GOPS2017
Shenzhen

GOPS

2017 全球运维大会

深圳站

指导单位：数据中心联盟 Data Center Alliance

主办单位：高效运维社区 GreatOPS Community　开放运维联盟 GOPSA Open OPS Alliance

DevOps可视化在电信企业的应用

顾宇 ThoughtWorks高级咨询师

ThoughtWorks 高级咨询师，9年工作经验。加入ThoughtWorks之前曾经参与中国移动，中国联通省级BOSS系统以及呼叫中心系统的研发、实施和割接。担任过售前，项目经理，维护工程师和开发工程师等职务。拥有丰富的大型系统开发及维护的实战经验。

加入ThoughtWorks之后，接触敏捷并在不同的敏捷项目中担任了不同的角色。现在专注于DevOps、微服务相关领域的咨询和实施。

于2015年和ThoughtWorks同事共同翻译并出版《七周七Web框架》。

目录

DevOps 原本的问题 —— 相互甩锅

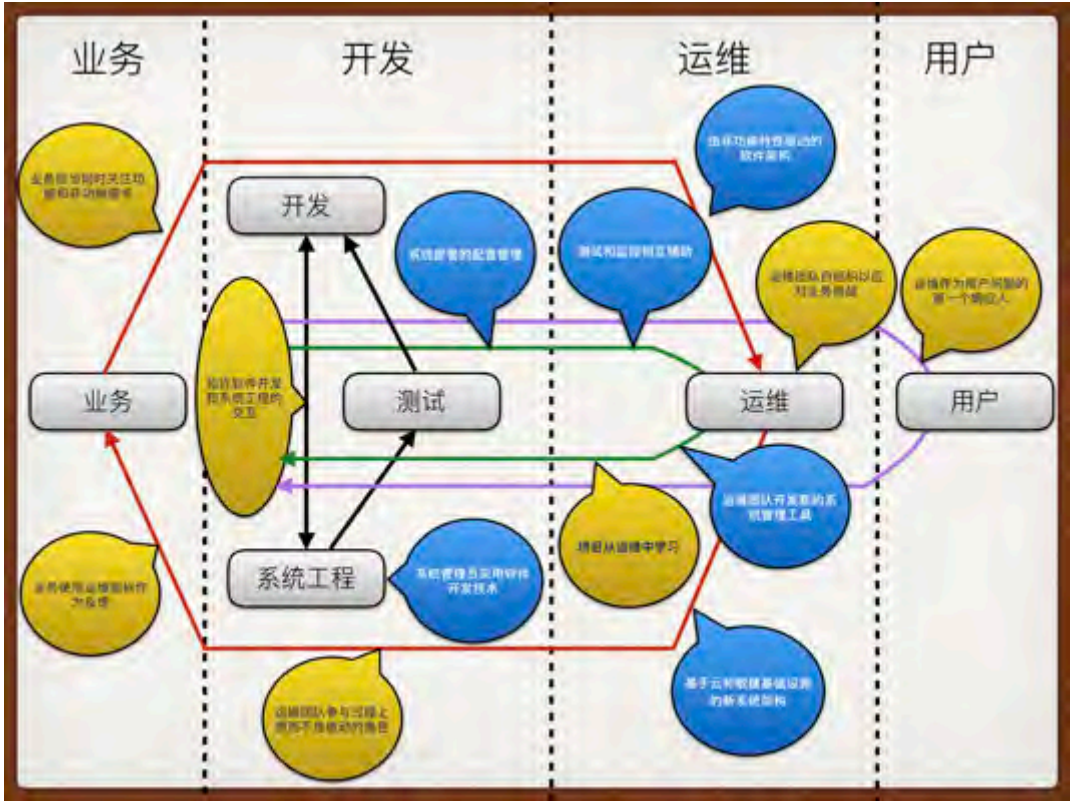
Dev　　Ops

DevOpsDays —— 共同背锅界的吐槽大会

DevOps —— 如何优雅的背锅？

DevOps 原本的答案 —— 共同背锅

DevOps —— 如何优雅的背锅？



DevOps 的核心

以快速交付业务价值为目标，
通过技术升级和流程改进，
减少业务价值交付流程中的浪费。

DevOps实践中的一些问题

DevOps 忽视组织改进

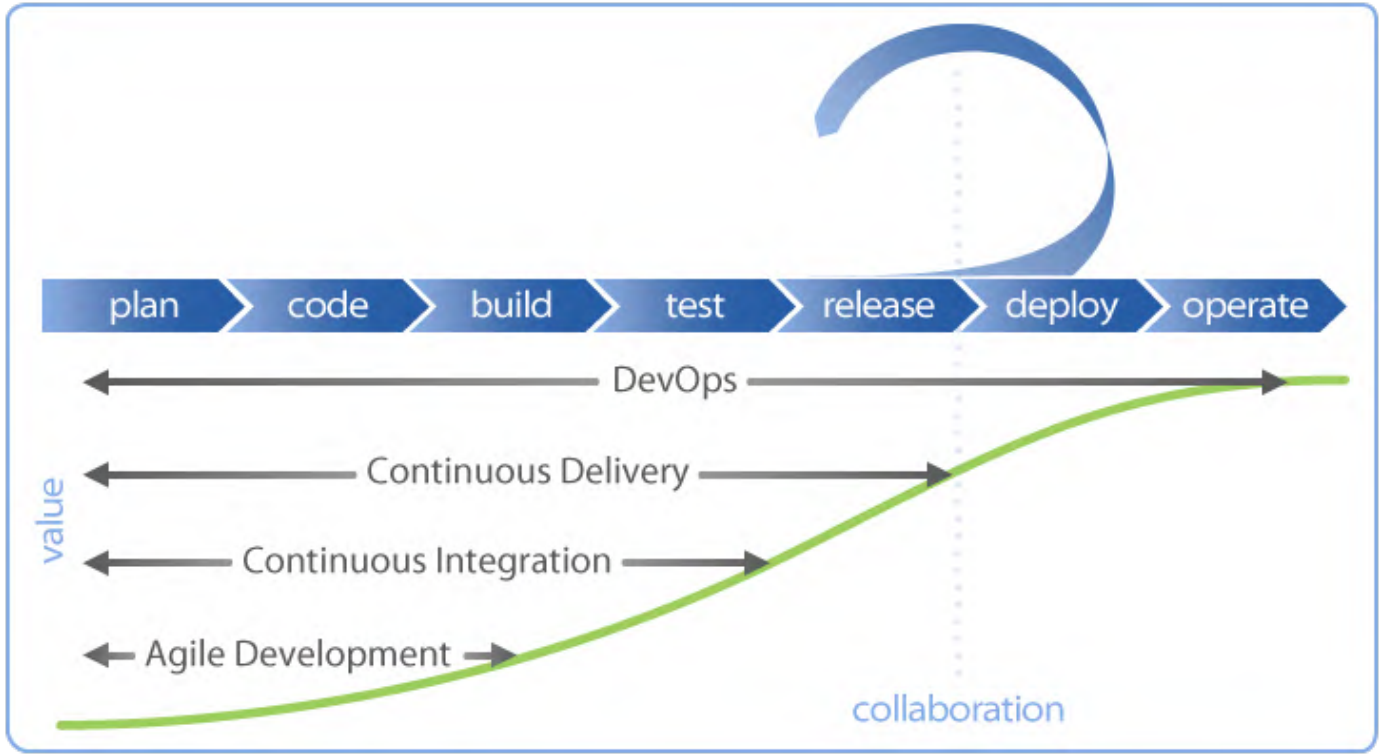
康威定律：设计系统的组织，其产生的设计和架构等价于组织间的沟通结构。

工具最后都变成了玩具

扔掉了质量，单独追求效率

人员技能培养和团队建设

DevOps 缺乏度量和约束分析



DevOps 缺乏度量和约束分析

“对非约束点的一切改进都是假象”——《凤凰项目》

DevOps 缺乏技术栈管理

DevOps 缺乏技术栈管理

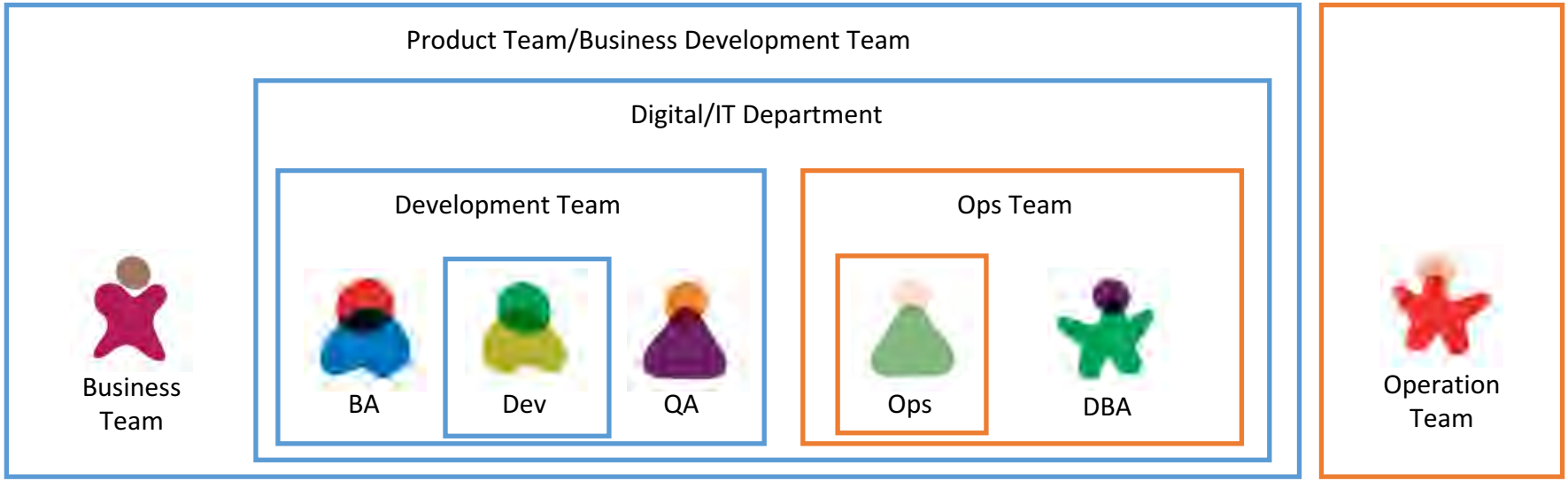
不知道选择什么样的技术

不知道如何进行技术升级

不知道培养哪方面的能力

不知道招聘什么样的人

DevOps 在不同的上下文里，意义是不同的



DevOps 是一种组织特质

不（仅仅）是把 Dev 和 Ops 放在一个团队

不（仅仅）是一个职位

不（仅仅）是一组技术

不（仅仅）是自动化运维

如何判断 DevOps 转型是否完整

是否有 DevOps 团队建设

是否有责任边界变动

是否有技术升级

是否处于不断改进的状态

目录

神奇的 DevOps 在哪里 ？

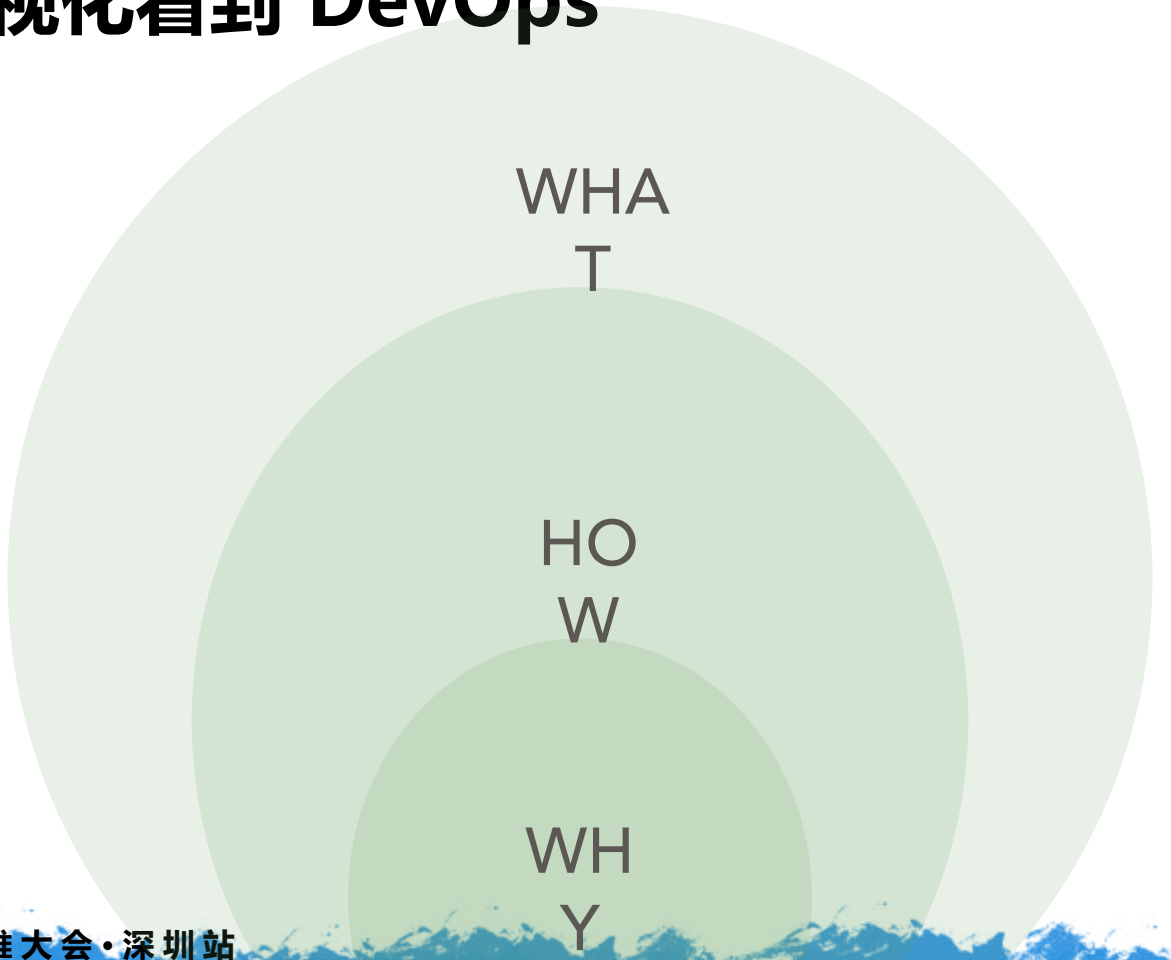
为什么需要可视化？

人们只关注看得见的东西。看不见的东西，人们往往认为不存在。

人们会选择性的接收信息。只会看到或者听到自己想看到或者听到的部分。

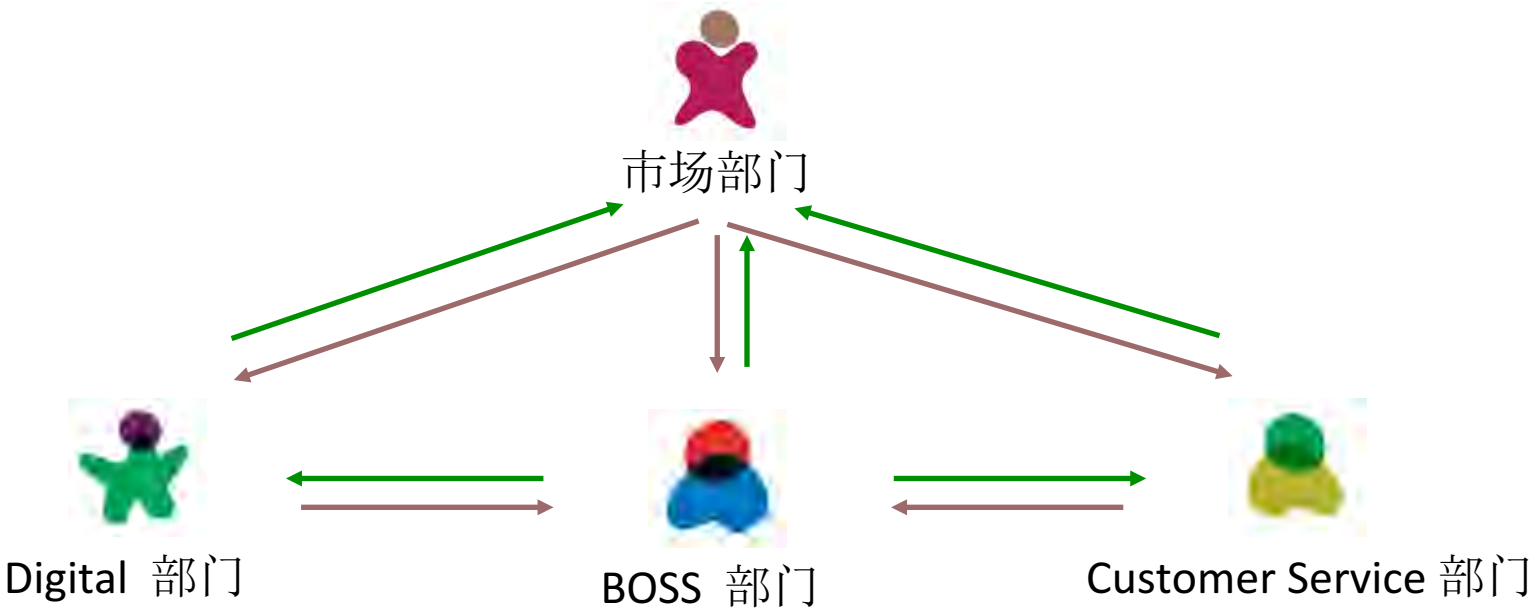
看见，是改变的开始

通过可视化看到 DevOps



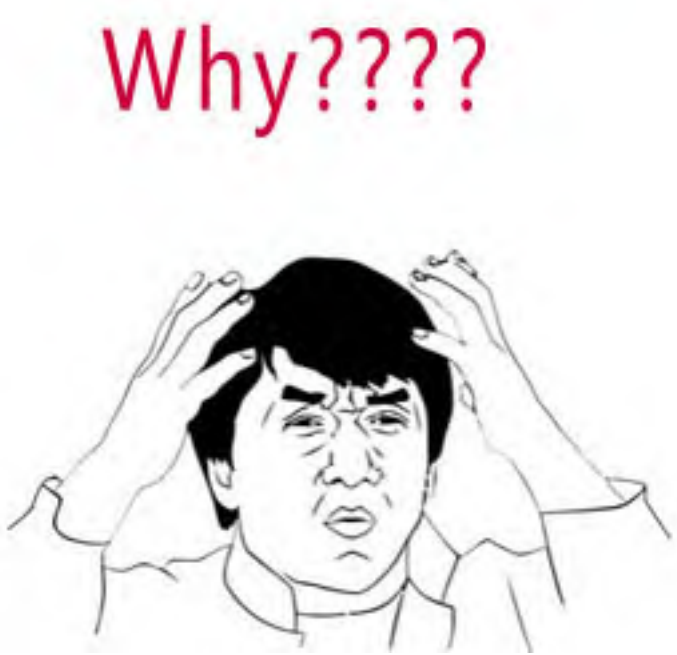
目录

一般电信企业的组织上下文



案例一：Digital部门内部DevOps

案例一：Digital部门内部DevOps



案例一：Digital部门内部DevOps

反模式：分离的DevOps团队

案例一：Digital部门内部DevOps

Digital Boss

Manager A

Manager B

DevOps
Team

Ops Ops

Development Team A

BA Dev QA

Development Team B

BA Dev QA

+

Ops Ops

案例一：Digital部门内部DevOps

Digital Boss

Manager A

Manager B

DevOps Team

Ops Ops

Development Team A

BA Dev QA Ops

Development Team B

BA Dev QA Ops

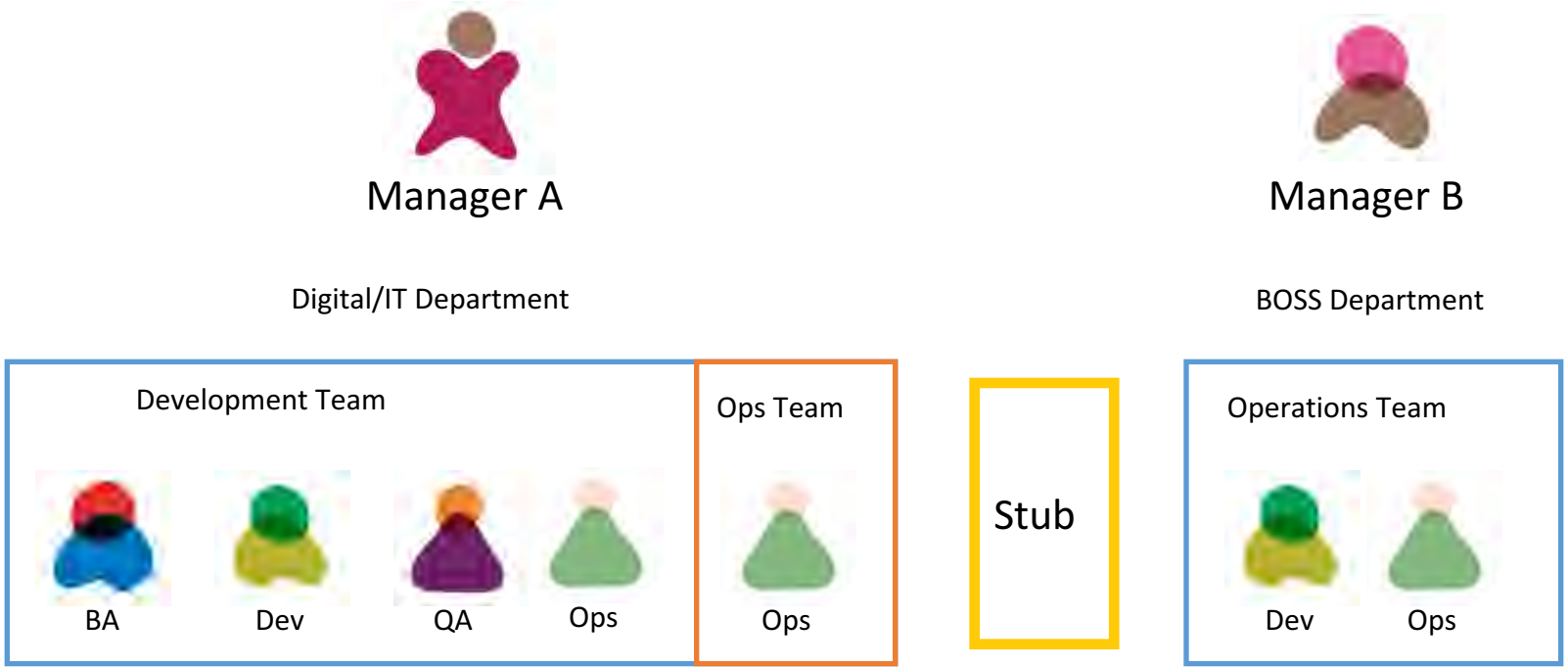
案例一：Digital部门内部DevOps

具备 DevOps 特性的团队应该是一个全功能团队

DevOps 团队可以是一个跨产品团队的虚拟团队

从资产管理视角 转向 产品演进视角

案例二：Digital部门和Boss部门的DevOps



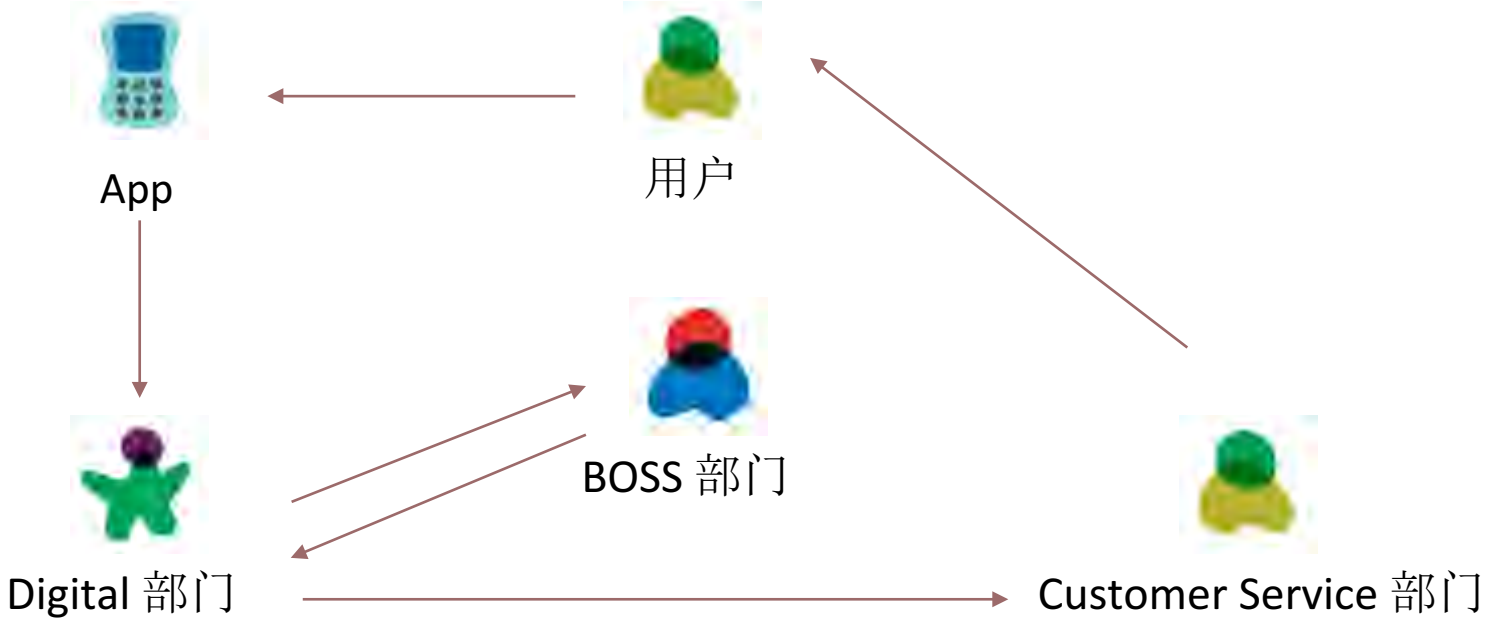
案例二：Digital部门和Boss部门的DevOps

跨部门 DevOps 组织变更之前要进行技术变革

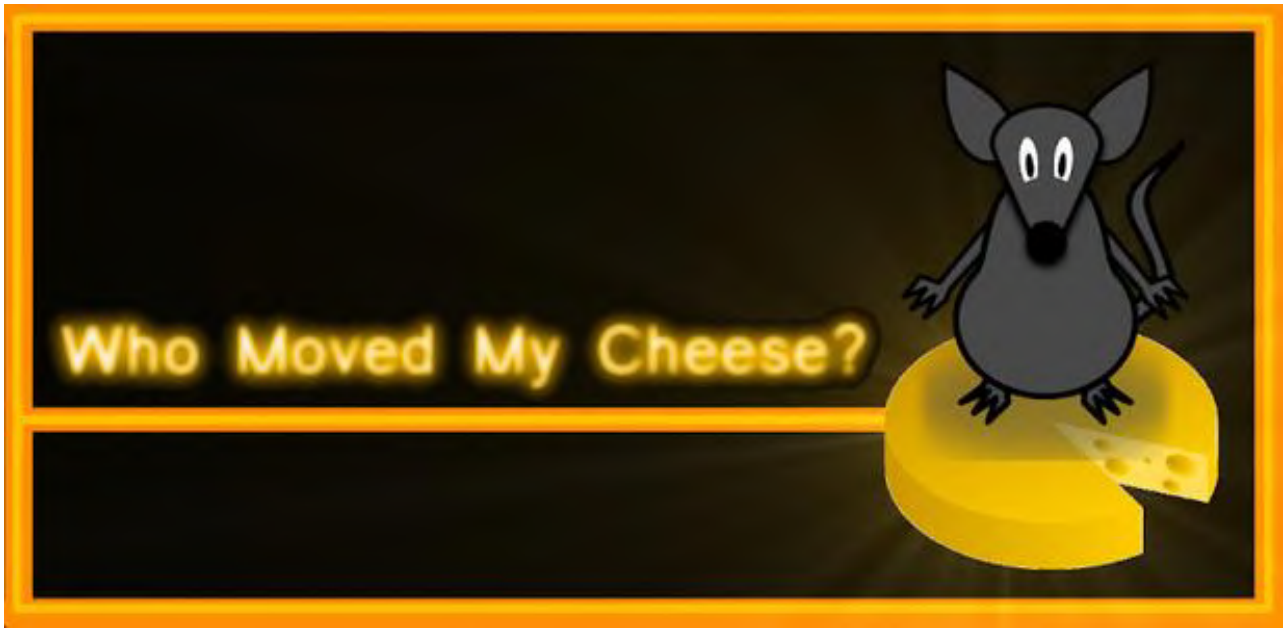
构建部门间“差速轮”

构建沟通机制和责任共担机制

案例三：Digital部门，Boss部门和Customer Service部门



动别人的奶酪



为什么需要看到 DevOps 的组织上下文

DevOps 转型困难的在于制度，而非技术。

看到在 DevOps 组织中谁是 Dev， 谁是 Ops 。

明白自己所能触及的改进范围。

基于技术的制度转型都是幻想，逃脱不了康威定理。

明白权责以及给 DevOps 带来的阻碍以及责任共担所带来的益处。

如何可视化 DevOps 的组织上下文

找到 DevOps 改进的利益相关人

找到 DevOps 的组织边界

可视化全责层级关系（职级不一定可靠）

构建组织间“差速轮”——沟通协调机制

可视化 DevOps 的组织上下文的意义

DevOps 转型一开始就是自上而下的

目录

神奇的“DevOps工作”在哪里？

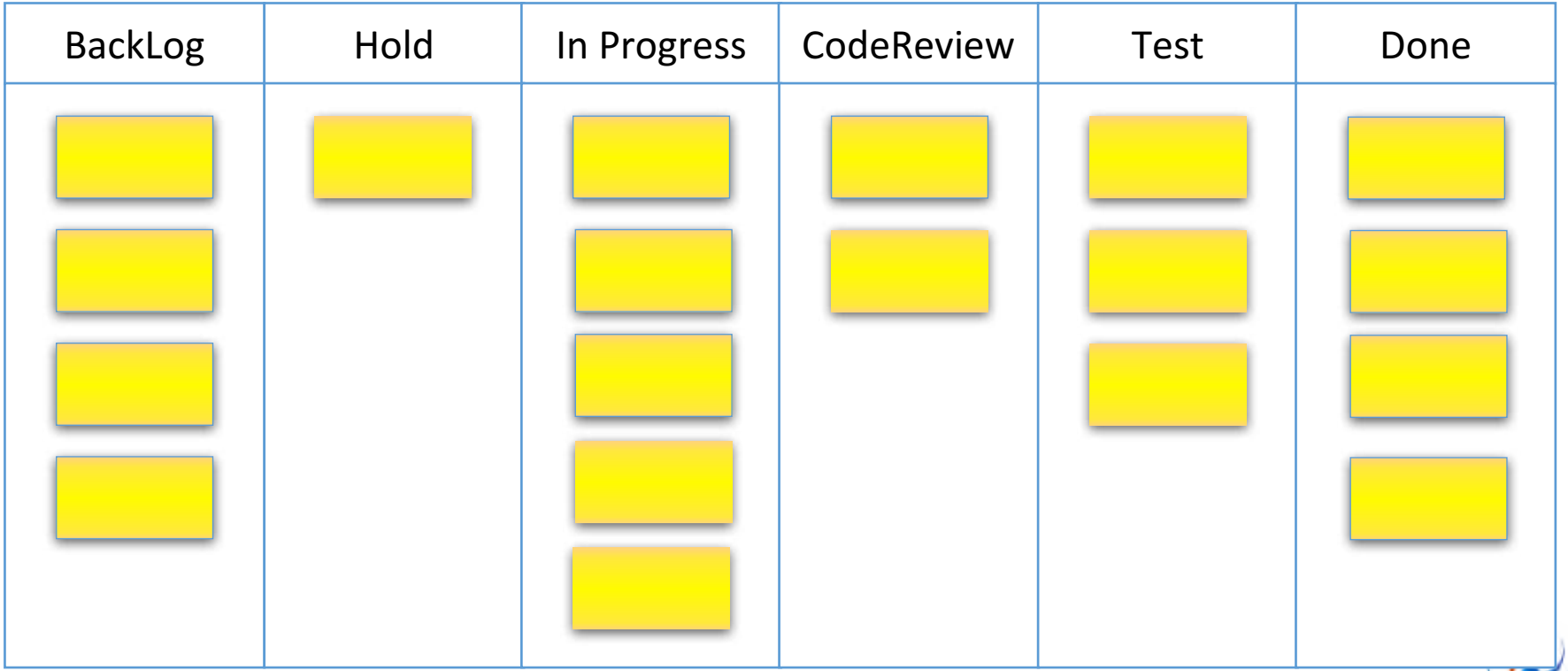
DevOps 的工作范围是什么？

DevOps 的工作目标是什么？

DevOps 的工作内容是什么？

DevOps 的工作阻碍是什么？

案例一：看不见的DevOps的工作内容



案例一：看不见的DevOps的工作内容

“DevOps 工作” 工作有什么？

“DevOps 工作” 的完成条件是什么？

“DevOps 工作” 的效果是什么？

案例一：看不见的DevOps的工作内容

Story: #213 As I User, when I click the login

3

Desc:
As a user, when I click the login button, I can

ACs:
1. User can …
2. User should …
3. User should not …
4. Ops could deploy …
5. Ops could monitor …

案例一： DevOps 工作内容的特点

Dev 不会

Dev 没有权限

Dev 不承担责任

案例一：看不见的DevOps的工作内容

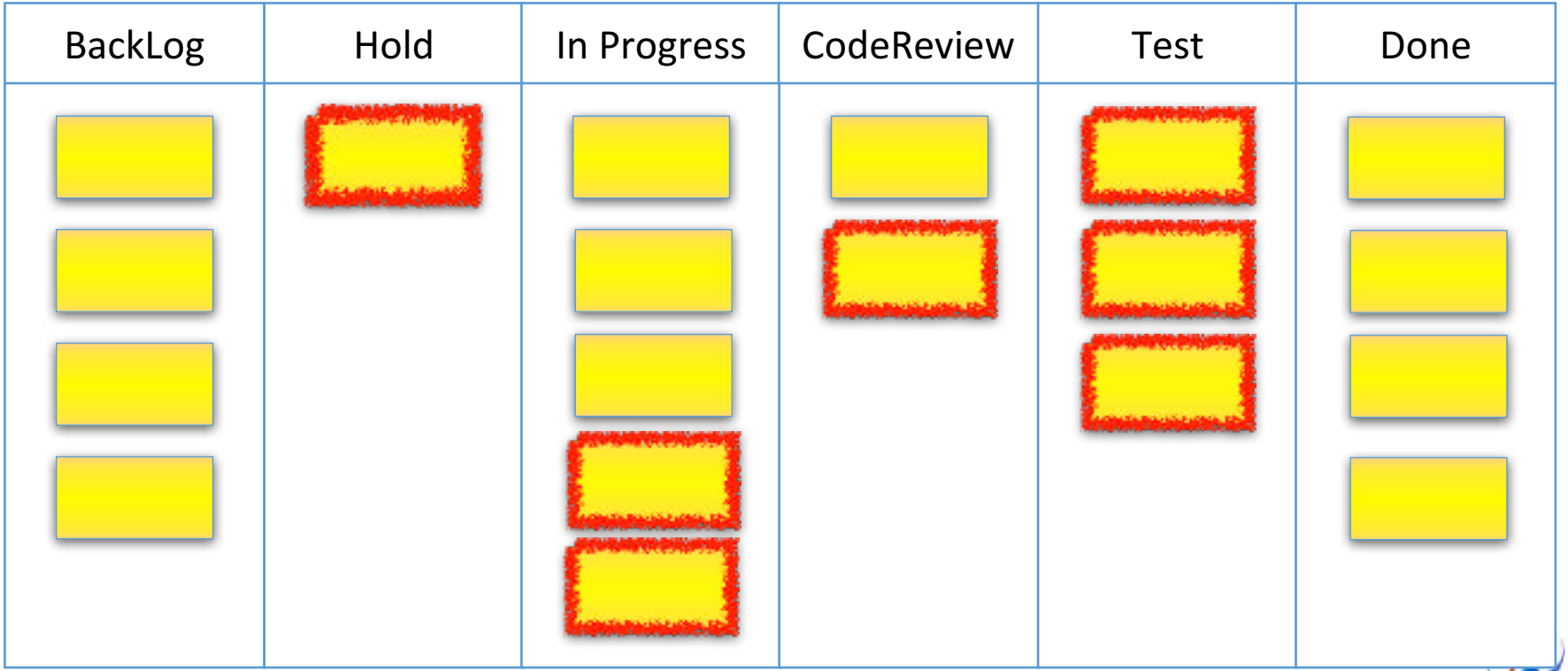
和 Dev 合作

把 Ops 当做用户

可验证

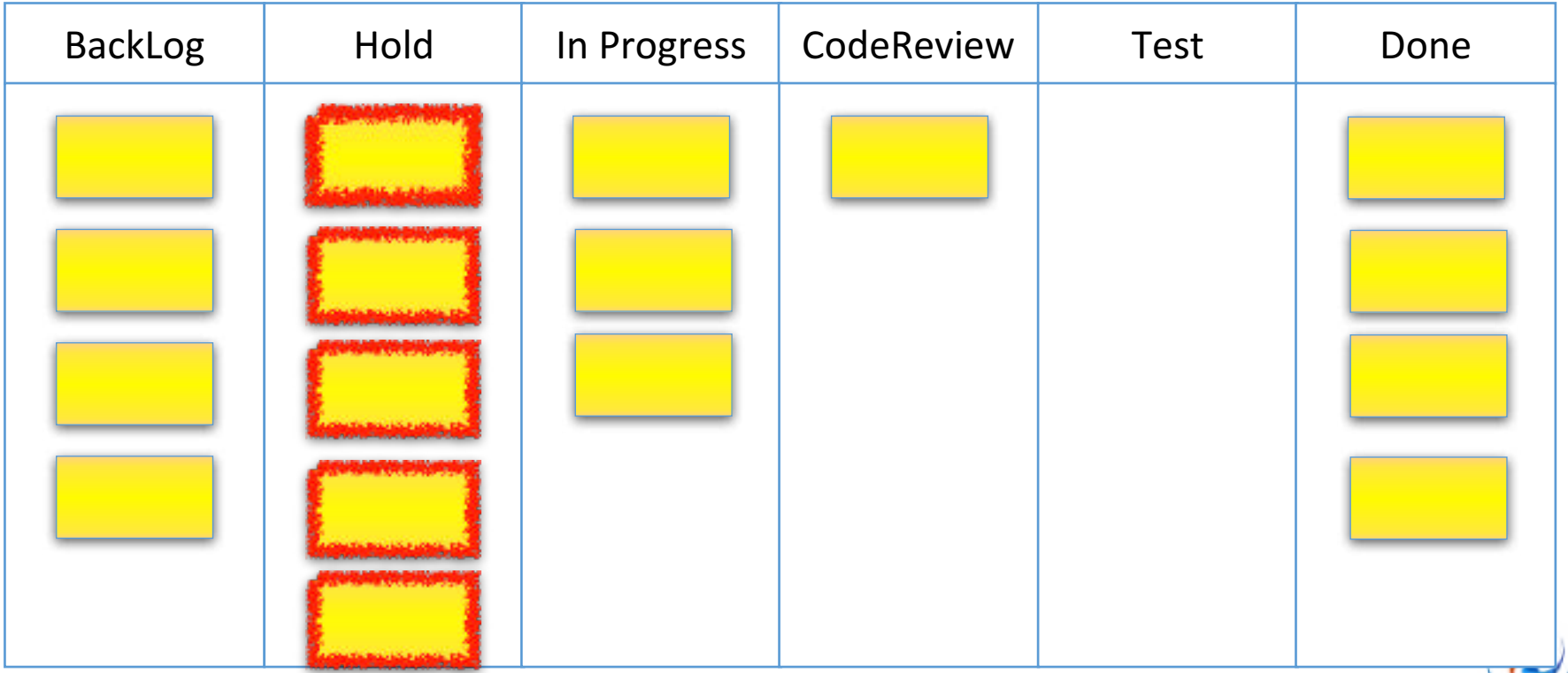
可自动化 *

案例二：看不见流程约束点



案例二：看不见阻塞点

案例二：看不见流程约束点



案例二：看不见流程约束点

第一时间汇报阻塞

第一时间处理阻塞

总结阻塞的规律并改进

案例二：看不见的DevOps的改进效果

Story: #213 As I User, when I click the login

3

Desc:
As a user, when I click the login button, I can

ACs:
1. User can …
2. User should …
3. User should not …
4. Ops could deploy …
5. Ops could monitor …

Time Consume:
1 days in BackLog
0 days in Hold
2 days in Progress
0 days in CodeReview
1 days in Test

案例三：看不见的DevOps的工作效果

改进前：

1. 每月发布：1
2. 环节时间：
    1. Backlog: 3
    2. In Dev: 7+
    3. In QA:3
3. 阻塞时间：8
4. 返工次数：4

六个月后：

1. 每月发布：12
2. 环节时间
    1. Backlog: 3
    2. In Dev: 4~6
    3. In QA: 2
3. 阻塞时间：2~3
4. 返工次数：3

流程改进的目标

看见 DevOps 的工作验证条件 / 效果

看见 DevOps 的工作量越来越少

减少“待办”库存

单一方向工作流

不断反思和学习和回顾

衡量 DevOps 的性能指标

Lead time for changes(变更前置期)

Release frequency(发布频率)

Time to restore service(故障恢复时长)

Change fail rate(变更失败率)

衡量 DevOps 的性能因素TOP5

Peer-reviewed change approval process(结对变更审批过程)

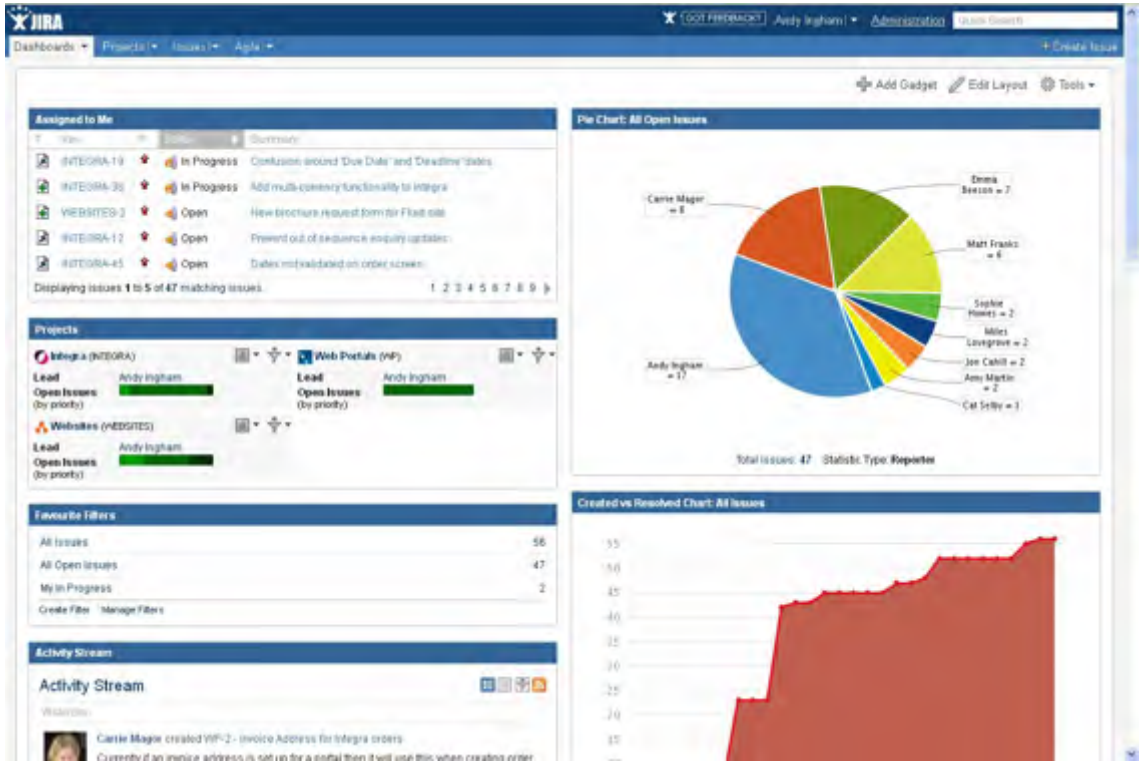
Version control everything(版本控制一切)

Proactive monitoring(主动式监控)

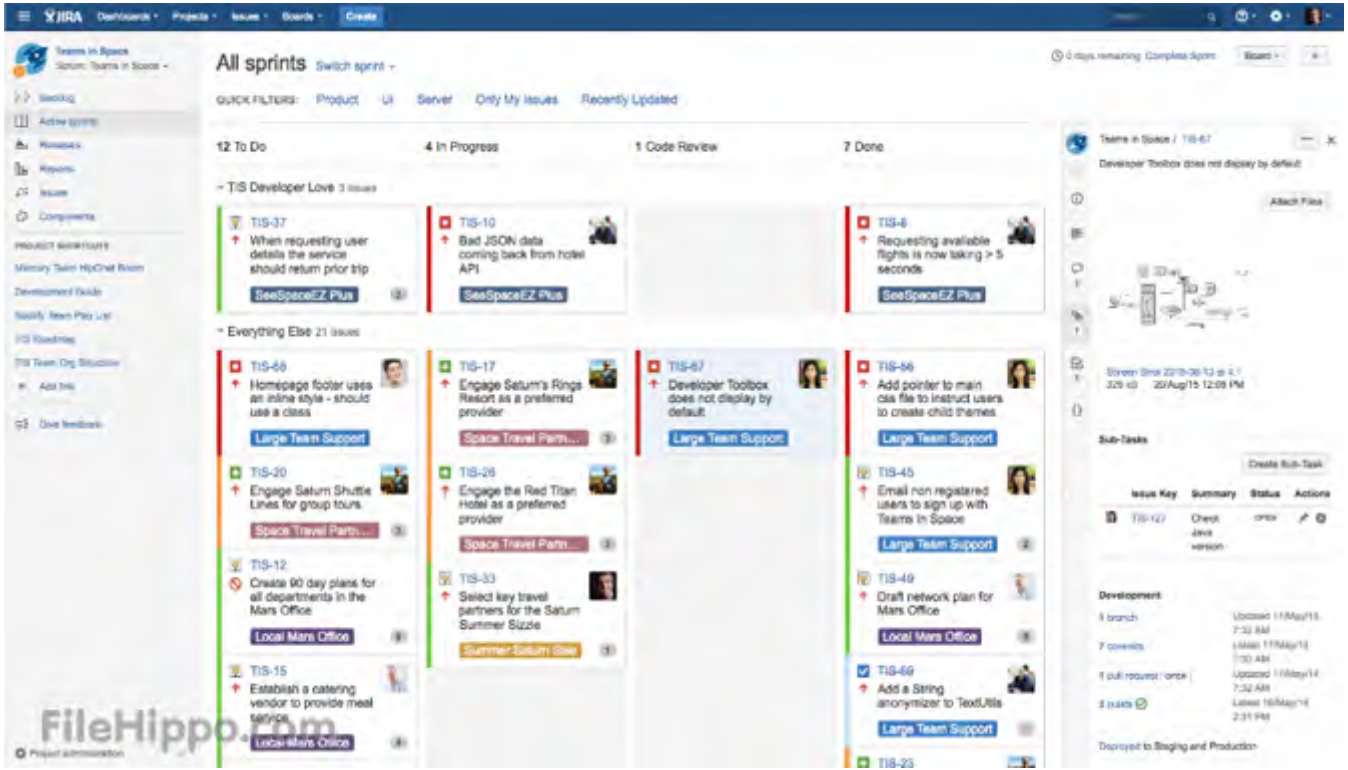
High-trust organizational culture(高度信任的组织文化)

A win-win relationship between dev and ops(Dev和Ops之间的双赢关系)

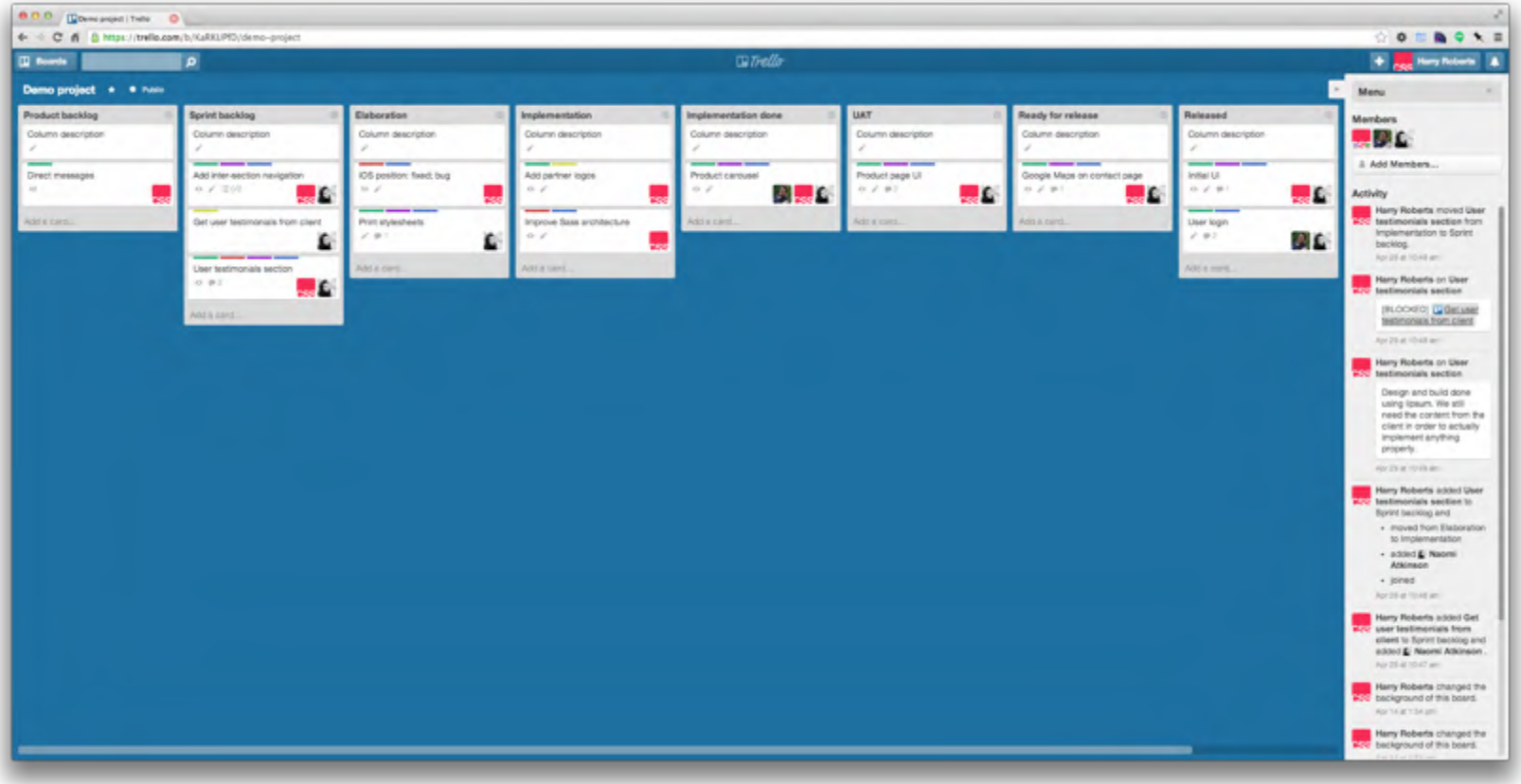
可视化工具 —— Jira


JIRA
Dashboards
Projects
Issues
Agile
Andy Ingham
Administration
Add Gadget
Edit Layout
Tools
Assigned to Me
In Progress
In Progress
Open
Open
Open
Displaying issues 1 to 5 of 47 matching issues
Pie Chart: All Open Issues
Carrie Magor
Emma
Matt Franks
Andy Ingham
Total Issues: 47
Statistic Type: Reporter
Projects
Integra (INTEGRA)
Web Portals (WP)
Websites (WEBSITES)
Lead
Andy Ingham
Open Issues
(by priority)
Favourite Filters
All Issues 56
All Open Issues 47
My In Progress 2
Create Filter
Manage Filters
Created vs Resolved Chart: All Issues
Activity Stream
Activity Stream
Carrie Magor created WP-2 - Invoice Address for Integra orders

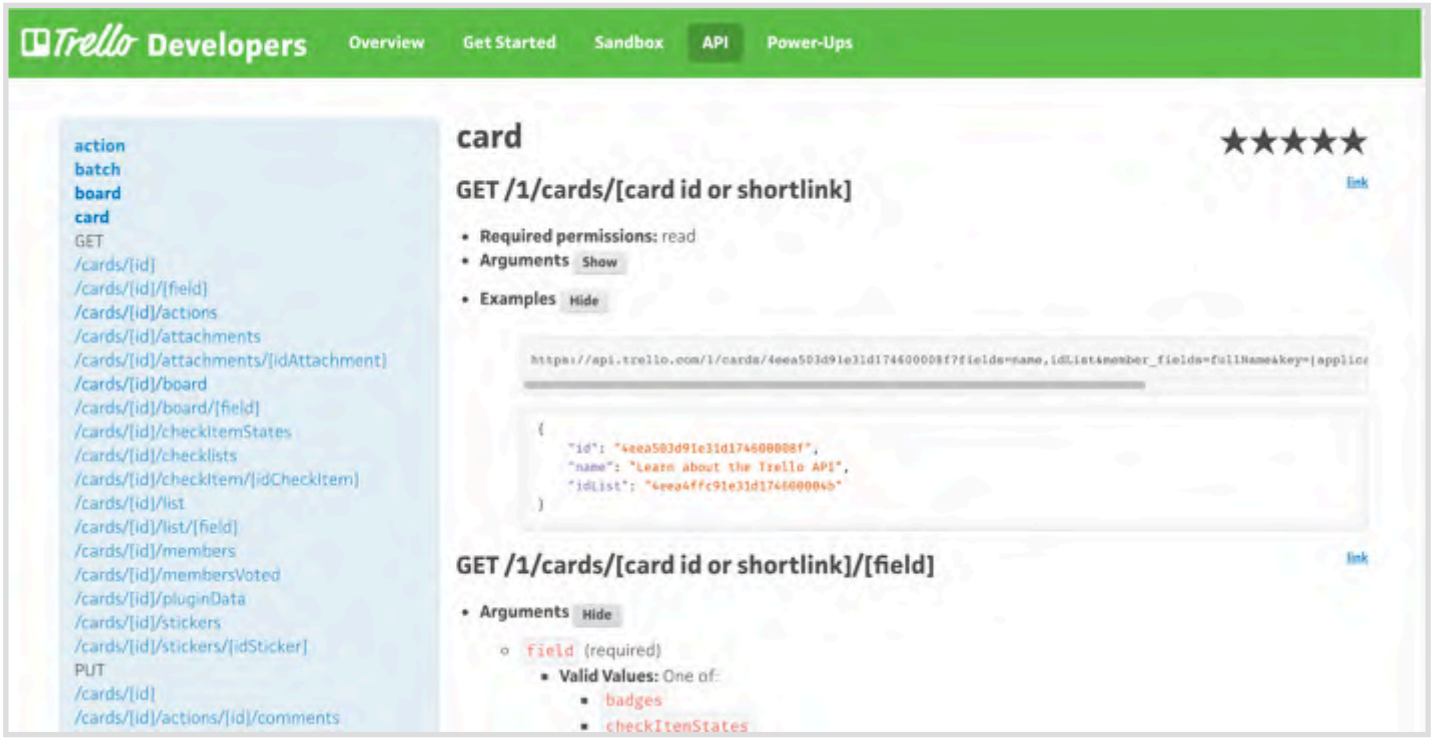
可视化工具 —— Jira


All sprints
12 To Do
4 In Progress
1 Code Review
7 Done
TIS Developer Love
Everything Else
FileHippo.com

可视化工具 —— Trello

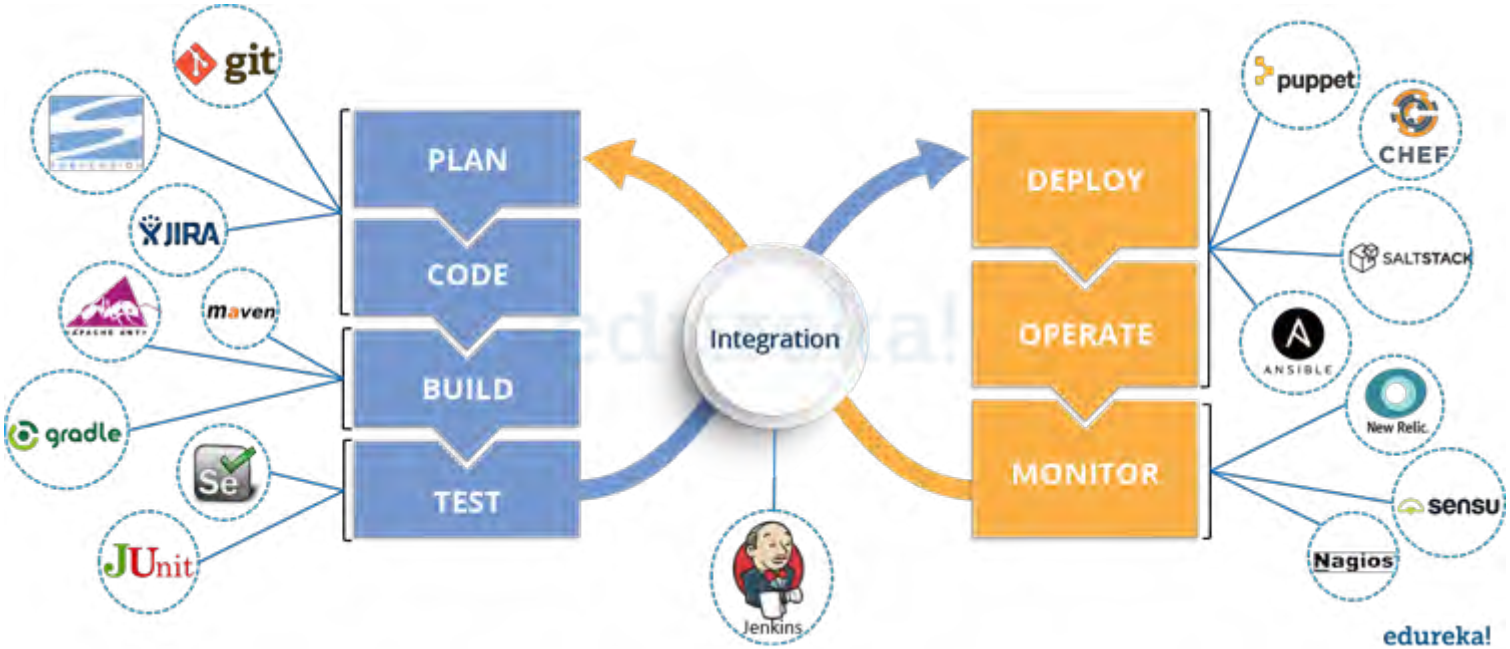


可视化工具 —— Trello



目录

“看见”DevOps 的技术上下文



“看见”DevOps 的技术上下文

案例一：有 Ops 辞职了……

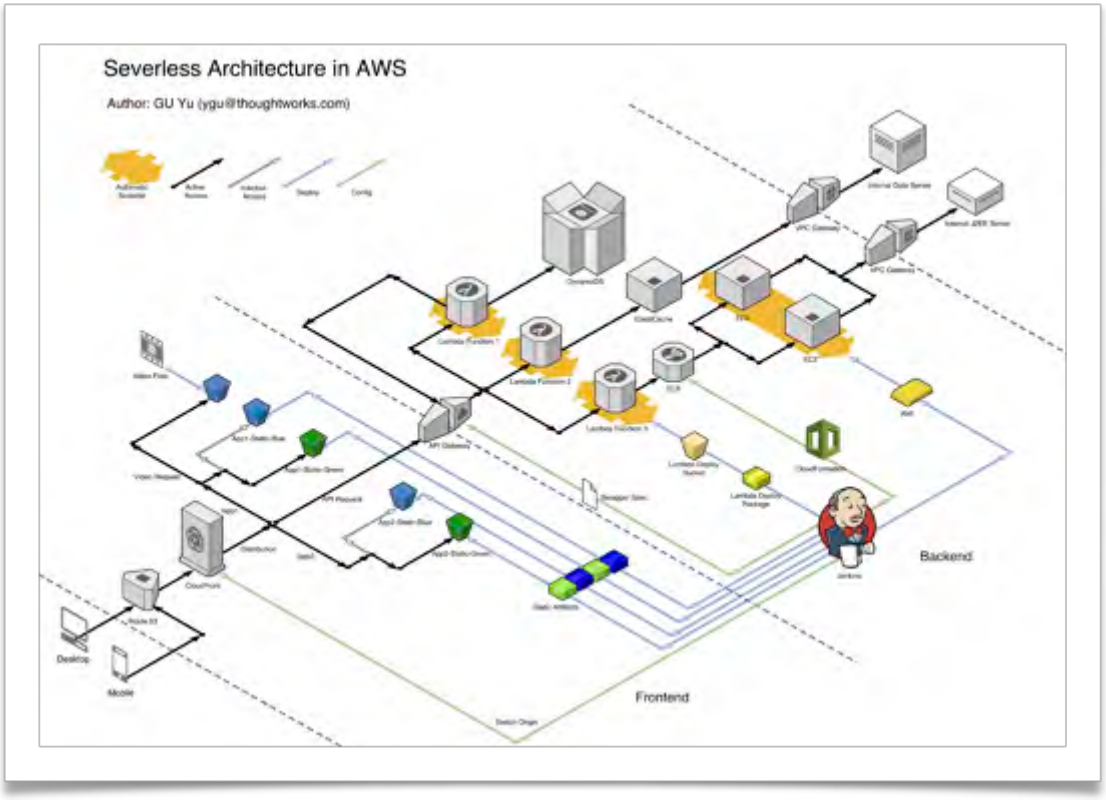
交接从来都是有交没有接

都要从头学起

万事开头难，然后中间难
最后结尾难。

没有架构图就像没有地图在徒步

案例一：看见 产品 的架构


Severless Architecture in AWS
Author: GU Yu (ygu@thoughtworks.com)
Backend
Frontend
Desktop
Mobile

案例一：看不见的系统的架构

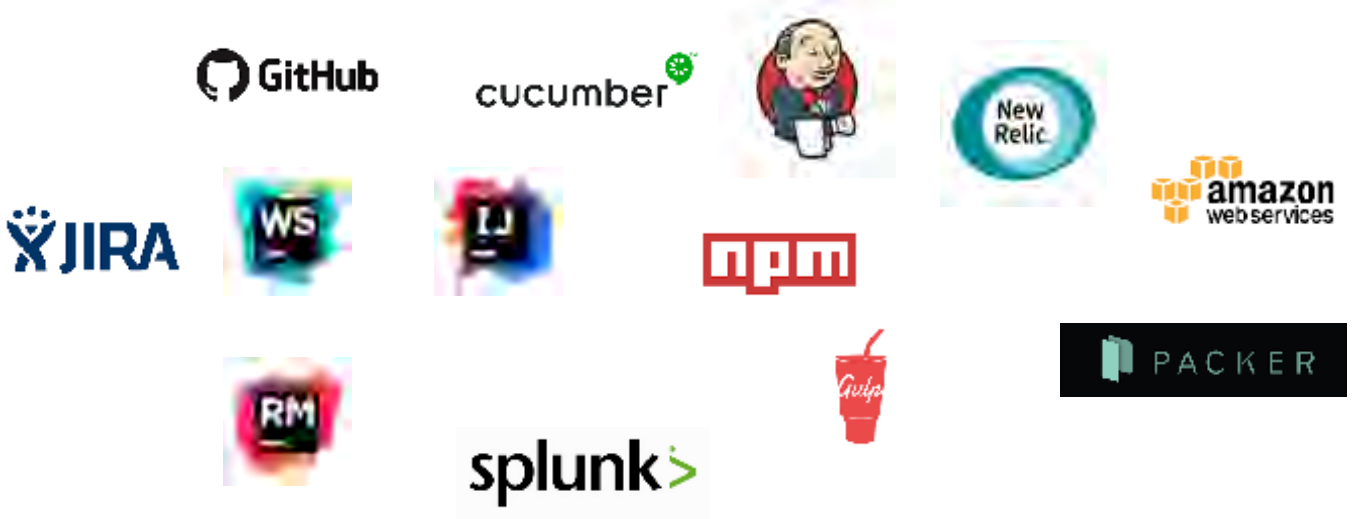
系统架构图给了所有人一个对产品技术点的上下文，共担责任

所有人都可以根据其进行学习和改进

从 Unknown Unknown 到 Know Unknown

案例二：看不见的DevOps技能

DevOps 需要哪些技能？根据所使用的工具的熟练度来定义：



案例二：看不见的DevOps技能

DevOps 需要哪些技能？根据所使用的工具的熟练度来定义：

0 - 没听说过
1 - 用过
2 - 可以讲出原理，最佳实践。
3 - 可以训练别人

案例二：看不见的DevOps技能

Karl	Vinny	Gary	Ajon	Zheng
0	0	1	1	3
1	1	2	3	1
2	3	2	3	1

案例二：看不见的DevOps技能

从团队的角度出发，而不是个体

DevOps 团队是自治的，鼓励每个人去学习，提升团队的短板

“Truck Test” —— 每迭代交接

客观评价 and 主观评价

案例三：看不见的DevOps技术债

现有版本

过时/有 Bug 的插件

新的技术栈

新版本

流水线即代码

Docker 支持

安全性稳定性的更新

案例三：看不见的DevOps技术债

Facts：
1. 基础设施不是一成不变的，基础设施软件也要更新。
2. 技术债会产生利息。
3. 随时都要明白自己的技术债。

常见原因：
1. 没预算
2. 没时间
3. 没人

案例三：看不见的DevOps技术债

饭要一口一口的吃，路要一步一步的走.....

技术债清理策略：
1. 建立技术债务清单
2. 评估技术债风险/收益
3. 建立技术债务优先级
4. 清理技术债：并行——替换

案例四：构建DevOps的技术雷达

ThoughtWorks 技术雷达

持续交付

团队技能矩阵

案例四：构建DevOps的技术雷达

第一步，勾画出端到端活动

案例四：构建DevOps的技术雷达

第二步，填入所用工具

Plan	Code	Test	Integration	Build	Deploy	Monitor
JIRA	GitHub	cucumber		PACKER	amazon web services	New Relic
		Rspec			AWS sdk	splunk>
			npm			
			gulp			

案例四：构建DevOps的技术雷达

第三步，根据工具找到活动最佳实践

Story	TDD	End-2-End	Pipeline	Versioning	BG deploy	
Plan	Code	Test	Integration	Build	Deploy	Monitor

Rspec

AWS sdk

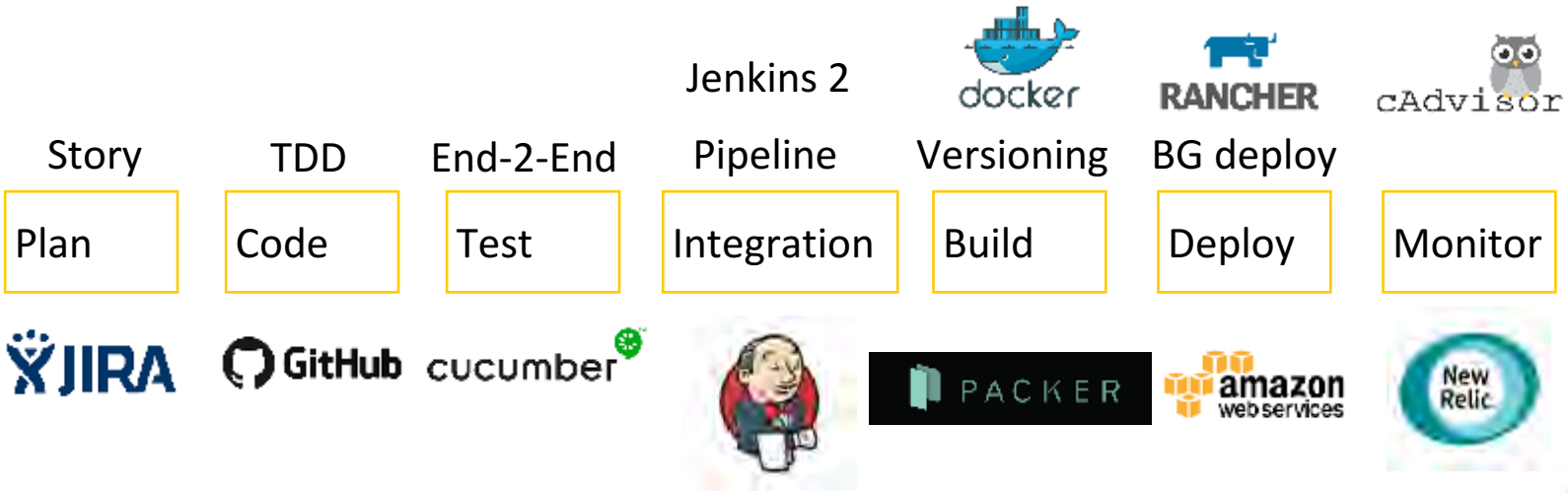
案例四：构建DevOps的技术雷达

第四步，根据工具的最佳实践制定熟练度

0 - 没听说过
1 - 用过
2 - 可以讲出原理，最佳实践。
3 - 可以训练别人

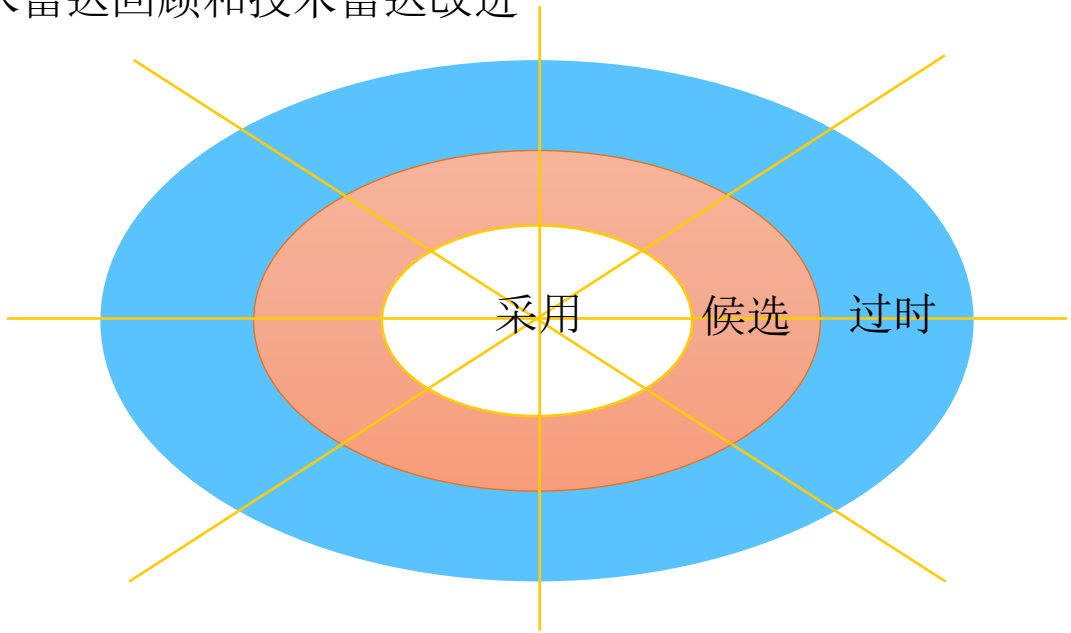
案例四：构建DevOps的技术雷达

第五步，根据最佳实践找到候选工具



案例四：构建DevOps的技术雷达

第六步，技术雷达回顾和技术雷达改进



目录

看见DevOps的全景

DevOps 的组织上下文 —— 确立 DevOps 的改进目的和范围

DevOps 的工作流上下文 —— 度量 DevOps 效果，构建 DevOps 团队和文化

DevOps 的技术上下文 —— 提升团队 DevOps 技能

高效运维社区
GreatOPS Community

会议

- 3月18日 DevOpsDays 北京
- 8月18日 DevOpsDays 上海
- 全年 DevOps China 巡回沙龙
- 4月21日 GOPS深圳
- 11月17日 DevOps金融上海

培训

- EXIN DevOps Master 认证培训
- DevOps 企业内训
- DevOps 公开课
- 互联网运维培训

咨询

- 企业DevOps 实践咨询
- 企业运维咨询

商务经理：刘静女士
电话 / 微信：13021082989
邮箱：liujing@greatops.com

Thanks

高效运维社区

开放运维联盟

荣誉出品

想第一时间看到
高效运维社区公众号
的好文章吗？

请打开高效运维社区公众号，点击右上角小人，如右侧所示设置就好