HUAWEI

HUAWEI IT大咖说 知识共享平台
bbs.huaweicloud.com

华为云技术私享会

HUAWEI

HUAWEI IT大咖说 知识共享平台
bbs.huaweicloud.com

华为云
技术
私享会

DevOps-软件研发效能提升之道

华为高级产品经理
翟然

自我介绍

教育背景

- 北京邮电大学，英国伦敦玛丽女王学院，电信工程及管理专业本科
- 美国南加州大学，计算机科学硕士

从业经历

- 美国工作经历：EA游戏项目，Disney 手游项目，3D云建模和大数据可视化云平台项目
- 国内工作经历：AR/VR 教育+人工智能项目

现阶段

- 华为：华为云DevCloud高级产品经理
- 关注领域：游戏开发，架构设计和产项目管理，AR/VR，Devops持续交付，云计算等

扫一扫加微信：

微信号：zhairan2012

翟然（Michael）
zhairan1@huawei.com

华为云技术私享会

Agile Game Development & DevOps

敏捷开发不只是一种软件开发方法，更是一套关注价值的行为做事哲学

游戏开发简史：看一款游戏从几个人月到百人团队历时数年

游戏开发危机四伏：

- **方法欠妥，丧失应有的乐趣：游戏行业日渐成熟，游戏开发复杂性和不确定性强，导致最后出品的游戏华而不实**
- **有志之士，满腔热情，希望给用户带来充满乐趣的游戏体验：在项目中暗无天日的加班加点的煎熬，带着多年经验黯然离开**
- **敏捷开发：确保游戏开发商业可行性，确保游戏创作本来的趣味性**

早期方法论：
软件部分和美术部分已经成为商业游戏最大成本，3-4人月激增至30-40人月：

- 瀑布式开发，经常迭代回顾
- 一将功成万骨枯
- 全球游戏市场持续稳定增长
- 游戏研发成本不断提高，对冲

三大危机：

- 创新乏力：避免失败而不去冒险，炒冷饭，蹭IP
- 价值缩水
- 工作环境恶化：管理方法创新

电子工程师，固定电路游戏机 – 网球，太空大战

1948- 1972

1970s

街机时代：快速迭代，雅达利Atari快速调整（投入极高，小心验证品质）

1996

MMORPG：网络创世纪

2000

索尼发售PS2

2004

Facebook：社交化页游

2006

索尼发售PS3
任天堂发售Wii

2007

iPhone 出世

2009

电子竞技：LOL

2011

微信

2012

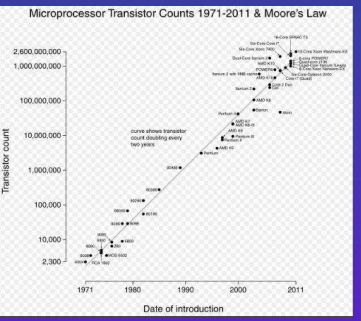
4G：移动互联网

AR\VR：Pokémon Go
AI：Alpha Go

2016

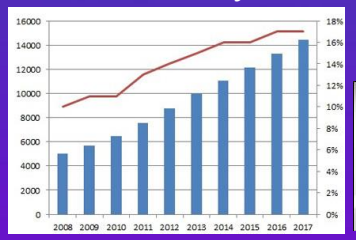
2018

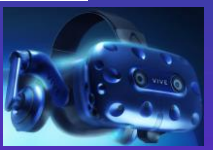
区块链



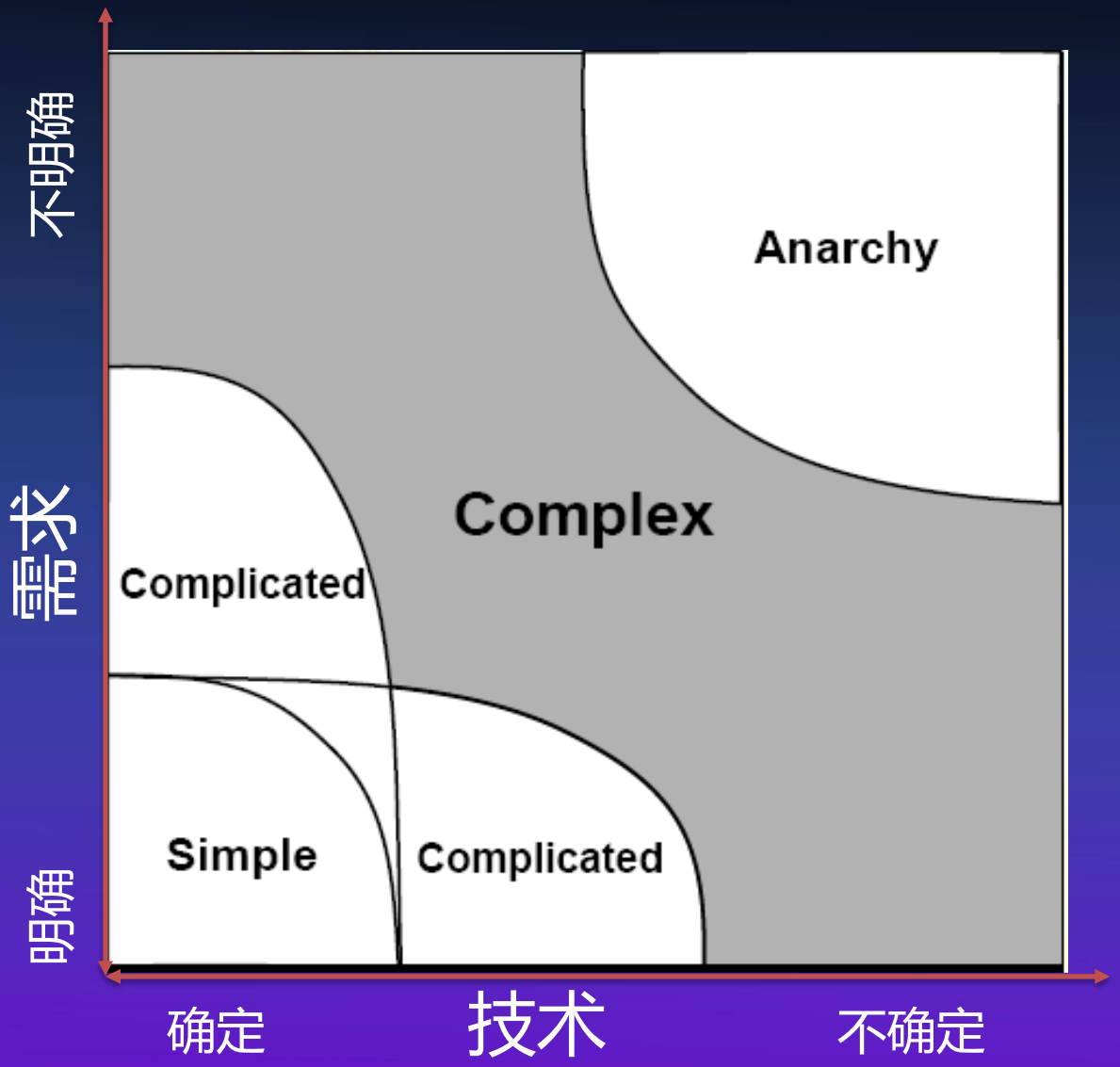
摩尔定律：
每隔几年，平台性能翻倍，芯片运算能力，GPU，内存更强：3D表现，AR、VR，AI...

Winston Royce





游戏开发面临的挑战



常见问题四大问题：

- 不同分工人员各自为阵
- 构建版本总是有问题
- 估计与进度表总是过于乐观
- 管理层总是忙于“救火”，没时间思考宏观战略

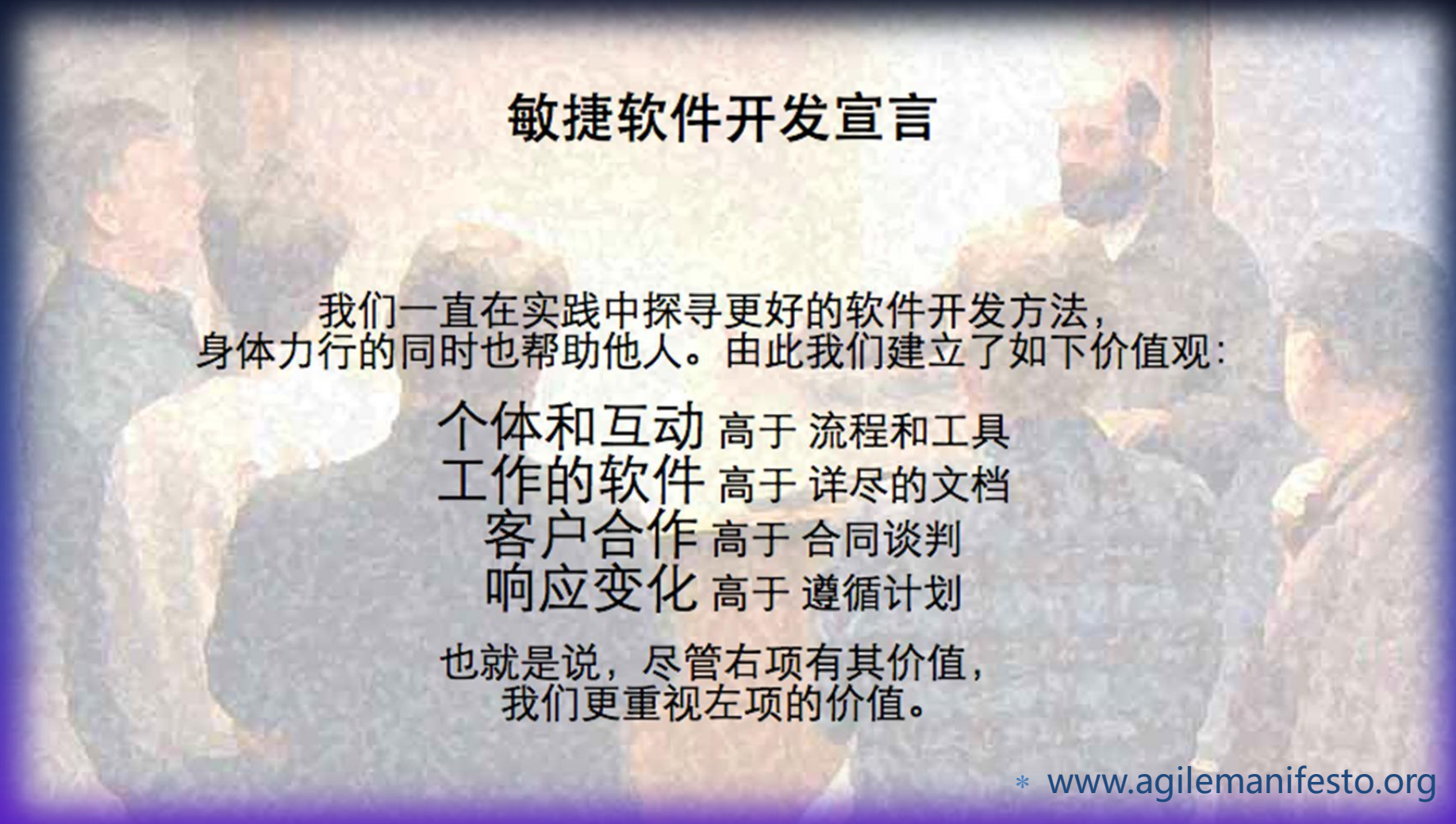
挑战

- 新增特性
- 盲目乐观的计划
- 产品制作过程中的挑战
- 成本与质量
- BDUFs (Big Design Up-Front)

解决思路？

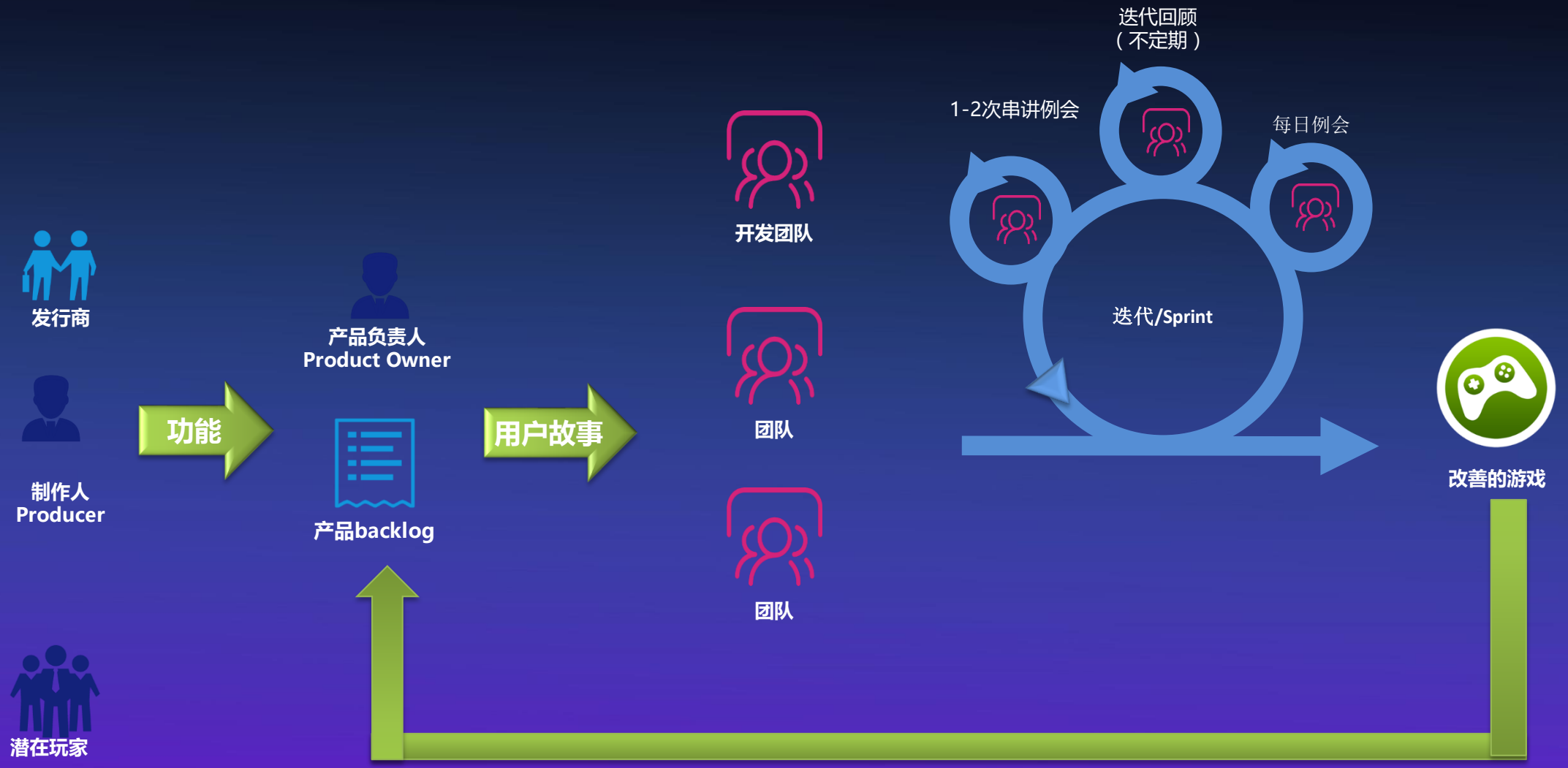
- 先明确核心乐趣
- 将敏捷价值观应用与游戏开发

敏捷宣言 Agile Manifesto

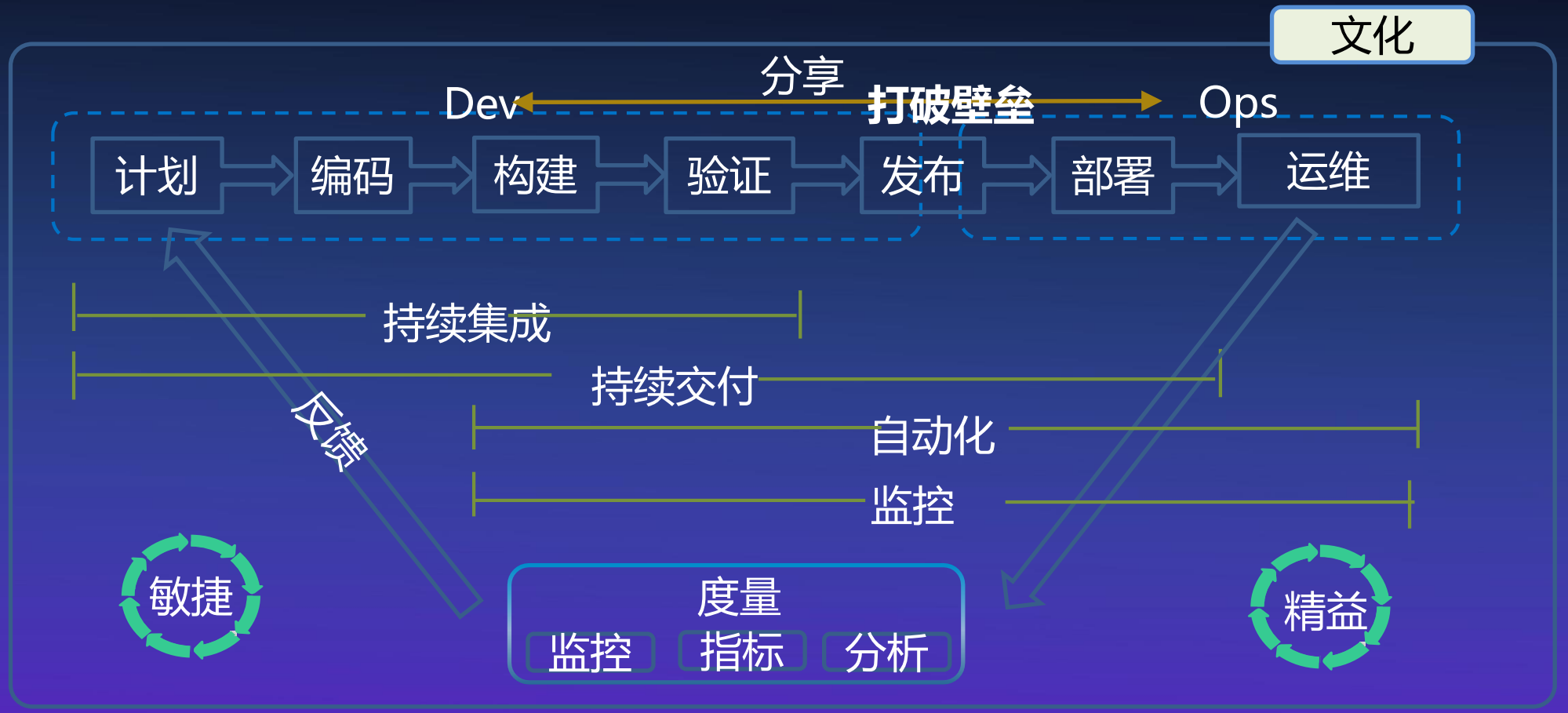


* www.agilemanifesto.org

基于Scrum的项目管理



敏捷：不是标准的标准，DevOps：是一套方法也是一种文化



Meet DevOps

计算机科学中的任何问题都可以通过创造一个概念来解决。如果一个不行，那就两个！

软件危机及软件工程诞生于1968年

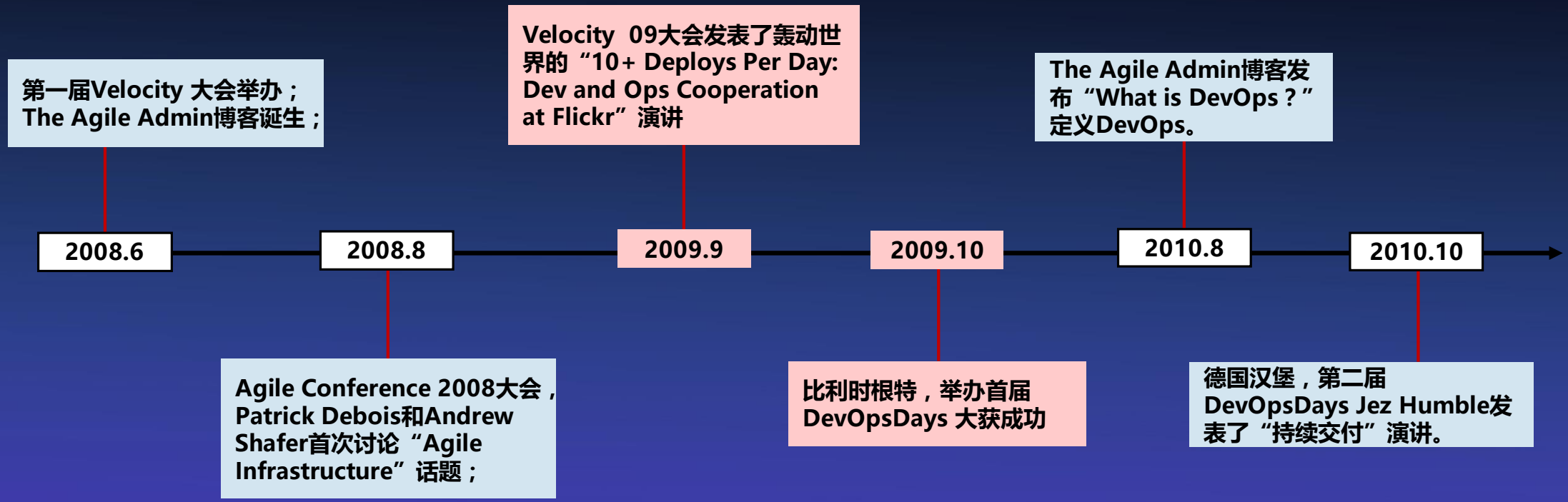
1960s中期，随着软件复杂性上升，软件开发问题大量爆发。1968年，北大西洋公约组织（NATO）在联邦德国的国际学术会议创造”**软件危机**”（**Software crisis**）一词并于同年召开会议提出“软件工程”(software engineering)概念以期解决软件危机中遇到问题。

“软件危机”中遇到的主要问题：

- 对软件开发成本和进度的估计常常很不准确，交付经常出现延期；
- 经常出现用户对“已完成的”软件产品不满意的情况；
- 软件产品的质量往往达不到要求；
- 软件通常是很难维护的；
- 软件产品往往没有适当的文档资料；
- 软件开发成本难以管理，经常远远超出预算；
- 软件开发生产率提高的速度远远不能满足社会对软件产品的日益增长的需求。

美国IBM公司于1963年～1966年开发的System/360系列机的操作系统，其开发过程被视为计算机发展史上最大的一次豪赌。。 IBM决定征召6万多名新员工，创建5座新工厂，写出近100万行的源程序。尽管投入了这么多的人力和物力，得到的结果却极其糟糕，交付时间时间不断的顺延，远远超出预算。

DevOps起源



DevOps 1.0：
强调解决开发和运维团队间的协同问题，将敏捷思想运用于运维领域。

DevOps 2.0：
强调解决产品、开发、质量、运营和安全等基于端到端价值流驱动的快速交付问题

什么是DevOps：一场运动，一种文化，一个软件交付方式

DevOps的概念:

维基百科

DevOps，**Development and Operations**，是一组过程、方法与系统的统称，用于促进软件开发、运维和质量保障部门之间的沟通、协作与整合。它的出现是由于软件行业日益清晰地认识到：为了按时交付软件产品和服务，开发和运维工作必须紧密合作。
DevOps可看作开发、运维和质量保障（QA）三者的交集。

维基百科，http://zh.wikipedia.org/wiki/DevOps

Gartner

DevOps运动源自于提高IT服务交付敏捷性的需要，早期出现在许多大型公有云服务提供商中，并被其认可。支撑**DevOps的理念基础是敏捷宣言**，它强调人（和文化），致力于改善开发和运维团队之间的协作。从生命周期的角度来看，DevOps的实施者也试图更好地利用技术，尤其是自动化工具，来支撑越来越多的可编程的动态的基础设施。

Gartner，http://www.gartner.com/it-glossary/devops

DevOps专家的观点:

DevOps**打破开发与运维之间的孤岛**，积极协作。广义上指整个组织级的协作优化，包括IT、HR、财务、和公司供应商。约束理论告诉我们，需要优化整体而非单个“孤岛”。整体是与客户问题相关的业务，从精益角度来说，是整个价值链。

Patrick Debois ，DevOps名词的发明和运动的推动者，是DevOps Days创办人之一。

DevOps通常指的是新兴的专业化运动，提倡开发和IT运维之间的高度协同，从而在完成高频率部署的同时，提高生产环境的可靠性、稳定性、弹性和安全性”。DevOps聚焦实现更快的上市时间，减少IT浪费，提高组织效率。

Gene Kim，DevOps领导者，《The Phoenix Project: A Novel About IT, DevOps》、《The Visible Ops Handbook》作者

DevOps与“如何”实现持续交付密切相关。如果没有开发和运维团队之间积极有效的合作文化，持续交付将变得非常困难，甚至不可能。你没有办法越过部门墙，去频繁实施增量式的变更。

Jez humble，Chef的Engineering VP，原Thoughtworks顾问，《Continuous Delivery》，《Lean Enterprise》作者。

DevOps行业公司观点:

亚马逊：
•DevOps侧重于改善协作，沟通，整合软件开发与IT运维。是一个总称，用来描述一种**理念、文化变革和模式转变**。
•DevOps帮助亚马逊**从软件开发敏捷走向业务和运维的敏捷**。
•开发阶段，DevOps重点放在代码构建，代码覆盖，单元测试，打包和部署；运维阶段，在基础设施上聚焦环境分配，配置，编排和部署。

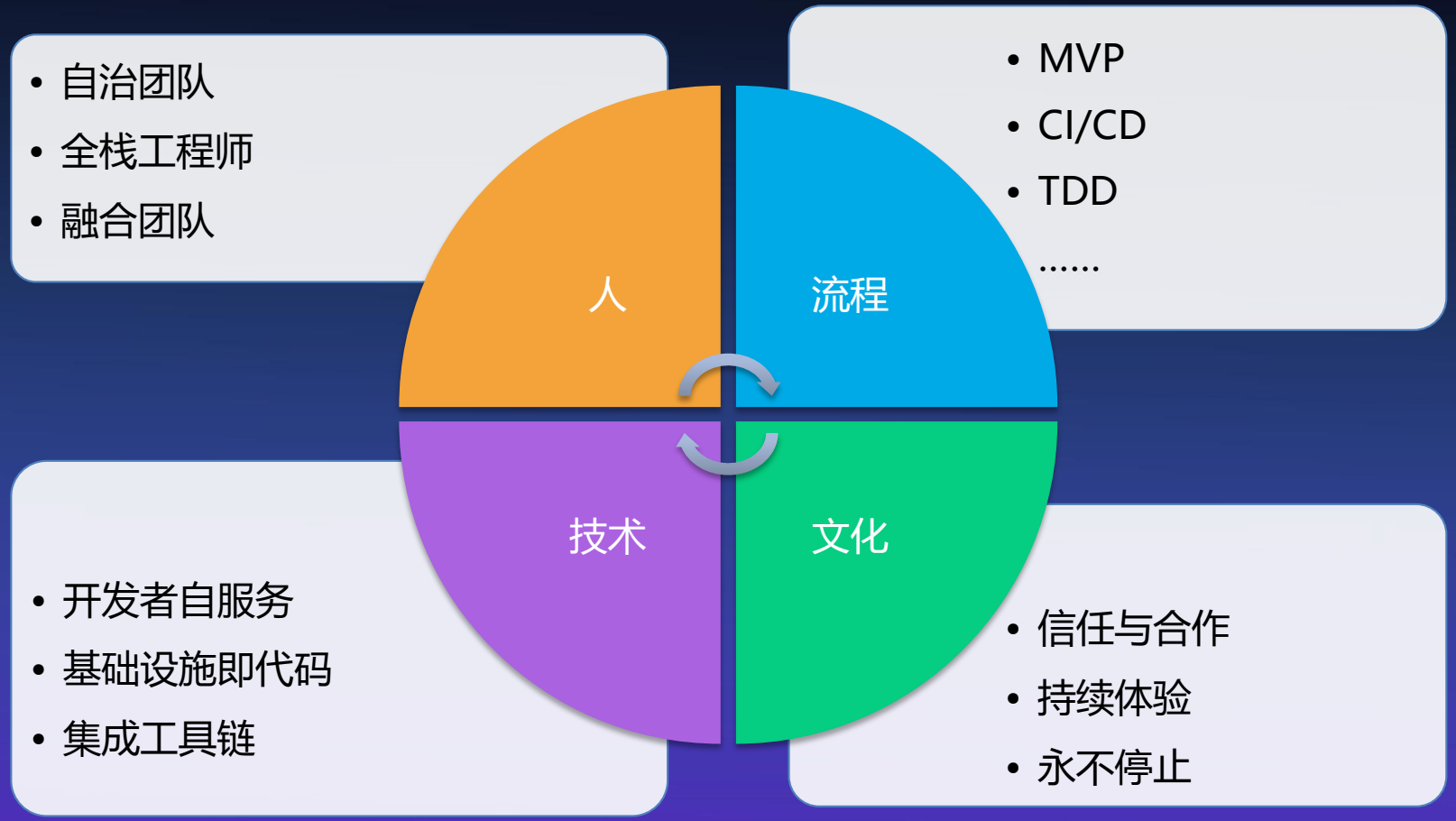
微软：
•DevOps不仅是一项技术或工具集，也**是观念和文化的转变**。它结合人、流程、适当的工具，使应用程序生命周期更快，更可预测。
•我们**应将DevOps视为过程而非目的**，应该选定适合范围的项目，增量式实施，通过这些项目展示成功，学习，并演进。

IBM：
•DevOps是一种提倡将开发机构的文化、流程和工具整合到一起的集成**软件交付方式**，跨越从业务规划、创建、交付到反馈的整个软件开发生命周期，旨在通过持续交付软件，从而帮助企业迅速抓住市场机会，更好地满足客户的需求。
•广义上讲，**DevOps是一种基于精益和敏捷原则的方法**。
•DevOps完善业务向客户、供应商、合作伙伴交付价值的方式。

DevOps的五个要素

文化	建立一体化的全功能团队，打破开发(Dev)与技术运营(Ops)隔阂
自动化	自动化一切可以自动化的
精益	以精益的方式小步快跑，持续改善
度量	建立有效的监控与度量手段快速获得反馈，推动产品和团队的持续改进
分享	不同职能、不同产品之间分享经验

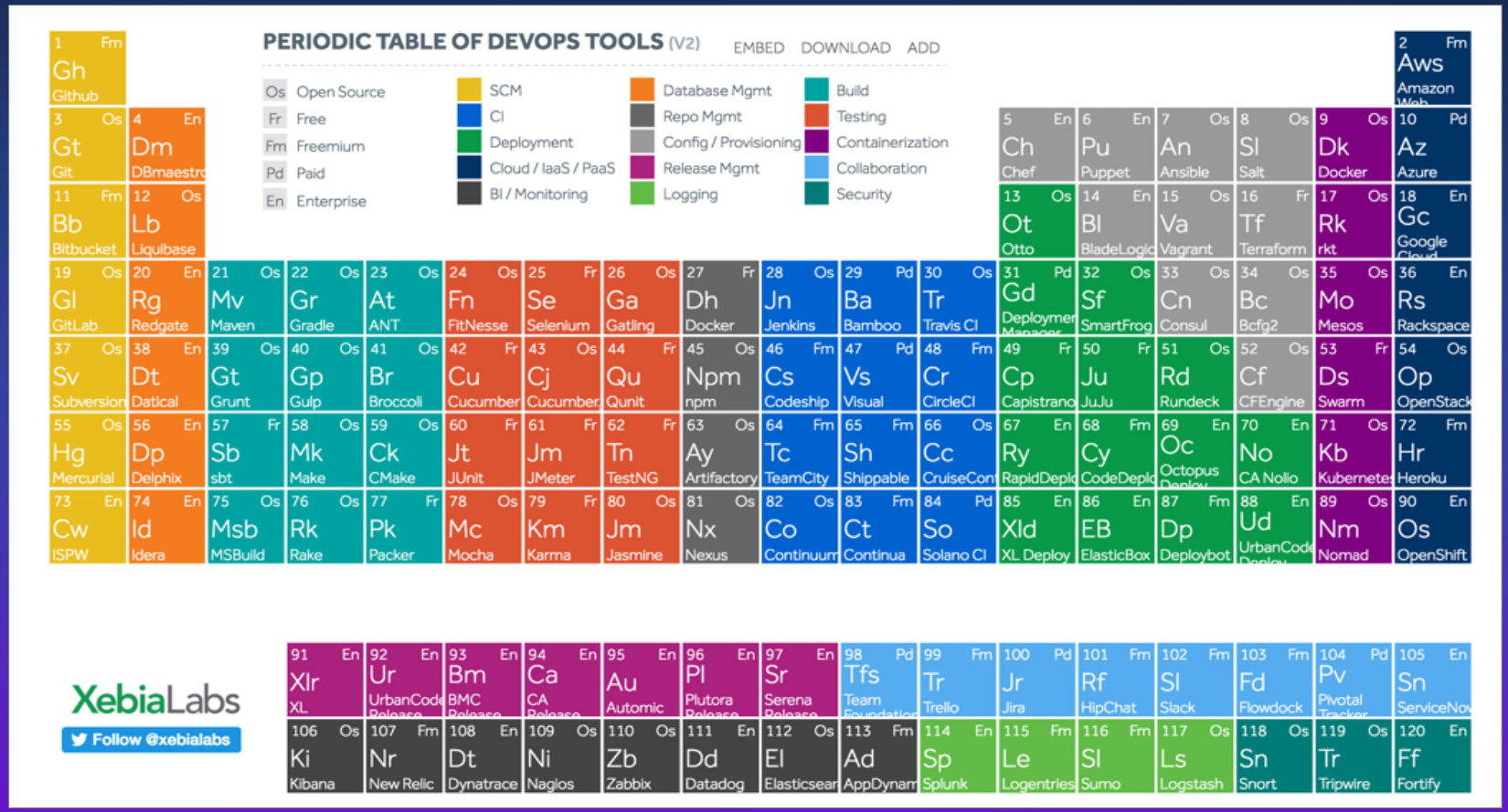
DevOps的关键模式与实践



来源：Gartner DevOps Model

DevOps生命周期过程相关工具

- Project项目管理
- SCM源代码配置管理
- CI持续集成
- Deployment部署
- Repo Mgmt组件库管理
- Build构建
- Testing测试
- Monitoring监控
- Release发布
- Collaboration协作
- Cloud云计算
- ……

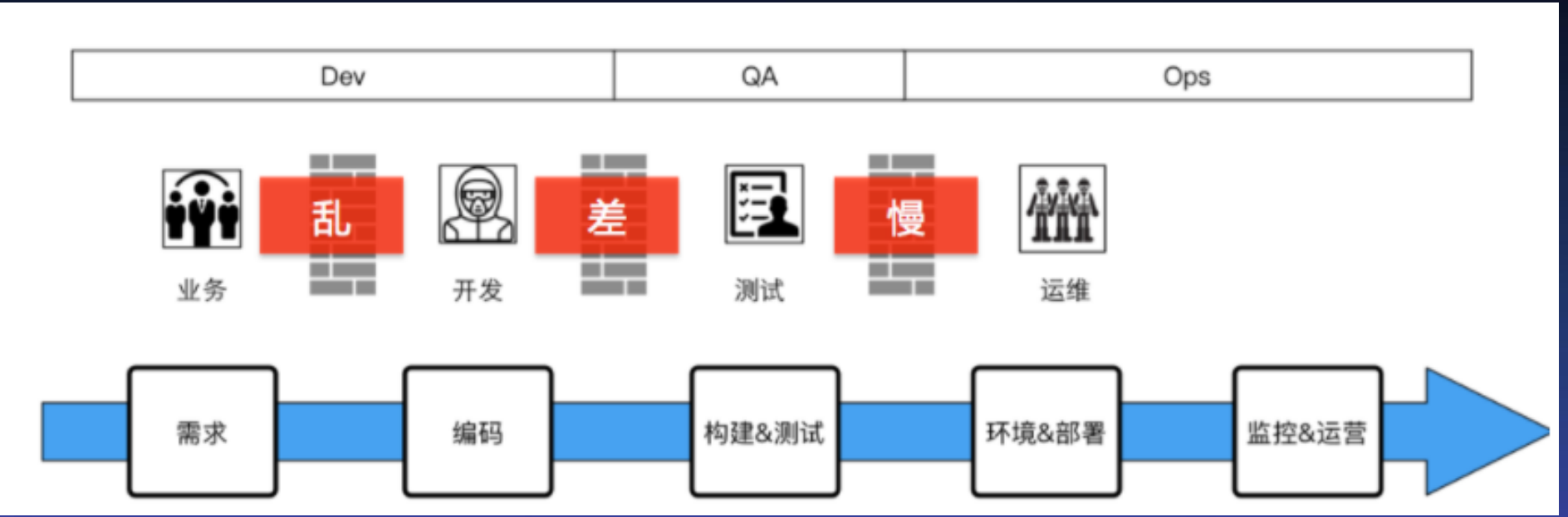


来源：XebiaLabs，15类，120种工具https://xebialabs.com/periodic-table-of-devops-tools

“多快好省”的软件交付挑战

	多	快	好	省
外部挑战	□更多的技术体系 □更多的需求 □更多的用户 □更多的竞争 □更多的创新 ……	□更快的交付 □更快的变化 □更快的响应 □更快的流动 □更快的颠覆 ……	□高价值 □高体验 □高安全 □高性能 □高可靠 ……	□价格低/免费 □优惠多 ……
内部挑战	□更多的技能 □更多的学习 □更多的协作 ……	□快速迭代 □高自动化 □快速恢复 ……	□要有创新 □要有优势 □高可靠 □高性能 □高扩展 ……	□高效率 □低成本 □资源复用 □按需 □MVP ……

我们会经常遇到的问题

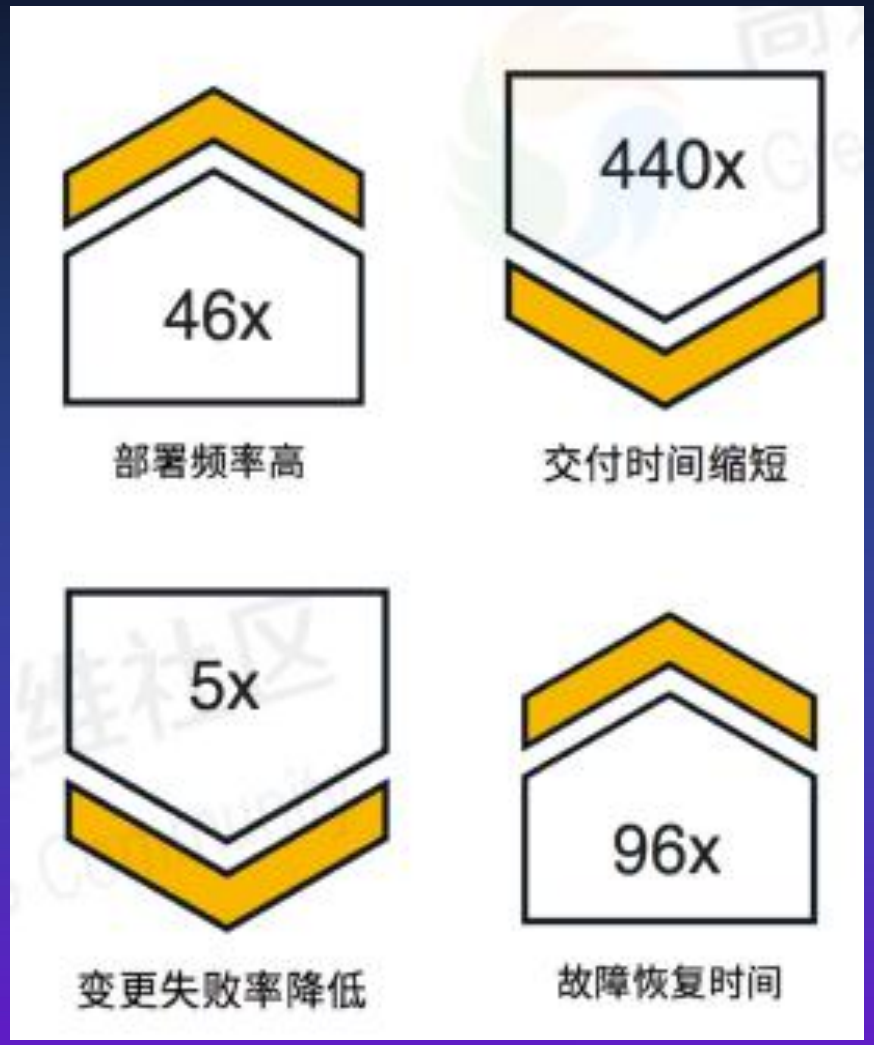


乱：
- 团队怎么划分会比较高效？
- 需求怎么分解？谁负责分解？
- 晨会怎么开？
- 工作量怎么评估才准确？
- 需求变更怎么办？
- 缺陷怎么管理？改不完怎么办？
- 需求描述不清晰怎么办？
- 需求依赖怎么管理？

差：
- 代码检视要不要做？怎么做？
- 主干开发？还是分支开发？
- 转测试质量很差怎么办？
- 持续集成怎么做？
- 单元测试要不要做？
- 版本质量不高，发布后频繁升级
- 项目没有持续改进的途径或机制

慢：
- 手工部署，重复劳动
- 手工测试，自动化测试怎么开展？
- 交付吞吐率太低
- 各交付特性间需要等待一起发布
- 项目总是多多少少延期
- 团队角色间协作效率低，目标不一致
- 改革怎么快速见效，持续进行

DevOps收益



Survey questions	High IT performers	Medium IT performers	Low IT performers
Deployment frequency *For the primary application or service you work on, how often does your organization deploy code?*	On demand (multiple deploys per day)	Between once per week and once per month	Between once per week and once per month*
Lead time for changes *For the primary application or service you work on, what is your lead time for changes (i.e., how long does it take to go from code commit to code successfully running in production)?*	Less than one hour	Between one week and one month	Between one week and one month*
Mean time to recover (MTTR) *For the primary application or service you work on, how long does it generally take to restore service when a service incident occurs (e.g., unplanned outage, service impairment)?*	Less than one hour	Less than one day	Between one day and one week
Change failure rate *For the primary application or service you work on, what percentage of changes results either in degraded service or subsequently requires remediation (e.g., leads to service impairment, service outage, requires a hotfix, rollback, fix forward, patch)?*	0-15%	0-15%	31-45%

	High performers	Medium performers	Low performers
Configuration management	28%	47%[a]	46%[a]
Testing	35%	51%[b]	49%[b]
Deployments	26%	47%	43%
Change approval processes	48%	67%	59%

Do DevOps（I）

实践是检验真理的唯一标准：一个平台DevCloud

软件开发云：一站式云上DevOps研发平台

软件开发云（DevCloud ）是集华为研发实践、前沿研发理念、先进研发工具为一体的研发云平台；软件开发云不是编程工具，是面向开发团队提供的一整套研发工具服务，让软件开发效率高、质量好

Web开发

微服务开发

.....

移动应用开发

Do DevOps（II）

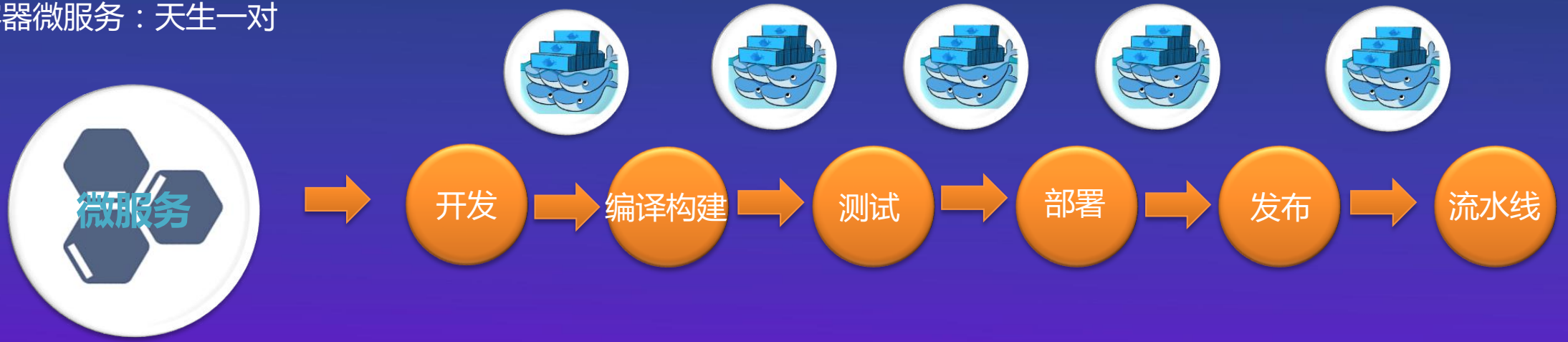
两项核心技术：容器，微服务

DevOps，容器和微服务

容器Devops
- 以应用程序为中心，来理解基础设施
- 降低了复杂度，方便开发和运维的协调
- 提供统一的环境 – 标准化

微服务Devops
- 小 - 独 - 轻 - 松
- 由小型团队创建和管理，并且可以快速更改

容器微服务：天生一对



Be DevOps

让DevOps变成团队的一种状态，而不仅仅是动作

领导力：变革始于“脑袋”

远见
- 传递组织的方向
- 引领团队的方向
- 明确团队3-5年的发展

激发力
- 挑战团队舒适区，提升危机感
- 牵引激发团队不停思考，提问
- 不断挑战团队基本工作方法，牵引改进

及时激励
- 及时认可团队的提升
- 及时认可工作质量的提升
- 亲自表扬个体的出色工作

富有激情
- 提升成员作为团队一员的自豪感和成就感
- 宣讲阶段性成果和好消息，提升团队正能量
- 挖掘机会，激发团队热情、鼓励成员，牵引不断向上的动力。

辅导力
- 换位思考，同理心
- 关注成员诉求、发展和兴趣

图片和建议来自《2017 State of DevOps Report》

变得敏捷，变得DevOps，适合自己的才是最好的

团队在透彻理解DevOps或敏捷理念的基础上，同步不断的反馈优化，选择最适合自己的实践，避免教条化

Q&A 问答环节

THANK YOU