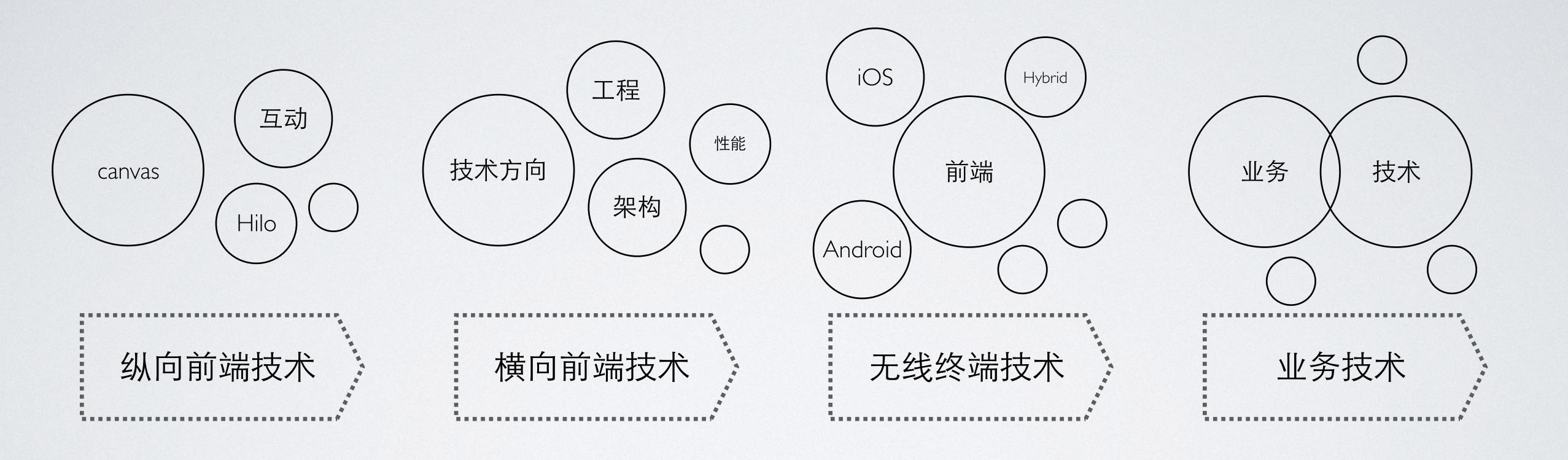


愿风裁尘,以心御匠

前端作为技术人的思考 — 岑安

关于我,一个愿意去思考技术的"伪"无线研发工程师

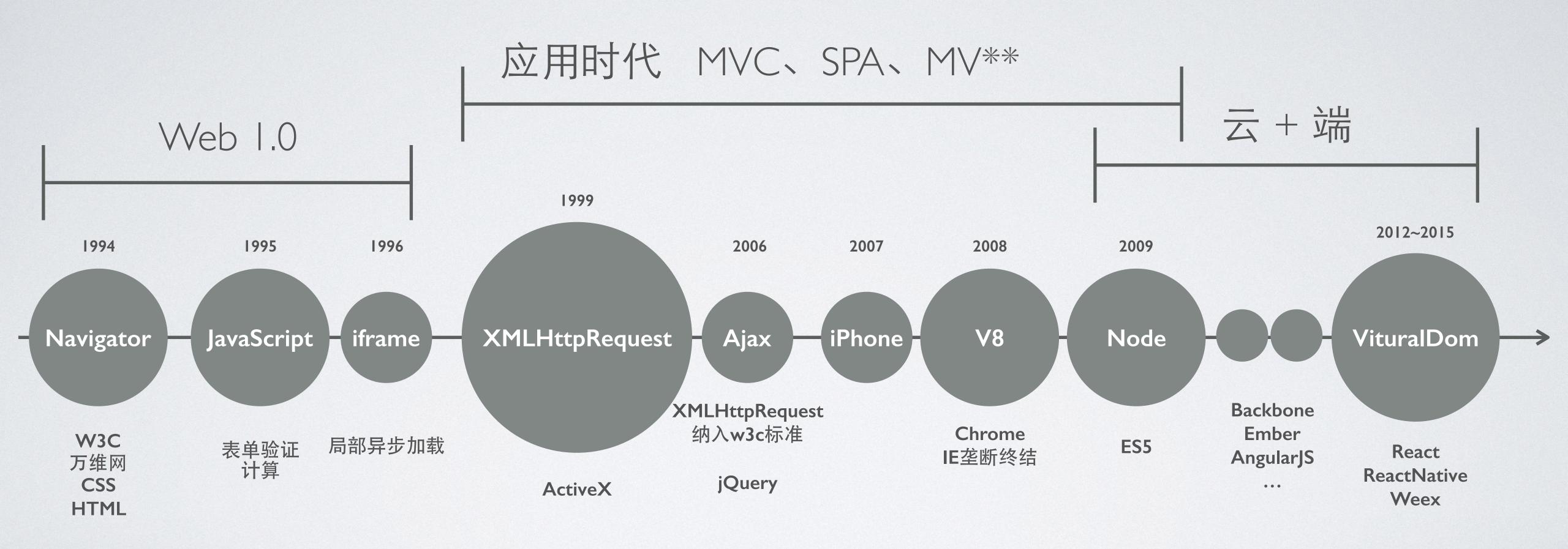


概要

- A. 前端发展历程,回归"技术人"的本质
- B. 技术为业务服务的思考路径"从点到线"
- C. 技术人成长的三个关键词
- D. 前端的核心竞争力的演进

前端发展历程

前端发展历程



阶段困境和解法

后Web 1.0时代

WebPage从Hyperlink的本质 衍生到ContentProvider的能力

内容承载、排版、阅读体验

- I. 基础组件, Tab、Dialog...
- 2. Js功能函数复用 prototype、util
- 3. css布局&排版最佳实践
- 4. xhtml规范,语义化

应用时代 MVC、SPA、MV**

云+端

从WebPage到WebApp级别的应用时代 迫切需要解决的是怎么向传统软件工程靠拢 无线崛起+泛Nodejs时代 怎么回归技术服务业务本身

框架、工程化、富交互&性能

- I. 从JQ到MV*, MV**
- 2. CommonJS规范下的模块化; RequireJS, SeaJS
- 3. 工程化借鉴软件工程能力:
 - 1. 包管理
 - 2. 研发全流程
 - 3. 部署能力
 - 4. 工程服务化
- 4. 应用富交互
 - 1. 互动
 - 2. 可视化

研发模式、云计算、面向五新

- 1. 万物上云之后的中后台能力 + 开发 者服务能力
- 2. 端统一+泛Node带来的研发模式和 组织的重组(Weex, Chair)
- 3. 新零售的线下场景,大LBS, iOT
- 4. 新金融,新制造,新技术,新能源 待大家深入实践和思考

殊途同归 18财年的大前端五大方向

企业中后台

开发者服务

泛Nodejs

端技术

图形

"组件通,协议通"

标准化:

DSL协议,数据接口

智能化:

从设计到代码

. 从端到云一体化应用

解决方案

₹ 2. OS系统深度链接, 闭环生态

: 3. ISV的开放与升级

一条基本线 + 一个突破点 ¦ Weex + Webview

基本线:

Node应用治理

突破点:

:相辅相成

Weex:

性能、iOT能力、国际化 ! 可视化:

Webview:

内核、渲染引擎、新标准上

可视化 + 互动能力

互动:

容器,引擎,框架,平台

智能化,数据联动

"前端"回归技术人的本质: 技术为业务服务

从业务到技术的思考路径

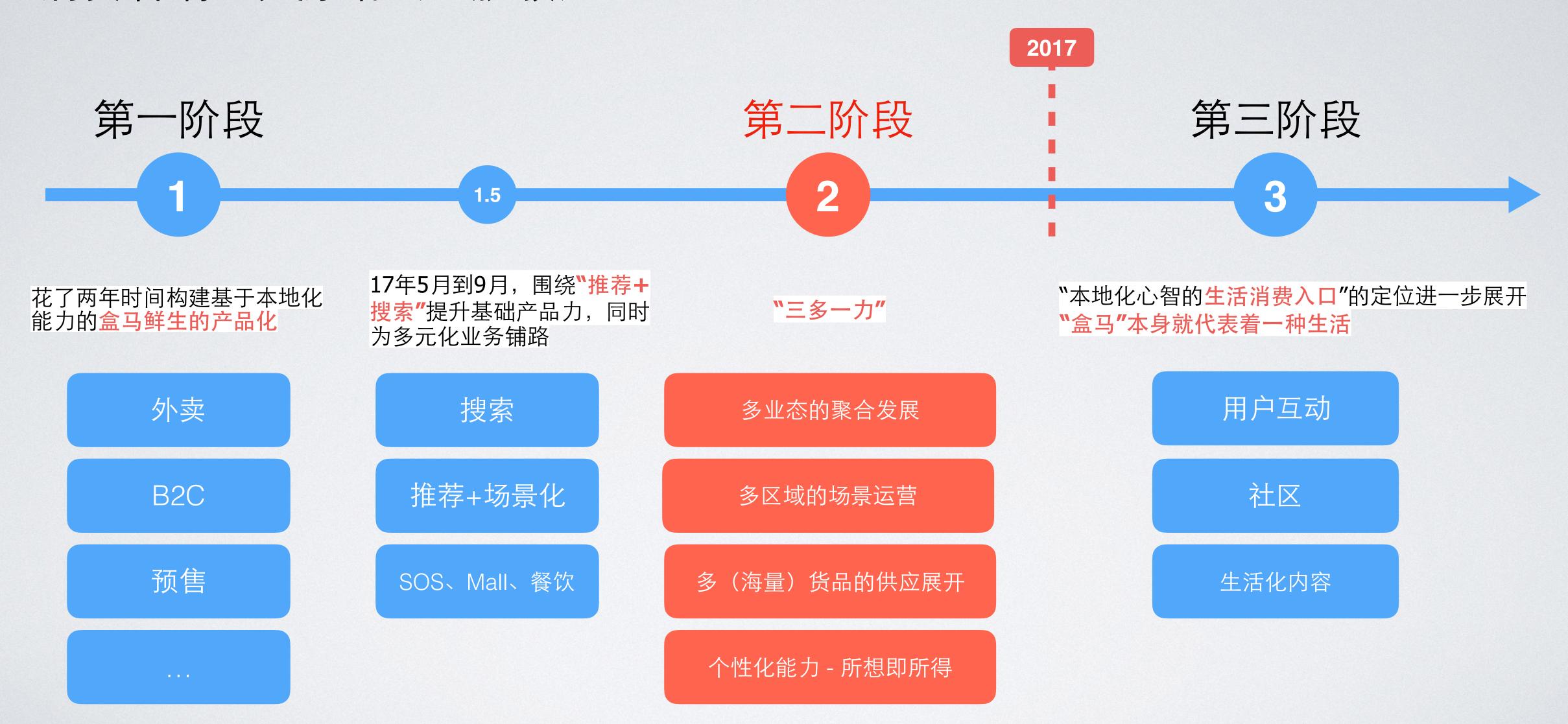
- ① 需求和项目分阶段,连成线来看
- ② 从过去的状态和问题推导下一阶段业务方向
- ③ 从下一阶段的业务挑战推导技术挑战
- ④ 从技术挑战拆解技术项目

在业务路径上,需要"从点到线"

上半年所在的业务基于项目维度的。



消费者端业务脉络(已脱敏)



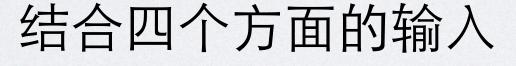
面向第二阶段的"三多一力",技术上应该拆解的事情

终端工程架构垂直 业务垂直,分布式解耦 支持"三多"的ERP Homs

供应链能力 缺货补货智能化 多业态的模型打通 mall,餐饮,美业,生活 商品、库存....
全链路升级

推荐导购能力 个性化,场景化,智能化

门店扩张 体量扩大 运营复杂度增强 系统负载爆发 客诉问题放大



1. 盒马业务因为"民生",因为"线上线下"的关键词,对于消费体验问题的"零容忍"

2. 业务发展带来的体量和复杂度本身对于"稳定性"要求的放大

3.对于像盒马这种每个项目都是横向,都是长链路协同的场景,业务支撑的能力需要被打破

4. 在用户端,体验是核心竞争力,产品闭环完成后,体验将是打用户心智的核心(包括线上线下)______



"L型纵横" 终端架构

"不可降级"的消费体验

"零容忍" 的稳定性

专注三件事三步走

第一步 (7~10月)

为业务的多元化平行扩展,并行开发迭代提供可能性

第二步 (11~1月)

能力短板的补齐, 核心中间件能力的升级建设

第三步 (2~4月)

研发支撑周边能力的整体提升

敏捷

研发模式**2.0** Bundle分治 体验

性能专项治理 内存、帧率、启动时间 稳定性

数据质量 监控+预警

动态化+组件化 Dinamic&协同组件化

基础体验三要素定位&容器&内容

稳定性能力 Monkey、Hotpatch、灰度

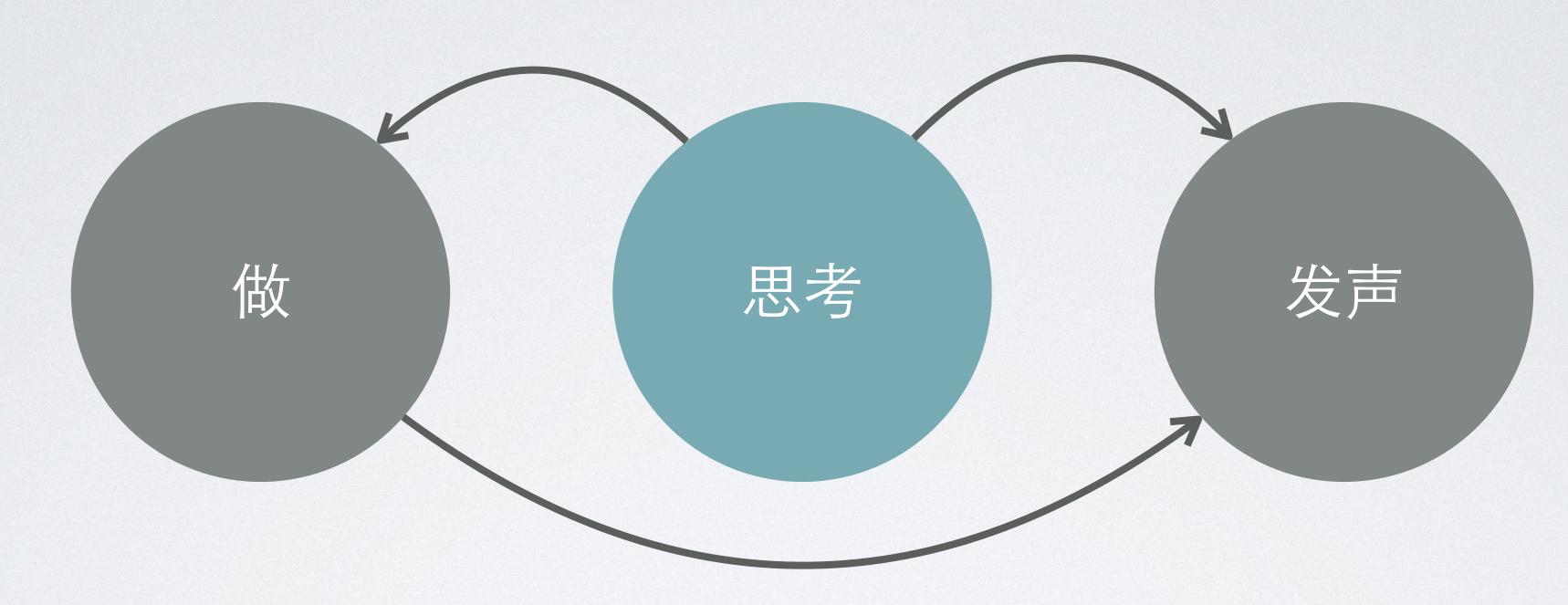
新工程能力 动态部署,开发者服务 互动能力 AR,语音,线下结合

文化生根 零容忍稳定性文化 技术架构和领域模型需要"从面到点"

技术人成长的三个关键词

三个关键词

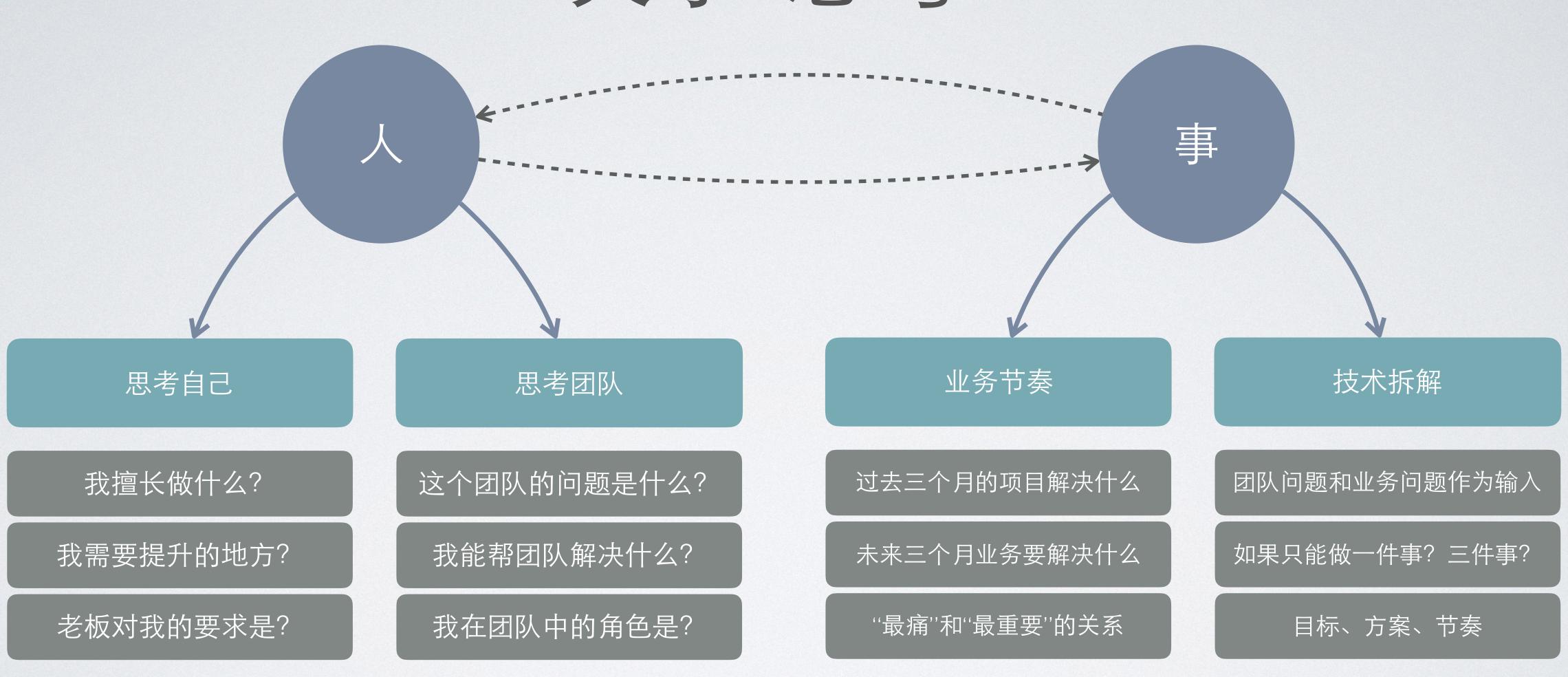
有目标的做分阶段的做的数型。



执行的总结思考的传达

思考是核心 Why有时候比How更重要

关于"思考"



关于"做"

Why 方案 节奏

大家往往一接到事情就埋头扎到"怎么做"这个维度上。可是比方案更重要的是"why"和"目标"

举个"稳定性"的例子

拿"稳定性"举例子

因为要"稳定性" 所以我们要做: 上报系统 crash治理 快恢能力 灰度发布

为什么?

因为故障多,稳定性不好

为什么故障多?

可能的情况

现象

继续挖

团队意识问题? 业务链路复杂度问题? 监控问题? 还是因为场景变了,方法和方式的问题?

找到本质,最重要的3点

- A. 流程不规范,意识问题
- B. 线下人肉上报,不合理
- C. 场景变了,套模的方式不可行

再抽象,可量化 形成目标

- I. PIP2监控发现率90%
- 2. PIP2恢复时长30min

再拆解



- I. 入KPI, 提意识
- 2. 全链路梳理,全链路压测模型
- 3. 监控全覆盖
- 4. 变更管控全接入
- 5. 异地多活
- 6. 监控&发布以门店为单位
- 7. 禁止营业时间发布
- 8. ...

关于"发声"

发声的本质

思考的传达

帮助自我梳理和总结

前端的核心竞争力

还原技术人的本质

