高并发实时直播弹幕研发实践

zhanghu@yunba.io
@Tiger_张虎

Yunba.io 云巴

直播间的特点

- 直播间的人数 > 聊天室
- 不能限制人数（500？1000?）

人数限制怎么来的

- 聊天室的人数限制
- 单服务器
- 单进程

直播面临的挑战

- 直播间，万级以上的实时互动
- 跨服务器
- 多进程并发

跨服务器

- 单一服务器接入数量限制
- 单一服务器发布消息吞吐限制

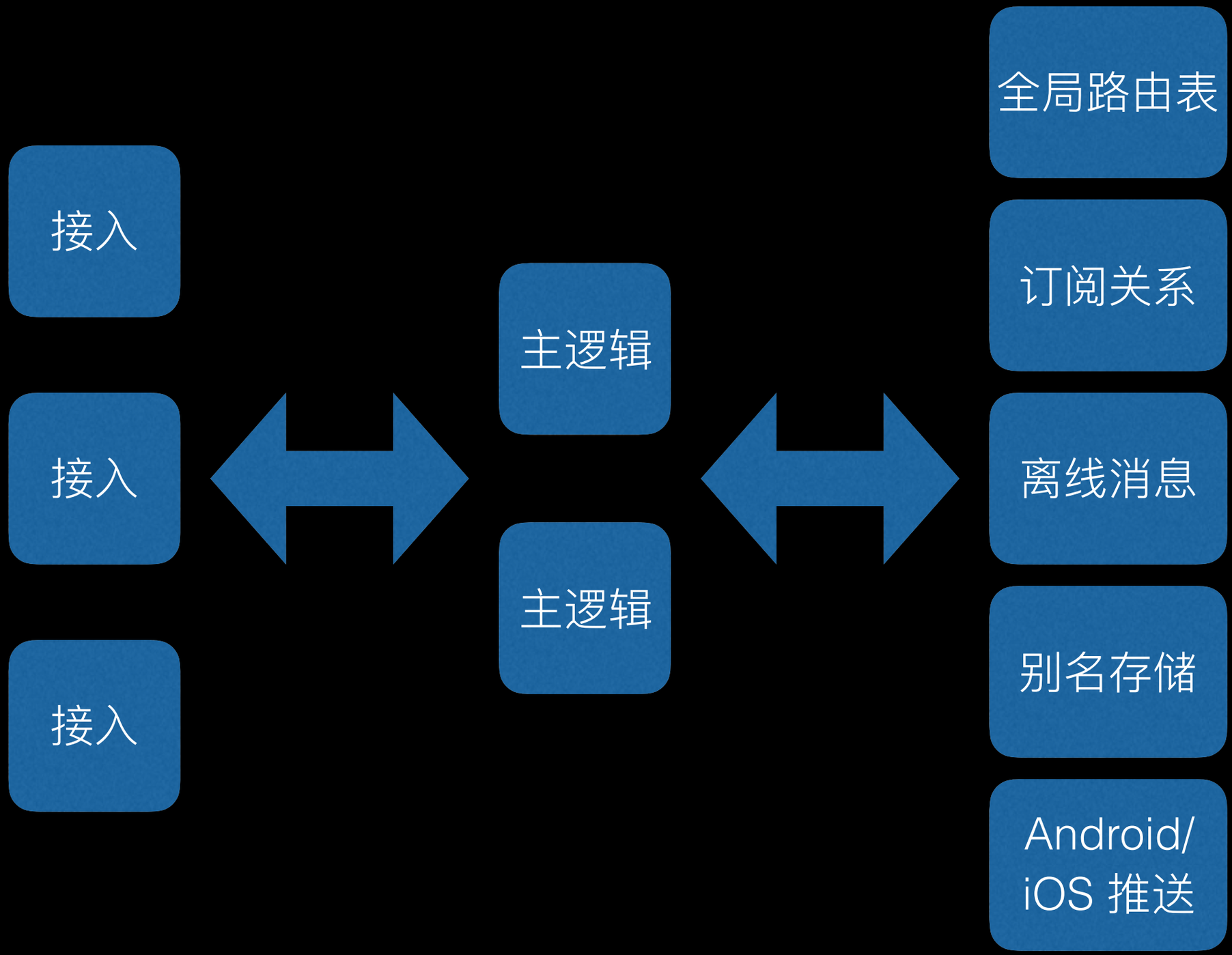
多进程并发

- 充分利用多核 CPU
- 减小一个循环的规模，降低延迟

云巴实时系统的设计

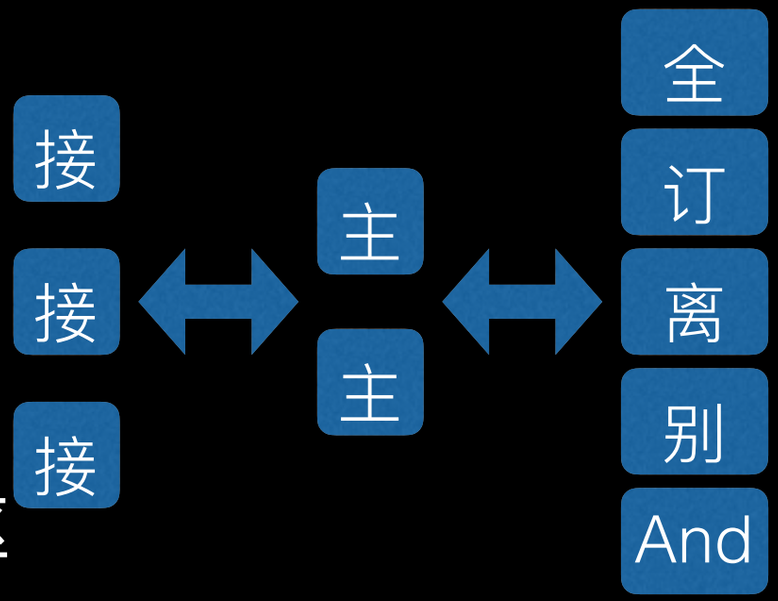
- 多层结构
- 微服务化

多层结构



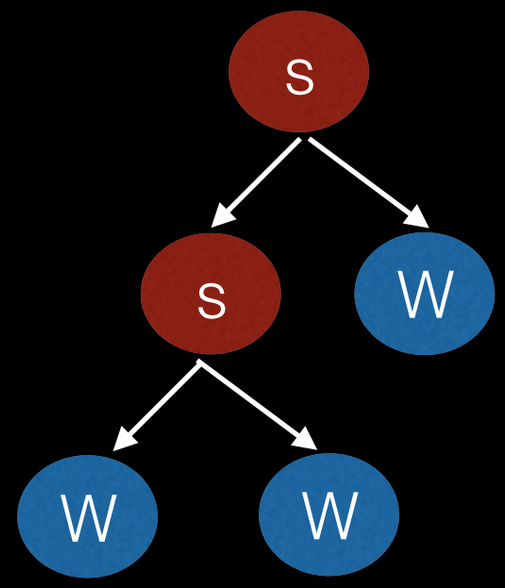
多层结构的特点

- 模块可独立运行
  - 提高开发、测试、部署的效率
- 细粒度扩容
- 隔离问题



微服务化

- 微服务是一个“老概念”
  - Erlang/OTP： 成熟已久的微服务架构
- 业务逻辑封装成一个 RPC Service
- RPC Service 部署为一个 OTP Worker



云巴实时系统的设计

- 海量轻量级任务
- 任务运行位置无关
- 水平扩展

轻量级任务

- 长连接：任务
- 请求：任务
  - 订阅频道
  - 发布消息

任务与运行位置无关

- 任务池
  - 动态把任务调度到不同物理机
- 数据独立存储

海量消息发布过程

- 订阅关系、路由 分片
- 切分任务发布执行
- 汇总结果

发布过程：订阅

- 订阅（加入一个直播间、关注一个主播）
  - 一个频道里万级、几十万级的 UID
    - 路由：发布消息需要
- 根据连接的接入服务器排序
- 控制分片大小，控制延迟（200 每片）

发布过程

- 一般过程（一）
  - 遍历列表里的每一个 UID，读取路由，逐一发消息
- 一般过程（二）
  - 遍历每一台接入服务器，发消息
  - 订阅关系需要保存在接入服务器

发布过程

- 云巴的消息发布 (类似 Map/Reduce)
- 发布任务计算 UID 列表分片
- 把分片分发给任务池，并行执行分片任务
- 发布任务汇聚分片任务的返回

发布过程：优点

- 有效控制最大延迟：200 ms 以内
- 平行扩展：扩大任务池，可以提升并发能力

谢谢

Q&A

Yunba.io 云巴