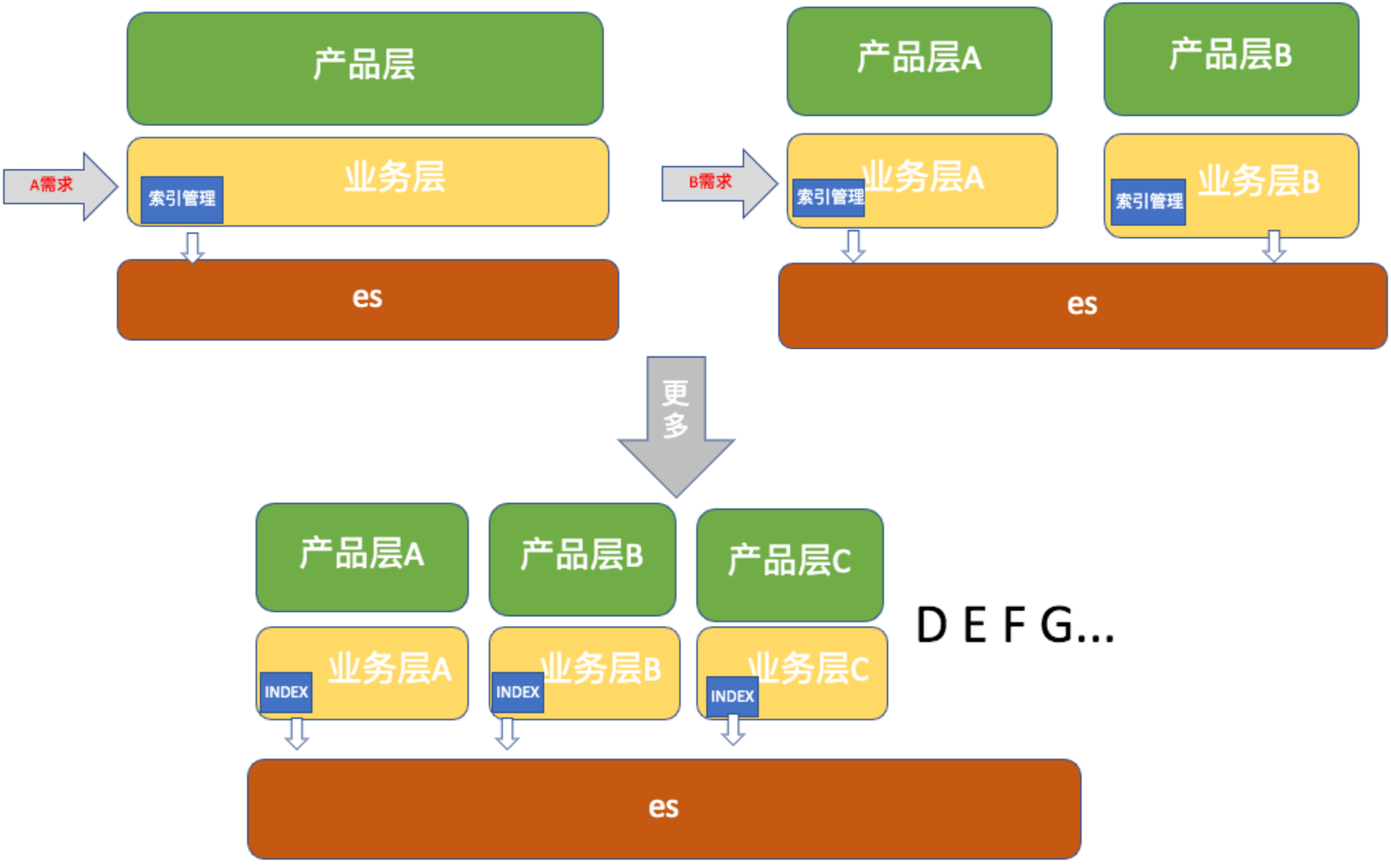
ES多数据源同步之道

字节跳动—张彪

分享内容

- 1 旧系统面临着的问题
- 2 为什么要做自研?
- 3 bukom 设计及其核心流程
- 4 bukom未来的规划

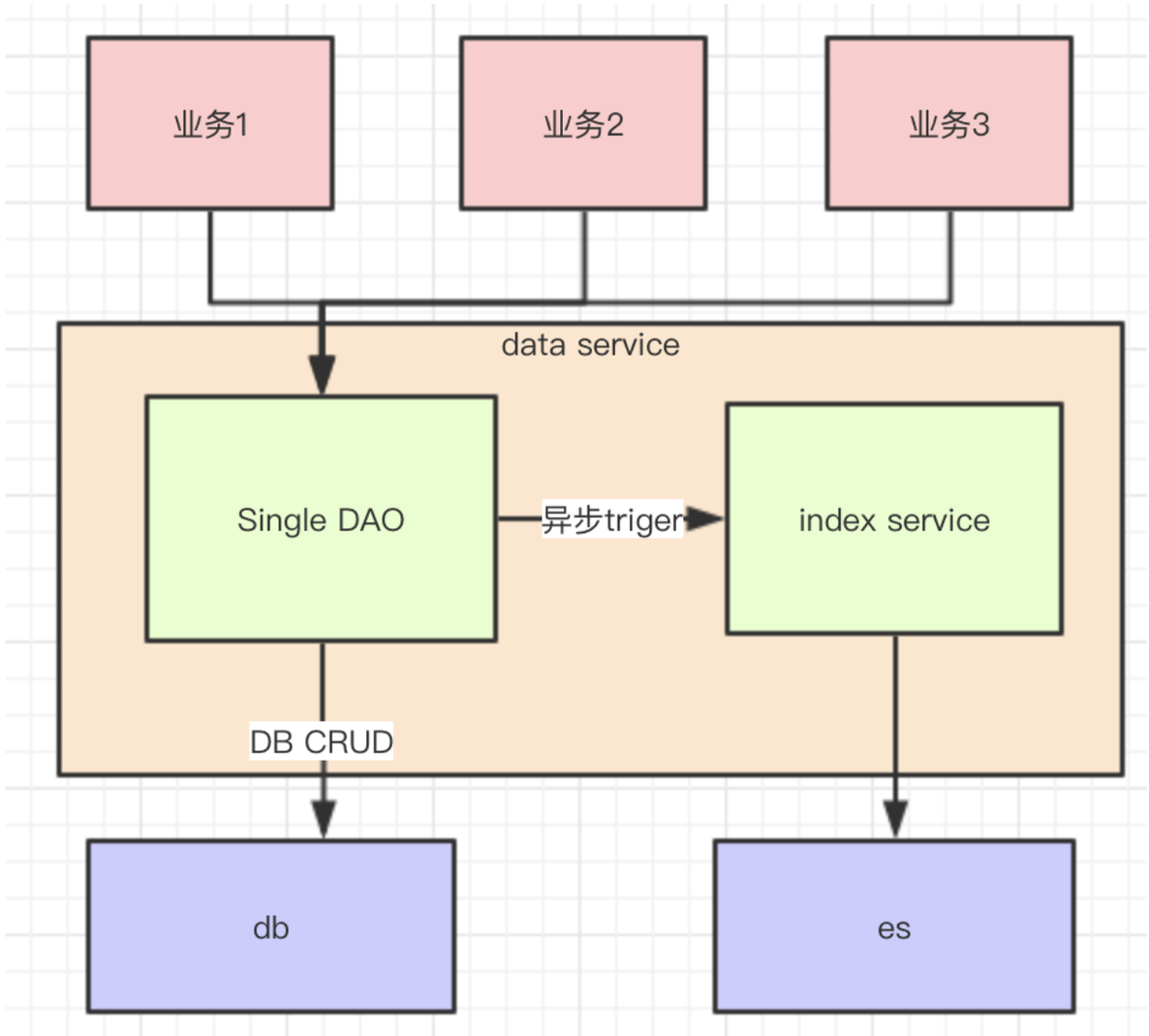
需求来了


产品层
A需求
索引管理
业务层
es
产品层A
产品层B
B需求
索引管理
业务层A
索引管理
业务层B
es
更多
产品层A
产品层B
产品层C
INDEX
业务层A
INDEX
业务层B
INDEX
业务层C
es
D E F G...

之前的架构

- 由于我们自己业务的特殊性(任何Aggregate的更新都会更新Aggregate root对应的索引)，即使采用横向扩展这种方式都无法实现。

- 因此我们之前采用的方案是对所有的DB的写操作进行收敛。DB操作完成后，异步触发索引同步。



痛点总结

- 1 每个业务都要花大量的时间写自己索引逻辑
- 2 业务代码和写ES代码强耦合
- 3 对于一些直接操作db的场景，无法触发同步
- 4 之前的架构，严重阻碍业务向微服务化架构进化

前期调研

- 基于以上痛点和诉求，我们尝试去开源产品或者成熟的平台寻找答案。

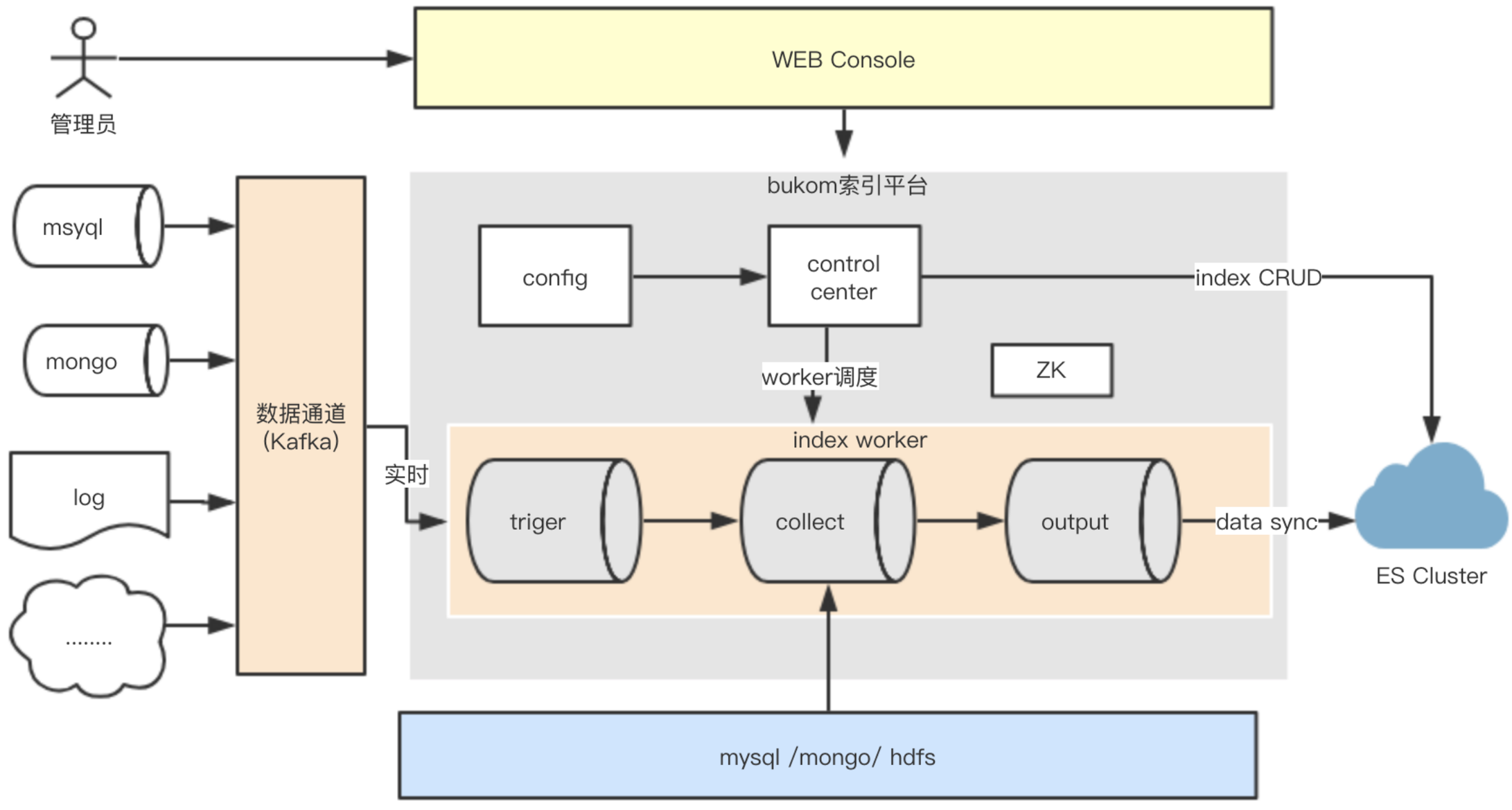
	Open search	canal	logstash
类型	平台	中间件	中间件
适用平台	阿里云	任何机器	任何机器
功能	搜索整体解决方案。但是主要侧重点在检索上，索引方面还是需要业务强关注doc处理。	采集binlog的模式。从mysql同步到kafka/es。支持一些简单的数据组装。	主动采集模式。多数据源同步到多目的地,对业务设计有侵入。支持一些简单的数据改写。
复杂doc支持	一般	较差	较差
语言支持	Restful/Java/Php	java	grok配置
结论	整体强大，但索引痛点未解决，且存在性能和安全问题	成熟，但功能单一，不是作为整体解决方案。	支持多数据源，多输出。不能做到时实，且数据处理功能单一。

bukom诞生

基于我们目前遇到的痛点，以及一些现成的产品并没法完全满足我们的需求。我们构建了自研的搜索索引平台：bukom。一期核心解决以下问题：

- 解藕业务代码和索引相关代码
- 以可配置方式，快速支持mysql->es的搜索需求
- 支持复杂doc组装(包括跨库处理)
- 解决数据库的直接更新无法triger

Bukom架构


管理员
WEB Console
bukom索引平台
msyql
mongo
log
.....
数据通道
(Kafka)
实时
config
control
center
index CRUD
worker调度
ZK
index worker
triger
collect
output
data sync
ES Cluster
mysql /mongo/ hdfs

bukom核心组件

bukom中最核心的组件有两个：control center和index worker

- control center：整个系统的控制中心，负责配置管理，es index的CRUD,以及增量/全量 job调度。

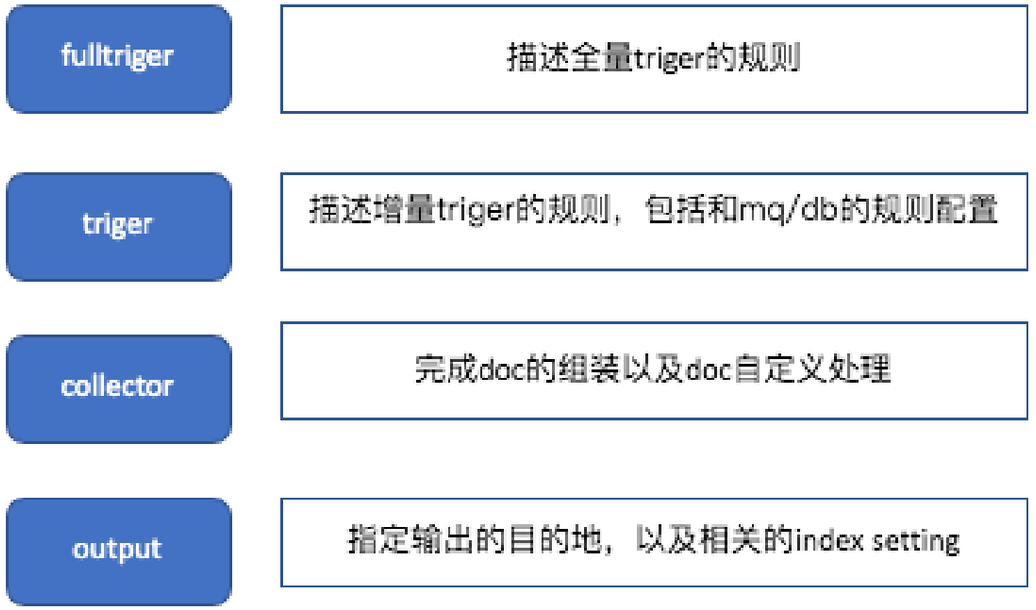
- index worker：索引相关的核心执行者，负责增量/全量的索引。

Control Center

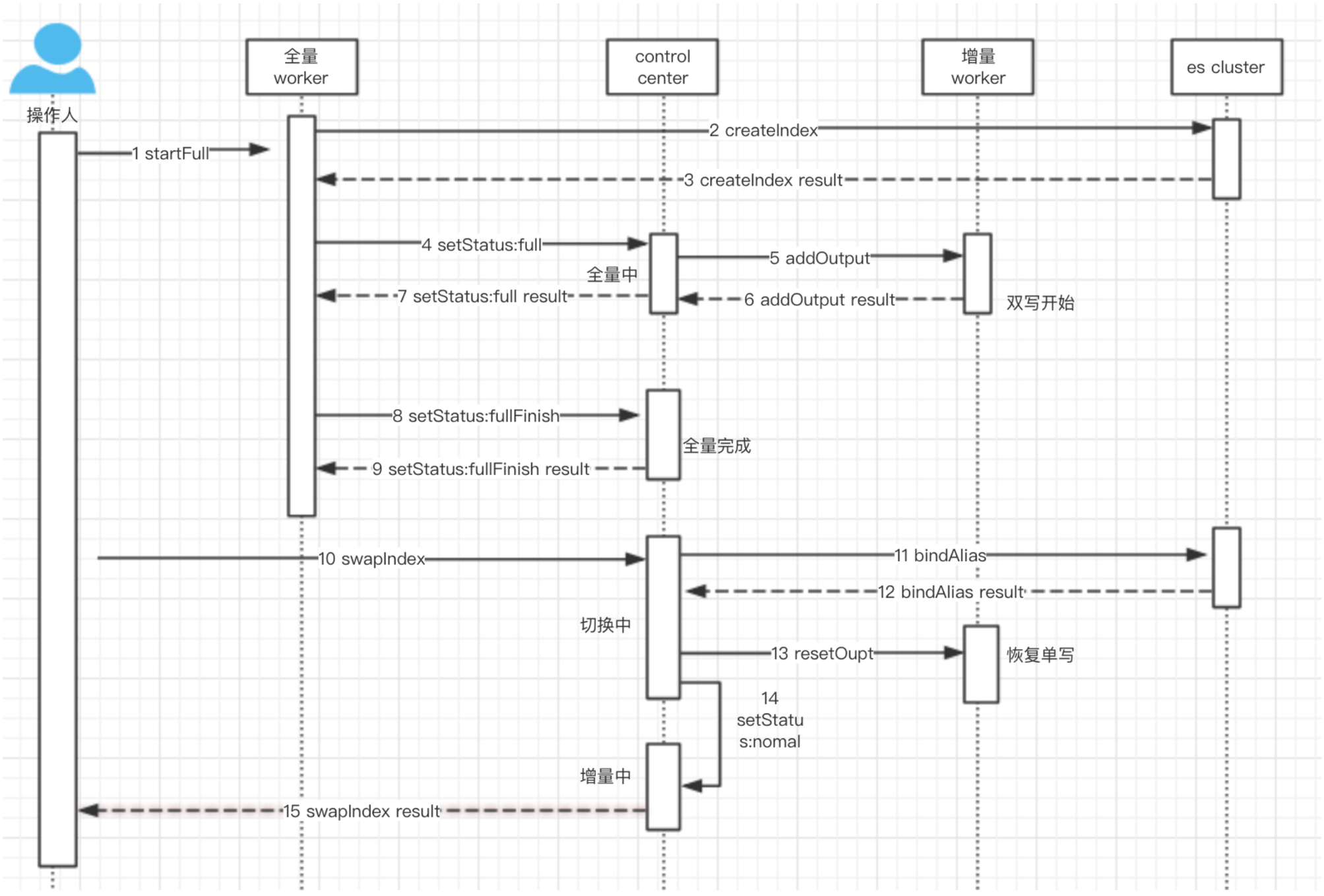
- 1 负责接口web端的操作指令
- 2 负责es Index 的CRUD,以及状态查看
- 3 index worker 的调度工作
- 4 维护整个索引的状态机流转

Index Worker

- Index worker内部包含四个流程（右图所示）
- index worker 实际会分成两种worker: 增量和全量。每个config会同时启动两个进程，由control center进行云平台调度。
- 全量worker是以cronjob的方式启动；增量以常驻worker方式启动
- 增量和全量不直接通信



索引流程


操作人
全量 worker
control center
增量 worker
es cluster
1 startFull
2 createIndex
3 createIndex result
4 setStatus:full
全量中
5 addOutput
7 setStatus:full result
6 addOutput result
双写开始
8 setStatus:fullFinish
全量完成
9 setStatus:fullFinish result
10 swapIndex
11 bindAlias
12 bindAlias result
切换中
13 resetOupt
恢复单写
14 setStatus:nomal
增量中
15 swapIndex result

项目中遇到的问题

- 1 可靠性和吞吐量问题
- 3 插件的模式问题
- 4 数据组装带来的db压力

新的挑战

一期我们比较好的解决了之前最核心的诉求。但一切到此为止？
实际上在我们上线后不久，新的诉求会不断涌过来。

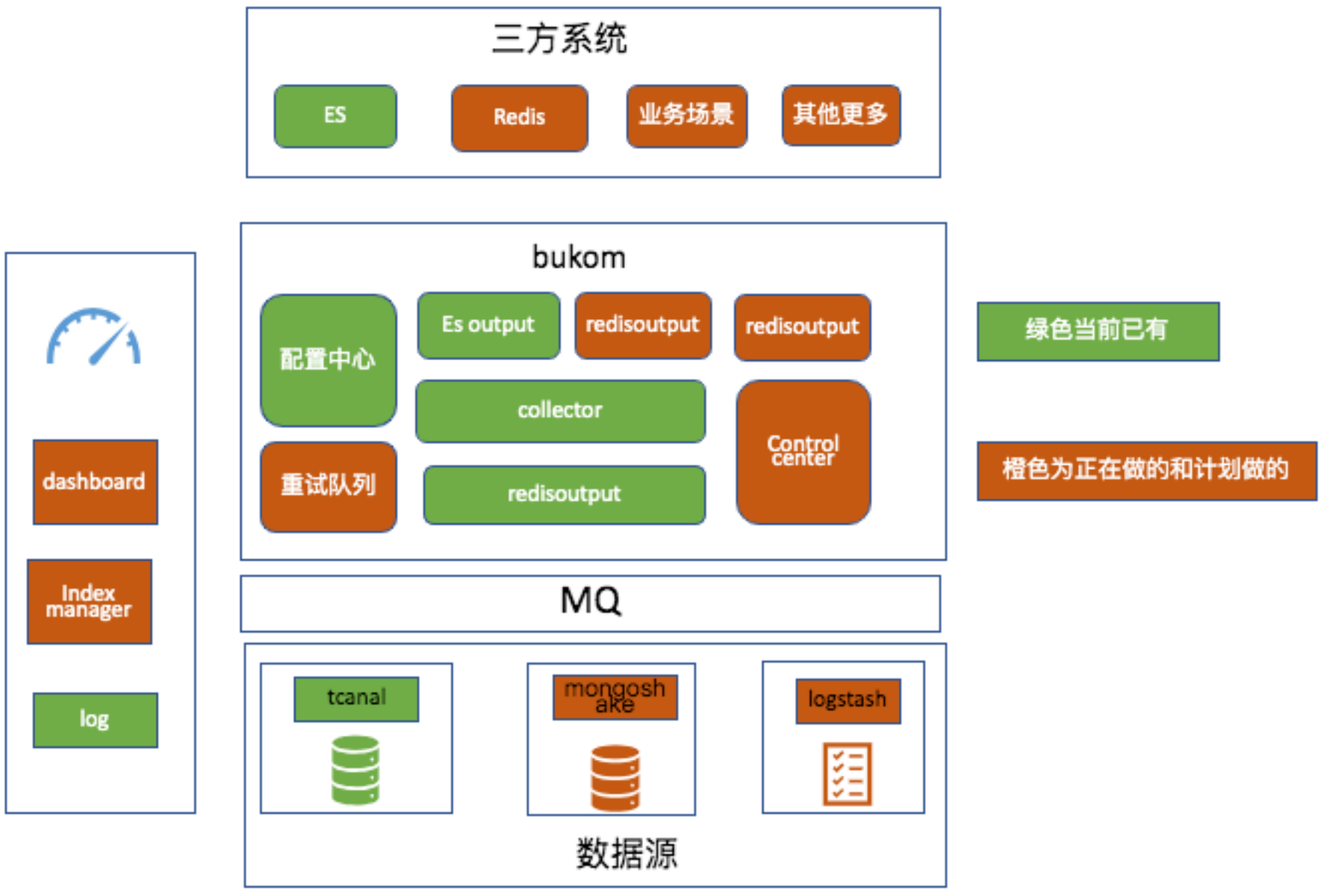
- 用户对系统产生了更大的诉求。特别是多数据源
- 除了es，缓存系统/业务系统的支持也很期待
- 系统稳定性和可靠性如何增强

新的定位

新得挑战使得我们进一步思考我们的系统，未来的定位是什么？解决什么问题？还能为业务做哪些帮助？　新的定位：数据工厂

- 统一化：把各种数据源同步的逻辑进行统一处理。
- 平台化：完成bukom从工具到平台的转变。提供多数据源同步的整体解决方案。
- 配置化：更灵活的配置化，减少接入的代价
- 健壮性：监控/日志/容灾/重试机制

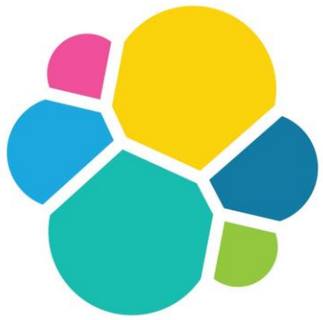
数据工厂


三方系统
ES
Redis
业务场景
其他更多
bukom
配置中心
Es output
redisoutput
redisoutput
collector
Control center
重试队列
redisoutput
MQ
tcanal
mongoshake
logstash
数据源
dashboard
Index manager
log
绿色当前已有
橙色为正在做的和计划做的

QA

Thank you

–张彪　邮箱：zhangbiao.007@bytedance.com



elastic
中文社区

专业、垂直、纯粹的 Elastic 开源技术交流社区

https://elasticsearch.cn/