞吐量
线上缺陷密度（交付质量）
关联代码质量（技术债务）

按部门下钻

按组织结构逐层下钻

按团队下钻

按产品/特性团队下钻

按阶段下钻

按阶段寻找低效瓶颈点

流效率分析

活动时间与等待时间占比

流负载分析

在制品数量和队列长度

累积流图

周期分解

明细分析

- 首页概览
- 全链路仪表盘
- 自定义仪表盘
- 单点分析
- 需求单点分析
- 研发部门视角
- 受理部门视角
- 代码库单点分析
- 关联分析
- 全链路关联分析
- 问题诊断分析
- 设置
- 监控预警



单点分析 | 需求单点分析

指标项:

实际开发周期

数据源:

卡片完成

按卡片完成时间

时间范围:

统计维度(X轴):

阶段停留时长

需求来源:

全部

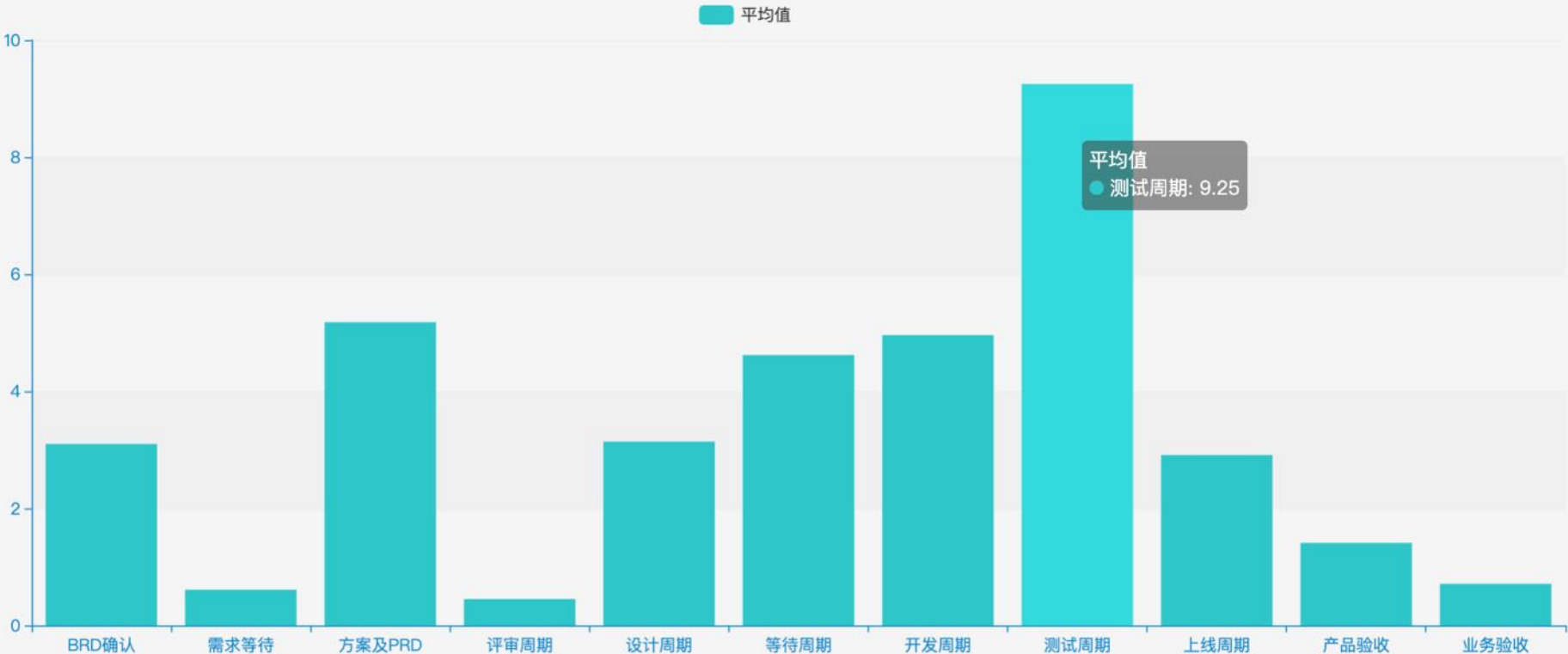
☐ 过滤团队空间

查询

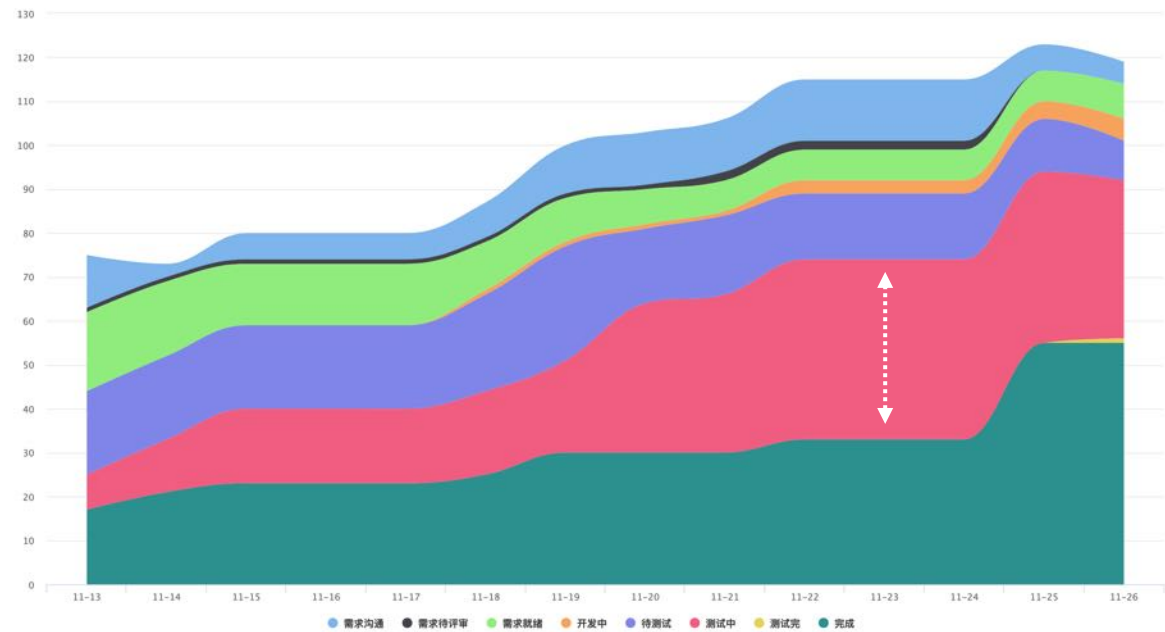
返回上级部门

导出

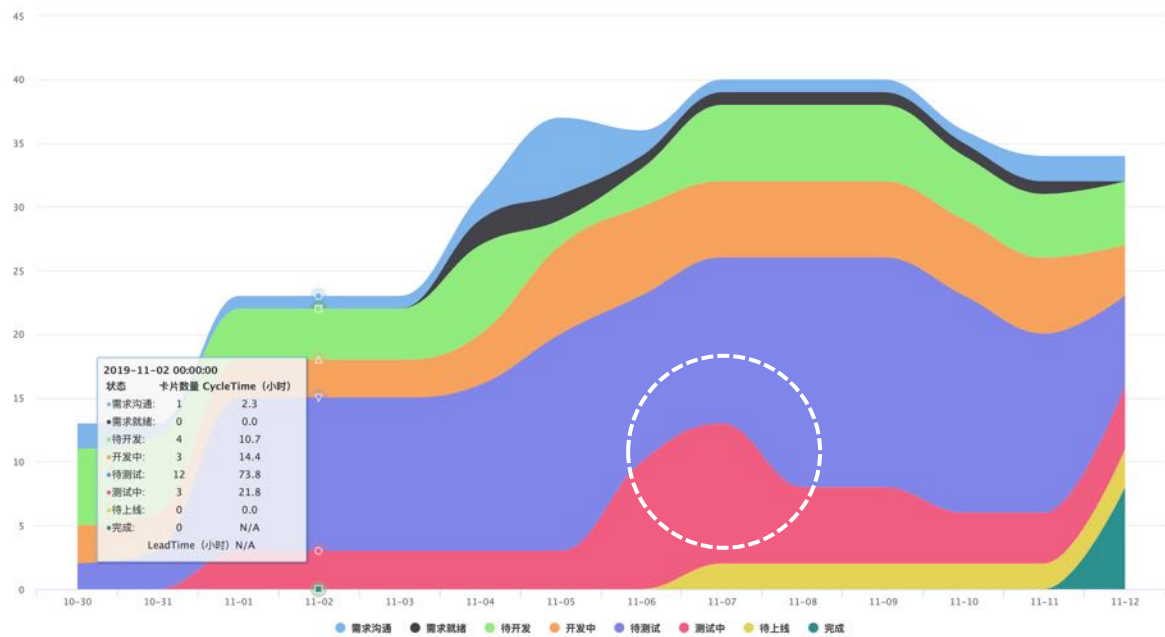
选择导出维度



卡片状态图累积流图



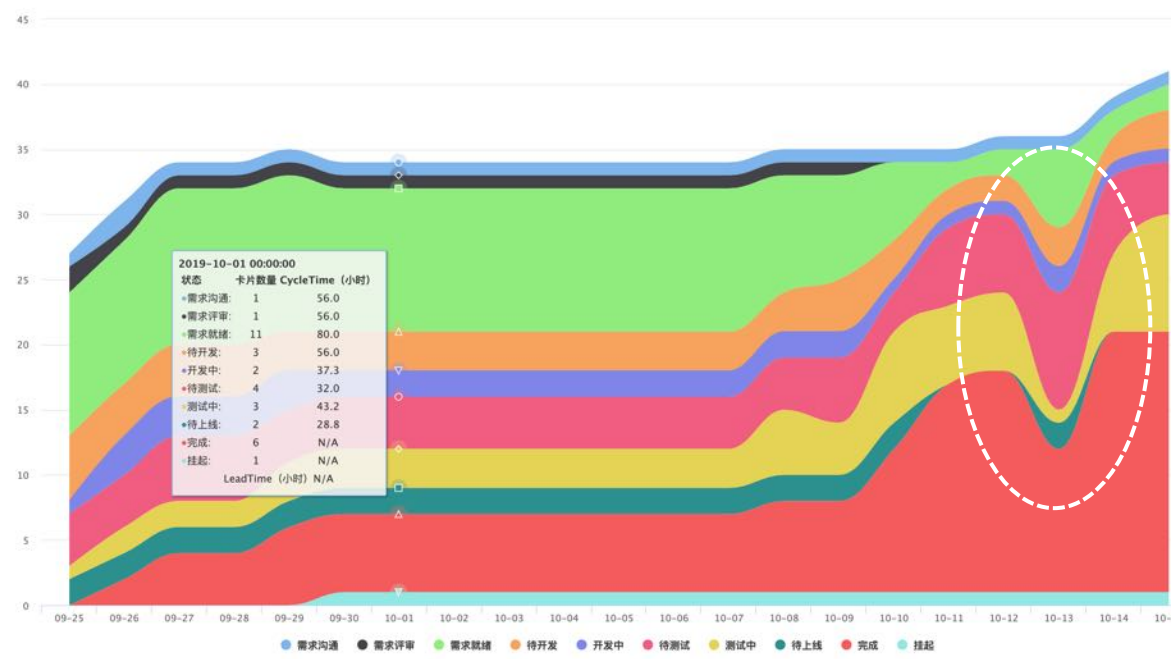
卡片状态图累积流图



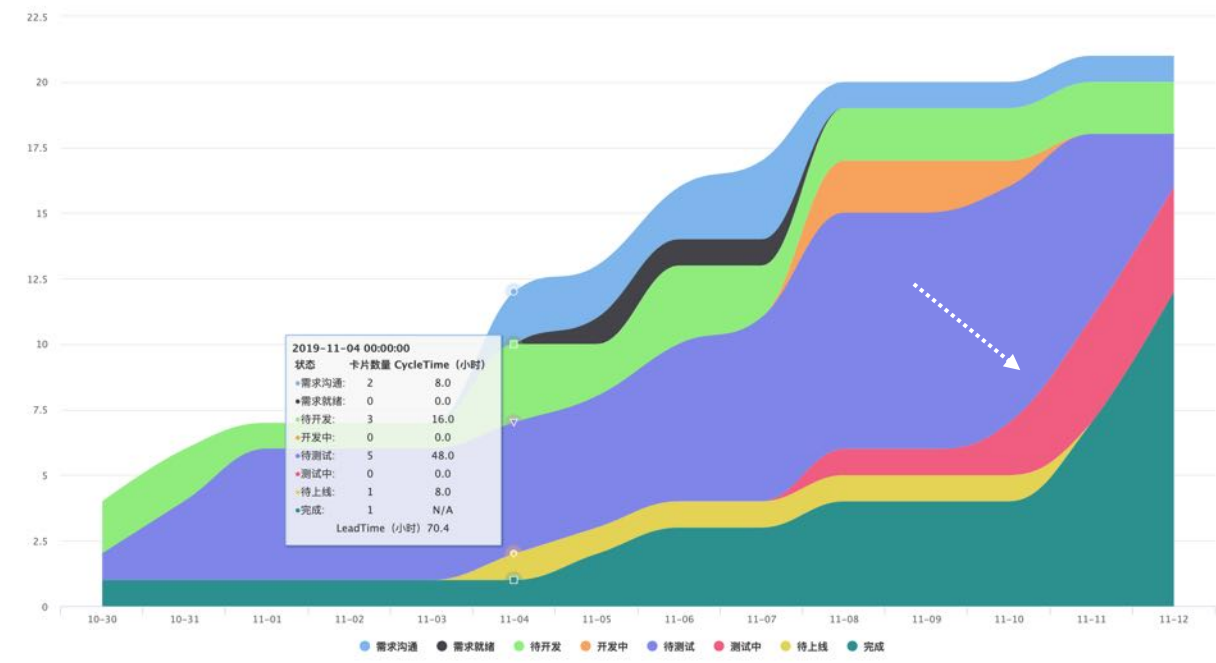
卡片状态停留时长散点图



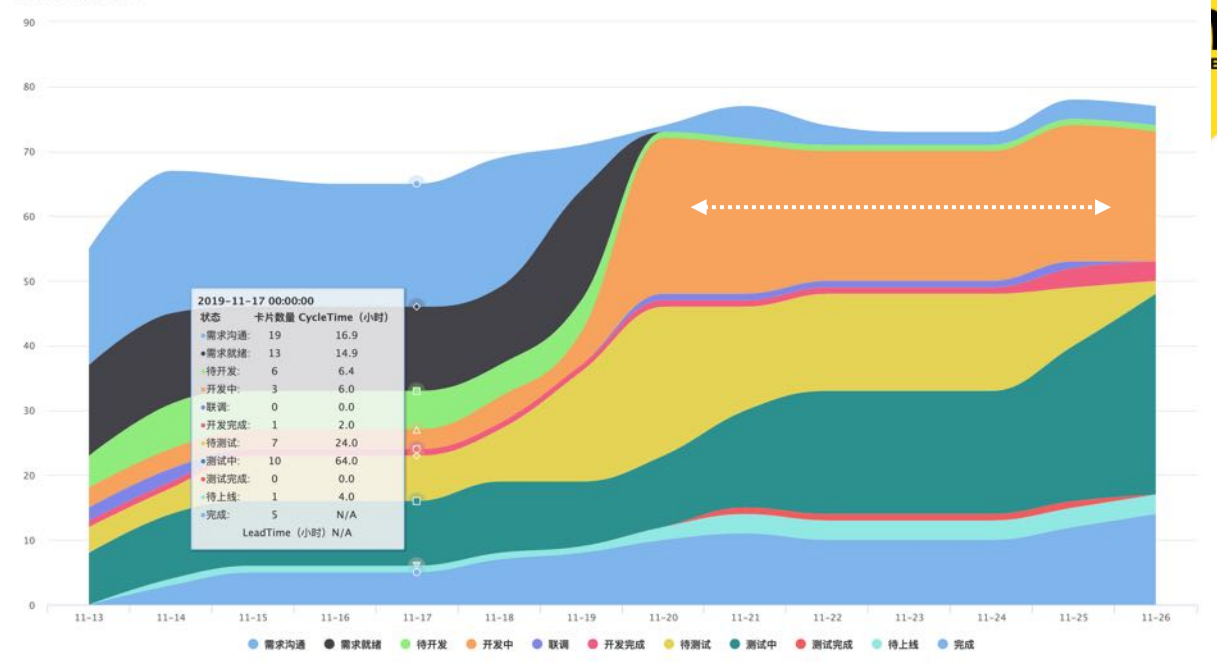
卡片状态图累积流图



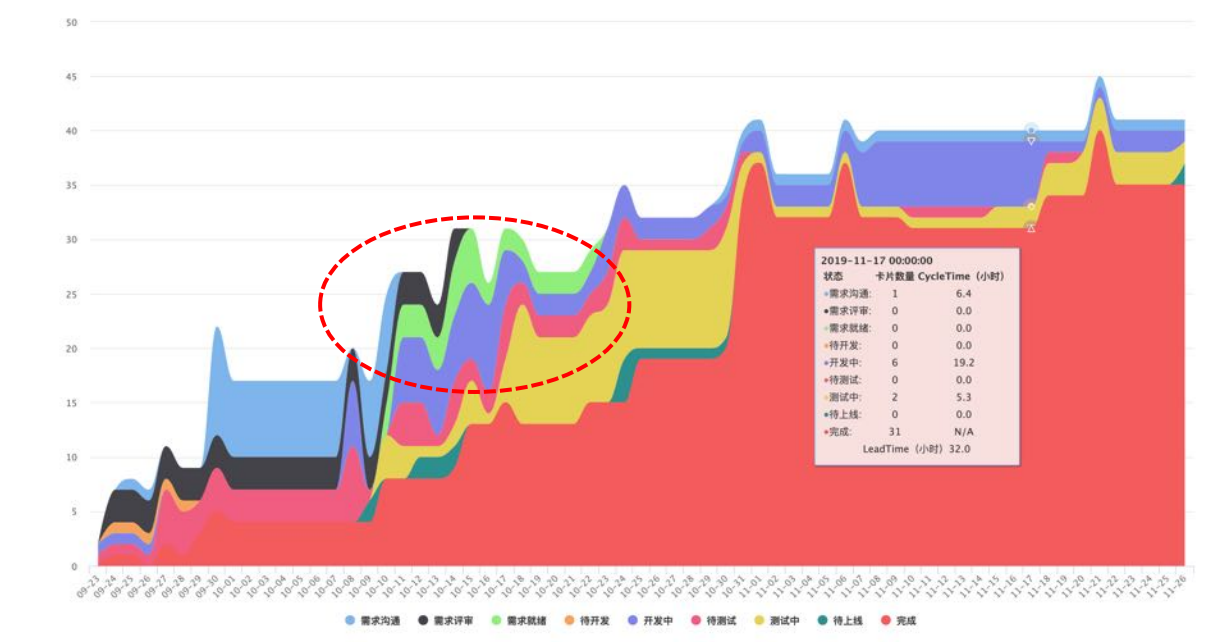
卡片状态图累积流程图



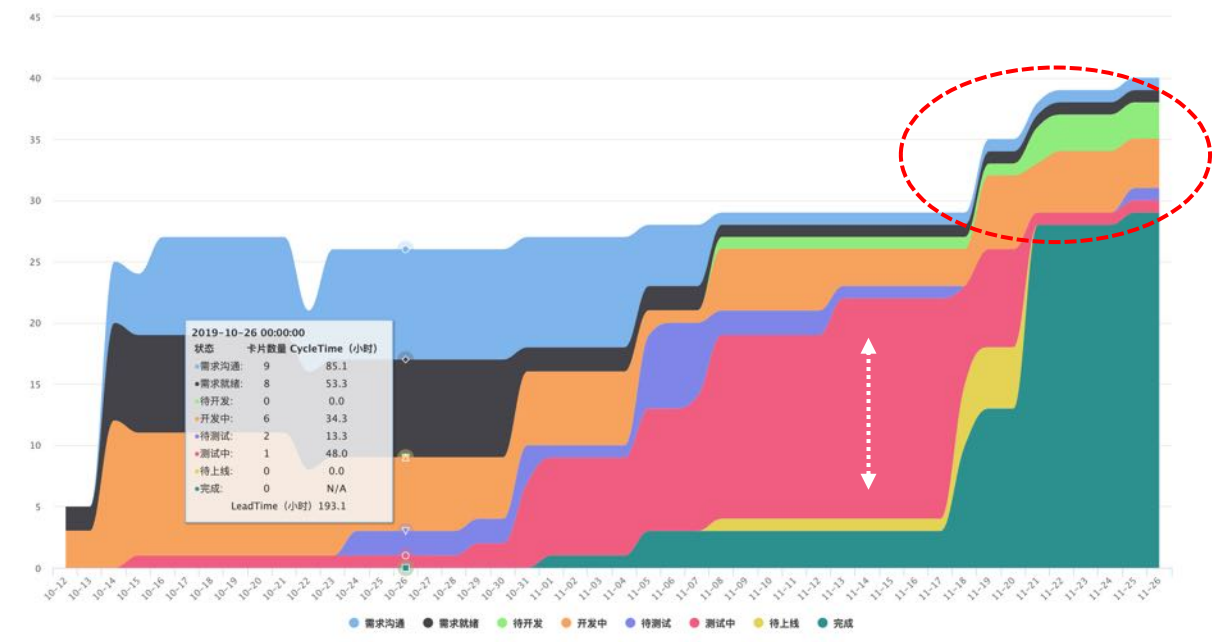
卡片状态图累积流程图



卡片状态图累积流程图



卡片状态图累积流程图



APP购物车

收起 V8510<0401-0414> (0 / 25)

燃尽图

排序

筛选

全屏

需求就绪 (0)

待开发 (1)

开发中 (2)

待测试 (17)

测试中 (5)

待上线 (0)

已集成 (0)

完成 (0)

【iOS】购物车网络层解耦降级删除-nReceiveCoupon接口

1h 03-03

0 / 1

极速版-appid、client变更逻辑适配

16h 03-18

0 / 1

服务端

倒排期

极速版-配合客户端联调验证

24h 03-23

0 / 1

服务端

倒排期

无障碍-iOS购物车单项商品焦点未按照正确顺序

7h 04-02

0 / 2

ios端

无障碍-iOS可进入店铺名称控件角标

6h 04-02

0 / 2

ios端

无障碍-iOS顶部存在一个朗读刷新按钮

8h 04-02

0 / 2

ios端

无障碍-iOS购物车界面满减信息行首的“满减”等文字未加“满减”等文字

6h 04-02

0 / 2

android端

无障碍-iOS购物车界面秒杀剩余时间未显示

6h 04-02

0 / 2

android端

无障碍-购物车-选服务，浮层中“服务介绍”旁的“i”有独立焦点

0 / 2

ios端

极速版-联调

0 / 2

ios端

明

android

极速版-删除

0 / 2

ios端

明

android

极速版-商品结算

0 / 2

ios端

明

android

极速版-商品展示处理

0 / 2

ios端

明

android

购物车商品界面店铺、商品名称的勾选框均无朗读

0 / 2

ios端

03

0 / 2

ios端

展开迭代详情

需求前置时间

趋势分析

周/月/季度趋势
同比环比分析
效能改进决策点

下钻分析

关联分析

需求规模、需求吞吐量
线上缺陷密度（交付质量）
关联代码质量（技术债务）

按部门下钻

按组织结构逐层下钻

按团队下钻

按产品/特性团队下钻

按阶段下钻

按阶段寻找低效瓶颈点

流效率分析

活动时间与等待时间占比

流负载分析

在制品数量和队列长度

累积流图

周期分解

明细分析

需求交付周期(天)

33.72

环比 ↑ 11.99%

产研交付周期(天)

18.07

环比 ↑ 19.83%

需求完成(个)

292

环比 ↓ 5.19%

产研团队流负载(个)

159

环比 ↑ 43.24%

研发侧平均规模(小时)

26.75

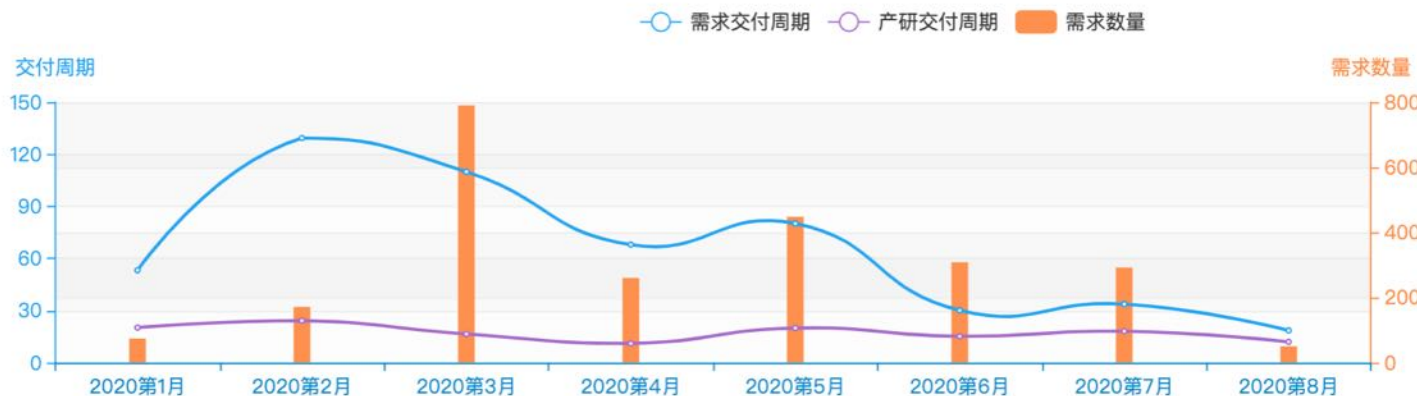
环比 ↓ 7.96%

需求新增(个)

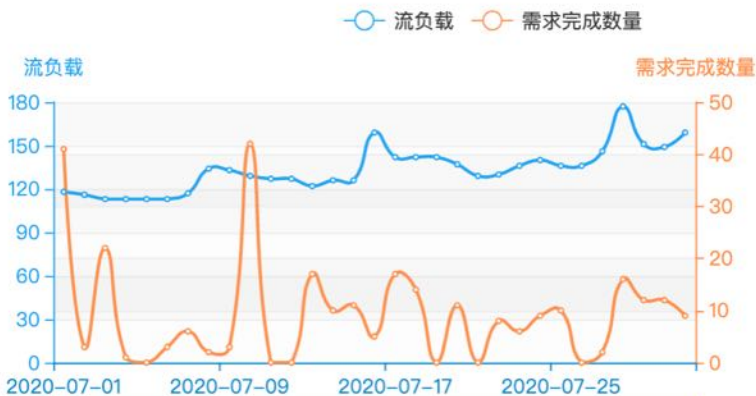
390

环比 ↑ 54.76%

交付周期



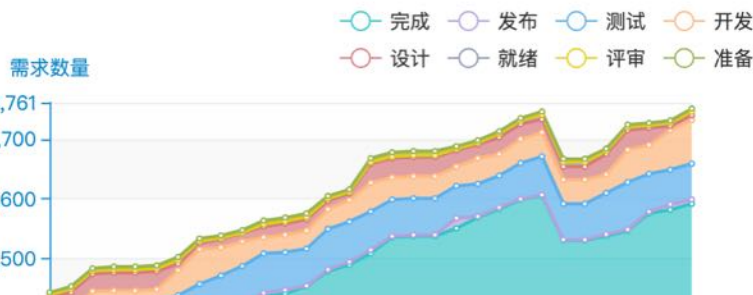
交付数量趋势



各阶段停留时长



卡片状态累积流



需求交付周期分布图



- 1 业界研发效能的情况
- 2 研发效能度量的误区
- 3 研发效能度量体系模型
- 4 研发效能度量驱动改进
- 5 经验分享和避坑指南

研发效能度量经验总结 – 通过自动化降低度量带来的额外成本

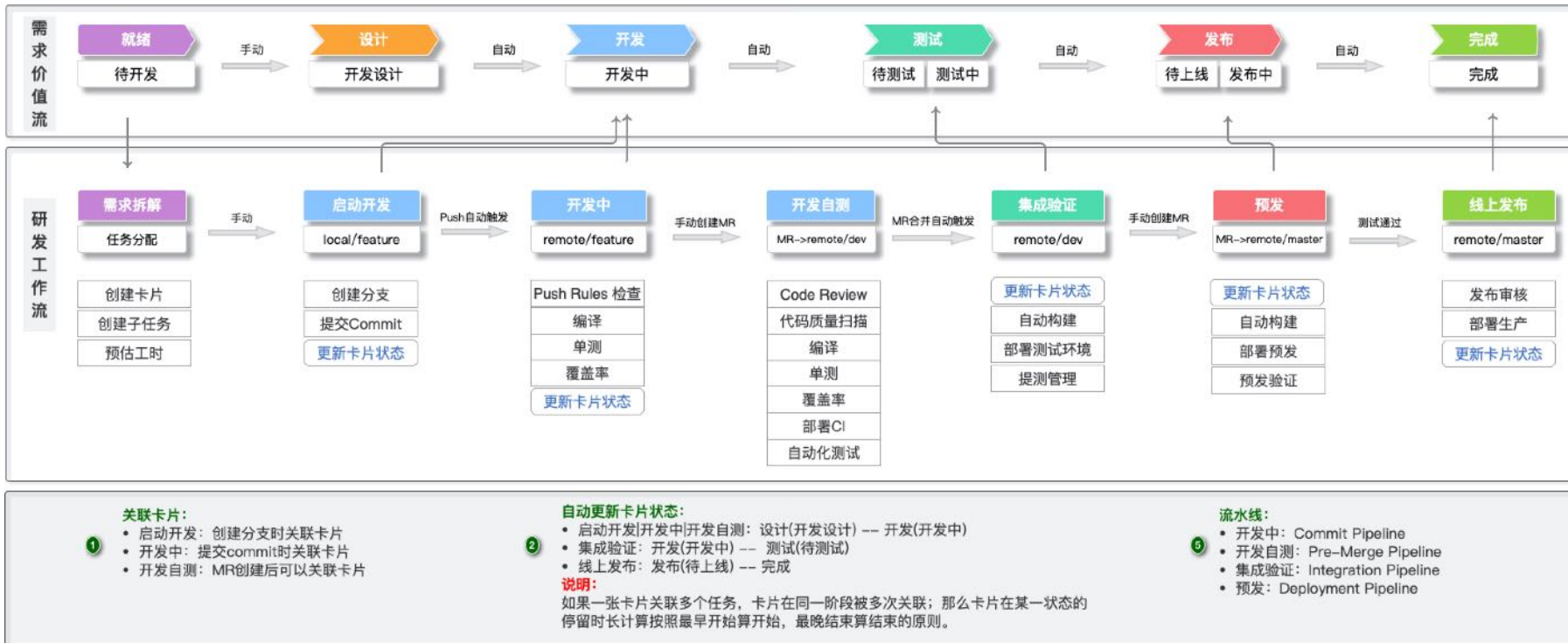
流程规范性

度量的准确性依赖流程的规范性
为了保证度量指标准确性，需要明确研发流程、制定相应规范，并确保相关的活动都在系统中完整记录

度量成本

度量有成本，不是免费的！
项目管理人员、产品经理、工程师需投入时间遵循流程规范，大量统计数据收集、分析和汇报都有成本

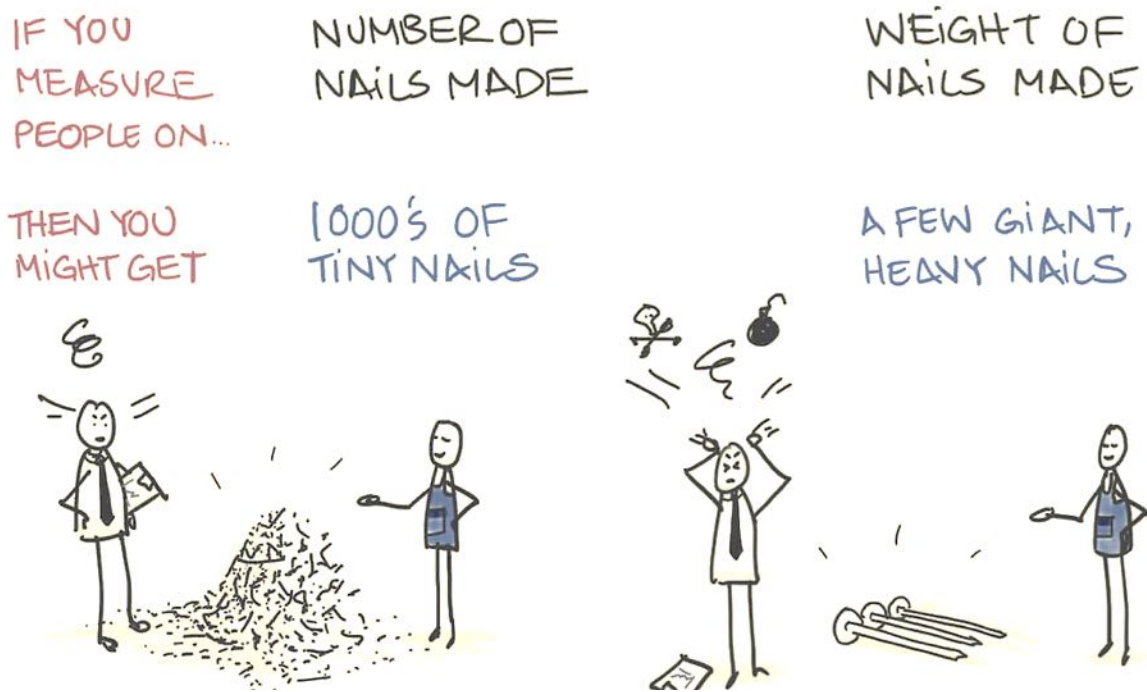
通过统一工程效能平台固化流程与活动，减少工程师繁琐操作，尽量自动化



度量不是为了控制，而是指导改进

古德哈特法则

当某个度量变成了目标，它便不再是一个好的度量



DevOps 团队

100
CASE STUDIES

目标对齐

自组织

跨职能

学习型

敏捷和精益产品开发实践

价值流管理

可视化管理

小批量交付

限制在制品

需求拆分和实例化

敏捷迭代规划

轻量级变更流程

持续实验和反馈

度量驱动持续改进

持续交付工程技术实践

分支策略

版本控制

自动化测试

测试数据管理

自动化部署

测试环境管理

安全左移

数据库变更管理

架构解耦

仿真模拟器

持续集成

低风险发布技术

代码可维护性

监控/可观测性

云原生 IT 基础设施

服务和流量管理

生命周期管理

云服务依赖管理

K8S 容器编排



People who can't understand numbers are useless. The Gemba where numbers are not visible is also bad. However, people who only look at the numbers are the worst of all.

— Taiichi Ohno —

那些不懂数字的人是糟糕的，
而那些只看数字的人是最最糟糕的





微信官方公众号：壹佰案例
关注查看更多年度实践案例