新型内容流服务架构方式

范怀宇 @ 轻芒

范怀宇

2009 年清华大学毕业，加入网易有道

2011 年加入豌豆荚，负责技术研发

2016 年底作为联合创始人创办轻芒

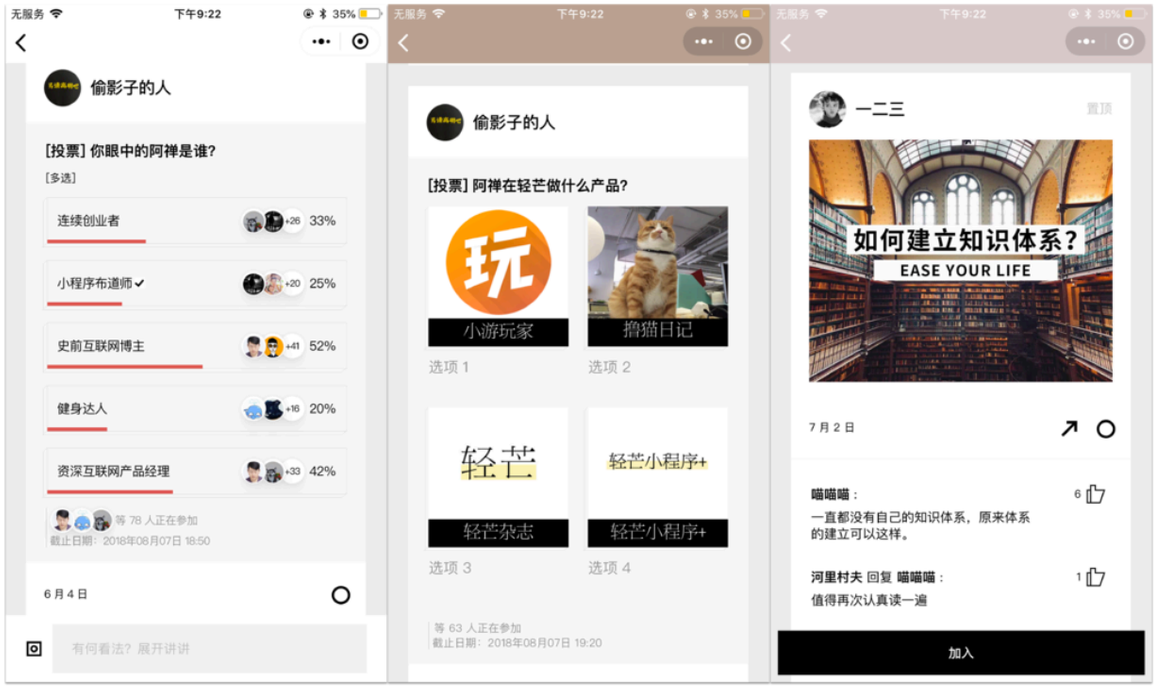
面向读者——

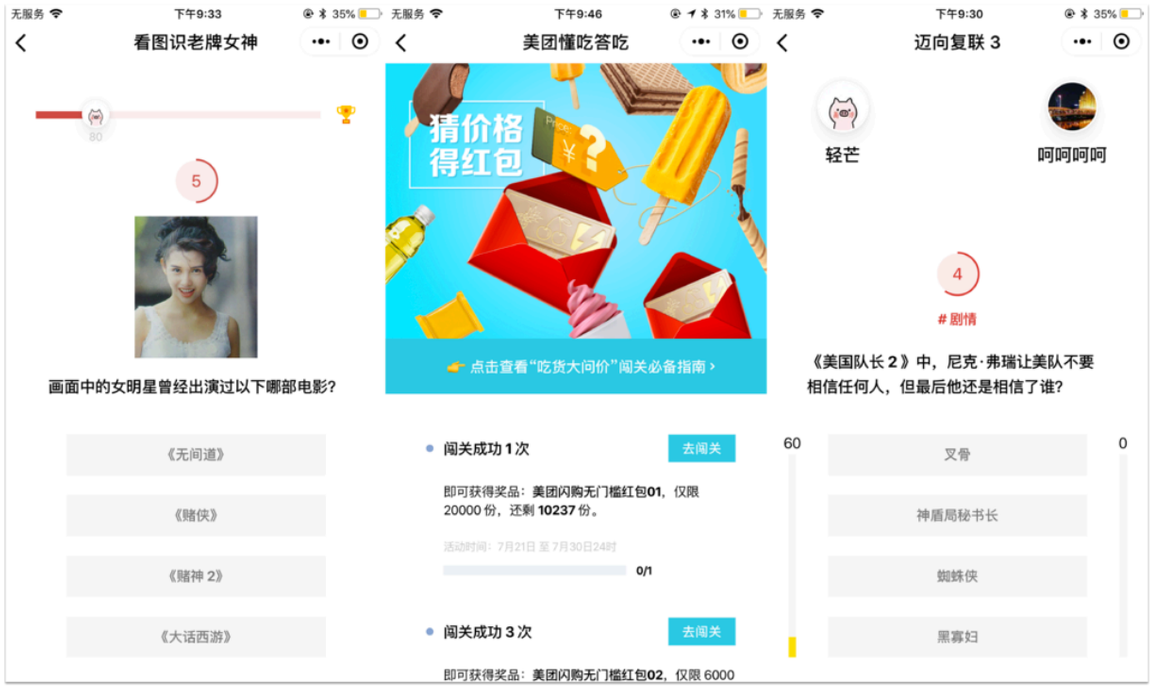
轻芒杂志

面向内容创作者——

轻芒小程序+


无服务
下午9:16
38%
Topbook
首页
课程
计划
选题
活动
七月二十三日
高效生活视频杂志。
EASE YOUR LIFE
了解详情
资源
思维
应用
常识
技能
人物
精选
个人
无服务
下午9:00
39%
mp.weixin.qq.com
长春长生生物冻干人用狂犬病疫苗生产过程中存在记录造假等严重违反《药品生产质量管理规范》
XIAOHS等 10 人 马克这段话。
XIAOHS： 长春长生疫苗事件引起无数人的震惊与恐慌，情绪过后，更需要的是寻求解决和改进之法，包括受害者如何依法索赔、犯罪逾规者该受到什么法律制裁、而其他国家又有什么值得中国学习的经验……轻芒邀请律师石赫（公众号 查尔斯学长）作相关解读。
生物前世今的内容更是让人感到不寒而栗："十年后再回首，他们手中已经掌握了中国疫苗的半壁江山——最大的乙肝疫苗企业、最大的流感疫苗企业、第二大水痘疫苗企业、第二大狂犬病疫苗企业……**他们生产的疫苗，每天都源源不断，注入你和你孩子的身体中。**"
石赫等 13 人 马克这段话。
石赫： 中国的疫苗生产本身就是垄断性行业，就像电信、石油一样，都是暴利行业，如果这样还不能满足他们的获利心理，那么他们的心真的是黑透了。
因，众生畏果。而在坏人眼中，无所谓因果，他们的价值观和我们是平行宇宙，但他们不断发射武器来摧毁我们的星球。
XIAOHS等 49 人 马克
分享给好友
全世界非常多的国家
无服务
下午9:02
38%
晚安荒野
七月二十三日
荒野气象台
晚安荒野
首页
个人

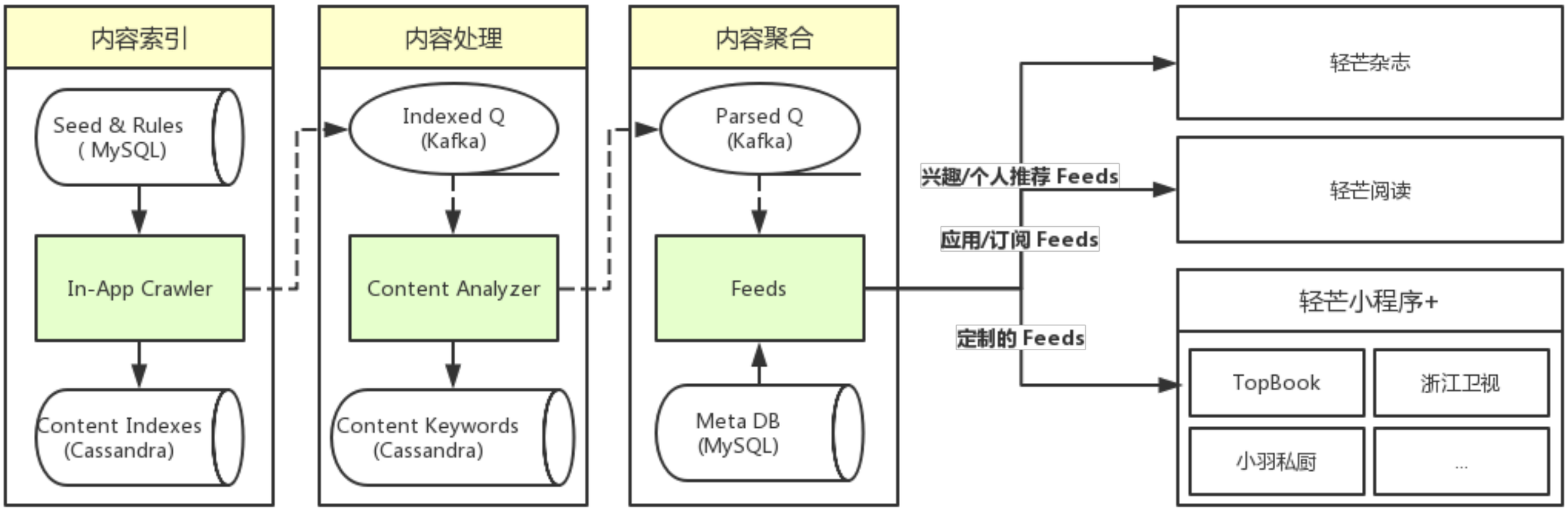

无服务
下午9:22
35%
偷影子的人
[投票] 你眼中的阿禅是谁?
[多选]
连续创业者 +26 33%
小程序布道师 ✓ +20 25%
史前互联网博主 +41 52%
健身达人 +16 20%
资深互联网产品经理 +33 42%
等 78 人正在参加
截止日期：2018年08月07日 18:50
6月4日
有何看法？展开讲讲
偷影子的人
[投票] 阿禅在轻芒做什么产品?
玩
小游玩家
撸猫日记
选项 1
选项 2
轻芒
轻芒杂志
轻芒小程序+
轻芒小程序+
选项 3
选项 4
等 63 人正在参加
截止日期：2018年08月07日 19:20
一二三
置顶
如何建立知识体系?
EASE YOUR LIFE
7月2日
喵喵喵：
一直都没有自己的知识体系，原来体系的建立可以这样。
6
河里村夫 回复 喵喵喵：
值得再次认真读一遍
1
加入


无服务
下午9:33
35%
看图识老牌女神
80
5
画面中的女明星曾经出演过以下哪部电影?
《无间道》
《赌侠》
《赌神 2》
《大话西游》
无服务
下午9:46
31%
美团懂吃答吃
猜价格得红包
点击查看"吃货大问价"闯关必备指南 ›
闯关成功 1 次
去闯关
即可获得奖品：美团闪购无门槛红包01，仅限20000 份，还剩 10237 份。
活动时间：7月21日 至 7月30日24时
0/1
闯关成功 3 次
去闯关
即可获得奖品：美团闪购无门槛红包02，仅限 6000
无服务
下午9:30
35%
迈向复联 3
轻芒
呵呵呵呵
4
剧情
《美国队长 2》中，尼克·弗瑞让美队不要相信任何人，但最后他还是相信了谁?
60
0
叉骨
神盾局秘书长
蜘蛛侠
黑寡妇

提纲

- 轻芒内容服务的设计目标
- 基于 Scala + Akka 模型来构建内容服务
- 新架构带来的挑战和应对

设计目标



- 每天数十万的新内容被检索
- 需要计算数十万的应用、兴趣内容流
- 需要为上千万的用户计算内容流
- 为数万个小程序提供内容支撑

- 成熟的技术方案倾向于平衡
  - 在性能、稳定性、可扩展性、研发效率，可维护性等诸多方面相对平衡
- 最小的人力投入，保持最大的弹性
  - 研发效率：统一编程模型，减少中间件的种类，和熟悉的技术栈兼容
  - 弹性：降低代码规模，在用户规模变化下开发模式相对稳定
  - 性能和稳定性：可以妥协单机性能和稳定性，可以通过扩容来临时解决问题

基于 Scala + Akka 的服务设计

Scala

- JVM 方言
  - 可以无缝和 Java 生态相融合
- 混合了面向对象和函数式编程的特征
  - 比之 Java 更为灵活，简练
  - 过于灵活，学习曲线比较陡峭
- 常用在数据分析和处理领域
  - 被 Spark，Kafka，等大型开源项目所使用
  - 被 Twitter，Linkin 等公司广泛应用

Akka

- Akka 是围绕消息机制来构建的分布式开发框架
  - 对开发者屏蔽掉分布式的诸多细节
- Akka 是 Actor 模型在 JVM 上的实现
  - Actor 是分布式系统下的对象，异步通信、全局寻址
- 可以部署成一个去中心的集群
  - 同构的部署策略更利于压榨单机性能
- 改变了传统 Web Server 的编程模型
  - 以及，改变了服务部署的方式

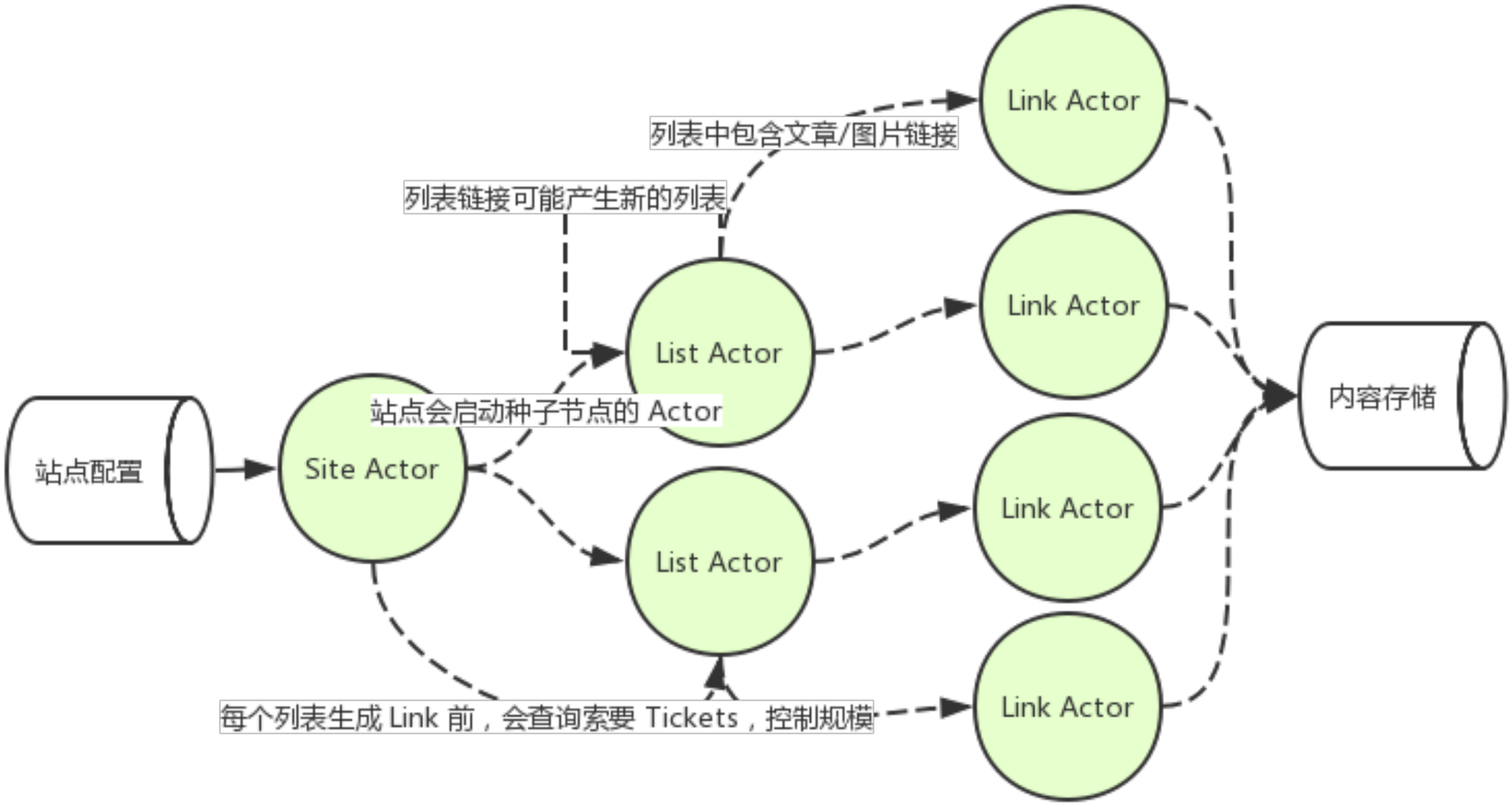
面向对象的编程

- 对象是对实体的抽象
  - 将数据和操作封装在一起
  - 让对象之间的关系清晰而明确
- 面向对象在分布式系统下碰到的问题
  - 数据同步
  - 寻址机制是单机化的
  - 同步的操作无法充分利用系统资源

统一的开发模型

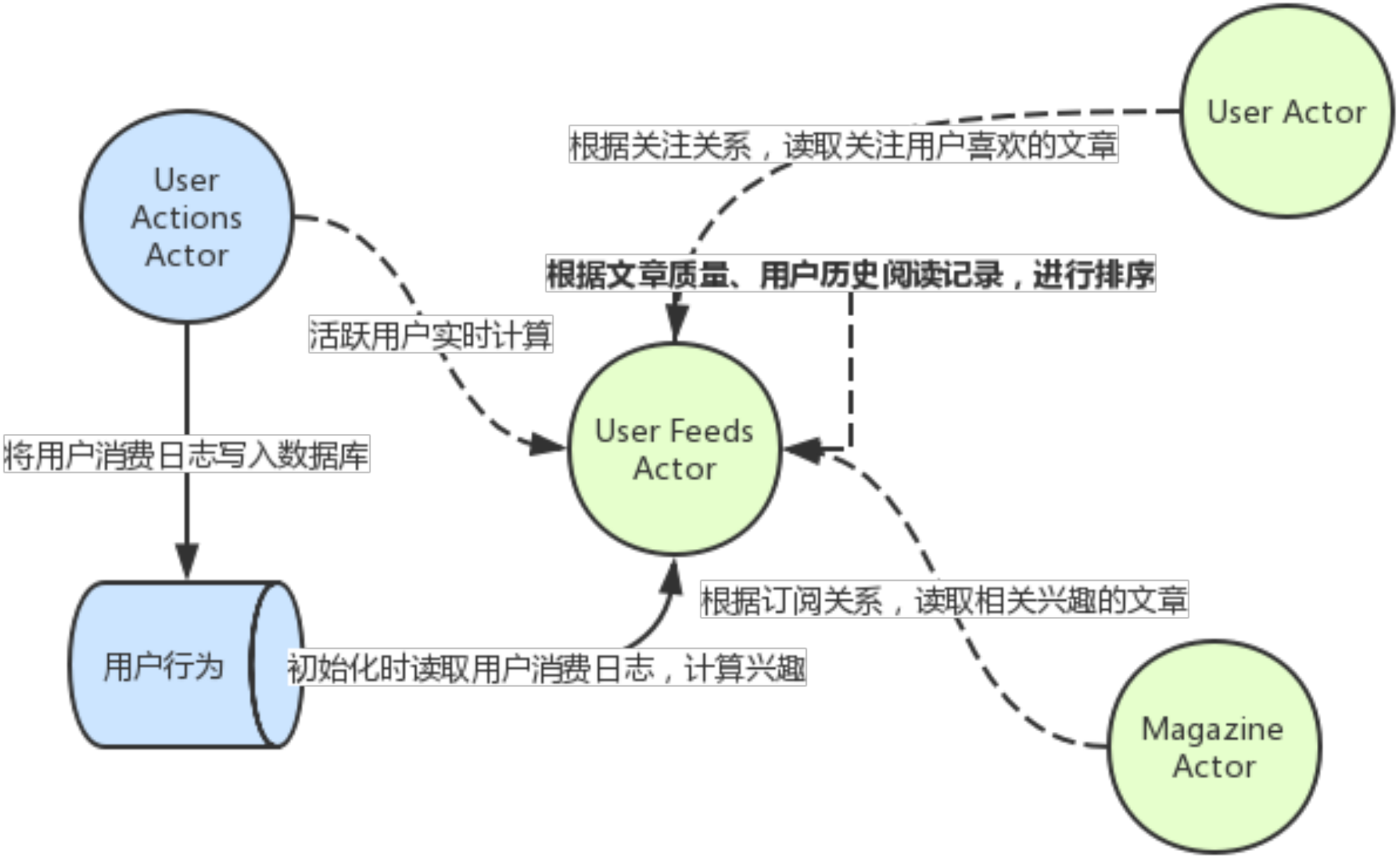
- 定义系统中的 Actor
  - Actor 是分布式下的对象
  - 理解业务，对实体进行抽象，规划好 Actor 在分布式系统下的地址
- 初始化 Actor 中的数据
  - 初始化时把所需数据加载到内存
  - 后续操作基本在内存中，数据 + 操作一体化
- 处理消息
  - 基于消息驱动的异步模型

轻芒爬虫服务的实现


Link Actor
列表中包含文章/图片链接
列表链接可能产生新的列表
List Actor
Link Actor
内容存储
站点配置
Site Actor
站点会启动种子节点的 Actor
List Actor
Link Actor
Link Actor
每个列表生成 Link 前，会查询索要 Tickets，控制规模

- 从流式的架构，变成了对象式的架构
  - 调度模块，解析模块，索引模块 …
  - 站点、列表、链接，很容易进行追踪和维护
- 解除了外部依赖
  - 没有中间的消息系统、存储服务之类的
  - 可轻松的单机部署，也可以部署成集群

轻芒内容流服务的实现


User Actor
根据关注关系，读取关注用户喜欢的文章
User Actions Actor
根据文章质量、用户历史阅读记录，进行排序
活跃用户实时计算
User Feeds Actor
将用户消费日志写入数据库
根据订阅关系，读取相关兴趣的文章
用户行为
初始化时读取用户消费日志，计算兴趣
Magazine Actor

- 统一了用户 Feeds 流的实现
  - 推拉一体
  - 核心用户的 Feeds 会自然的停留在内存中提升效能
- 天生具有实时计算的能力
  - 初始化是从数据库中加载数据
  - 实时监听用户行为数据
  - 由于大量关联 Actor 位于内存中，实时计算速度较快

优点

- 编程模式统一
  - 对 Actor 的设计和抽象是核心
- 在分布式下的部署和开发变得简单
  - Akka 承包了很多细节
  - 全异步更易于分布式
- 机器会被充分利用
  - 缓存和计算一体，会充分利用内存和 CPU

新架构的挑战和应对

Actor 的冷启动

- 冷启动的时长，影响整个系统的响应时长
  - Actor 的调用关系可能很繁杂，需要做好自己
- 降低 Actor 的抖动，把重要的 Actor 留在内存中
  - 根据业务特征来调整存活相关的参数
  - 扩大集群规模
- 通过临时缓存来降低再次启动的耗时
  - 个人 Feeds 会在退出的时候记住没有消费完的内容
  - 在下次启动的时候优先使用，异步初始化
- 建立有效的性能监控
  - 记录各个步骤的日志，汇总到阿里日志服务进行分析

系统资源的调度

- 分治经营
  - 每个 Actor 控制存活时长、占用资源
- 集中调度
  - 收集系统中全部 Actor 的数量和资源消耗情况
  - 根据时间、资源等对 Actor 进行调度
- 动态平衡还是很困难
  - 比较容易导致雪崩
  - 高质量的实现和监控每个 Actor 是关键

部署的挑战

- 大集群 vs 微服务
  - Akka 本身就包括了集群的解决方案
  - 引入微服务会在开发模式上导致一定的不统一
- 包含全部业务的集群会在运维上带来挑战
  - 基于 Plan B 的发布和容灾方案
- 做到任何一个 Actor 都可以无损的重启才可以实现更轻松的发布

Akka 集群的挑战

- 更频繁的序列化开销
  - 屏蔽了部署细节，会有更多的数据会被序列化
  - 单机性能的降低可以通过集群化来解决
- 无中心集群的稳定性
  - 心跳检测需要容忍度，在容忍度下服务可能不可用
  - 各种阈值的配置需要仔细把控，不然会引起抖动

总结

- 基于 Actor 的编程模型在很多场景下可以统一开发模型
  - 降低理解和设计的复杂度
  - 长期看，简化开发带来的价值是更大的
- 有效减少了中间件的使用，降低为运维复杂度
  - 但需要更熟稔 Akka 集群的运维

- 新的开发模式，需要新的思维
  - 对业务的抽象比存储的设计更重要
  - 优化局部可以带来整体的优化
  - 围绕新的模式来设计日志和监控机制

加入我们

hello@qingmang.me

Q & A