

# NDC

## 平台化的海量结构化数据同步传输系统

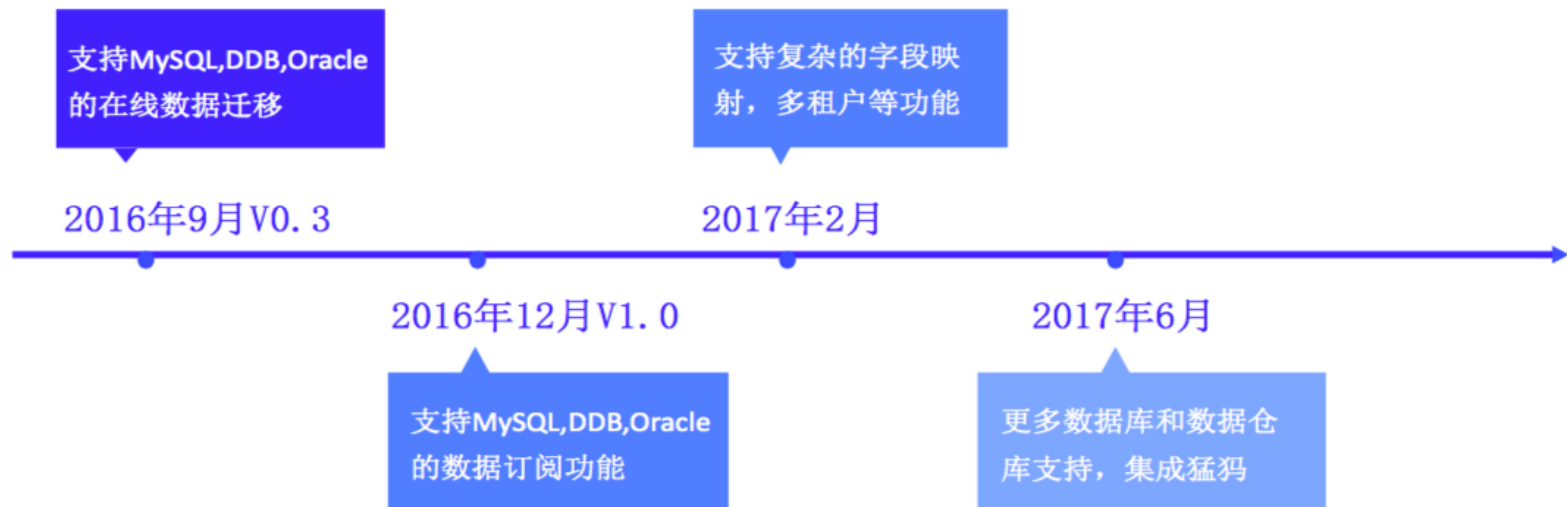
大数据平台组  
杨宇佳

# 大纲

- 应用场景
- 产品形态
- 系统架构
- 核心实现

# NDC是什么

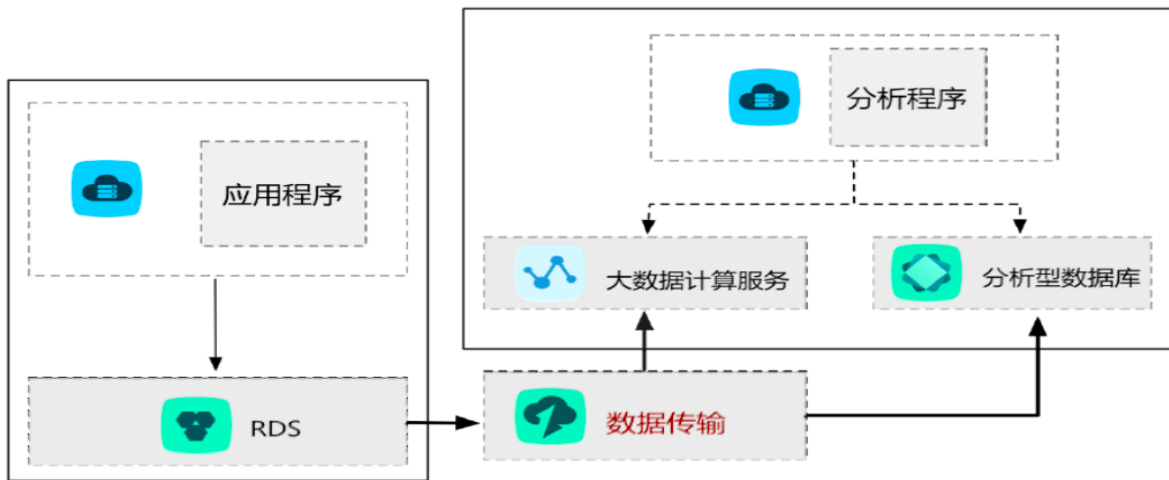
- 全称Netease Data Canal，是网易大数据平台组研发的一套用于结构化数据同步，迁移和订阅的中间件系统。



# 应用场景

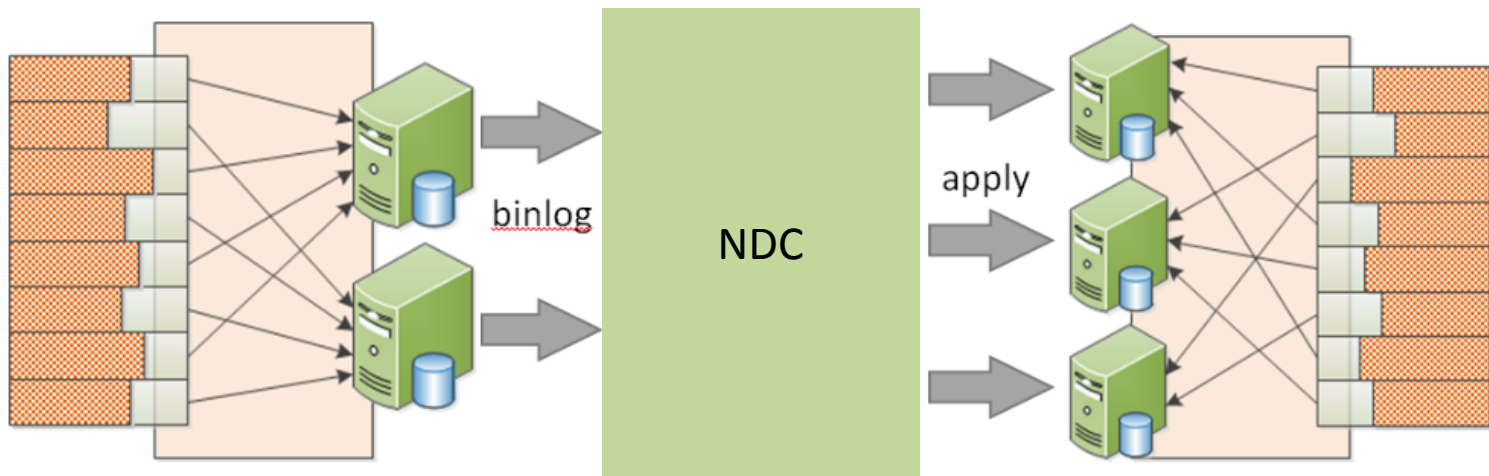
- OLAP数据同步

- OLTP实时同步到OLAP系统 ( Kudu, HBase, GreenPlum )
- OLTP同步到队列，定时merge到OLAP系统 ( Hive ,HDFS )



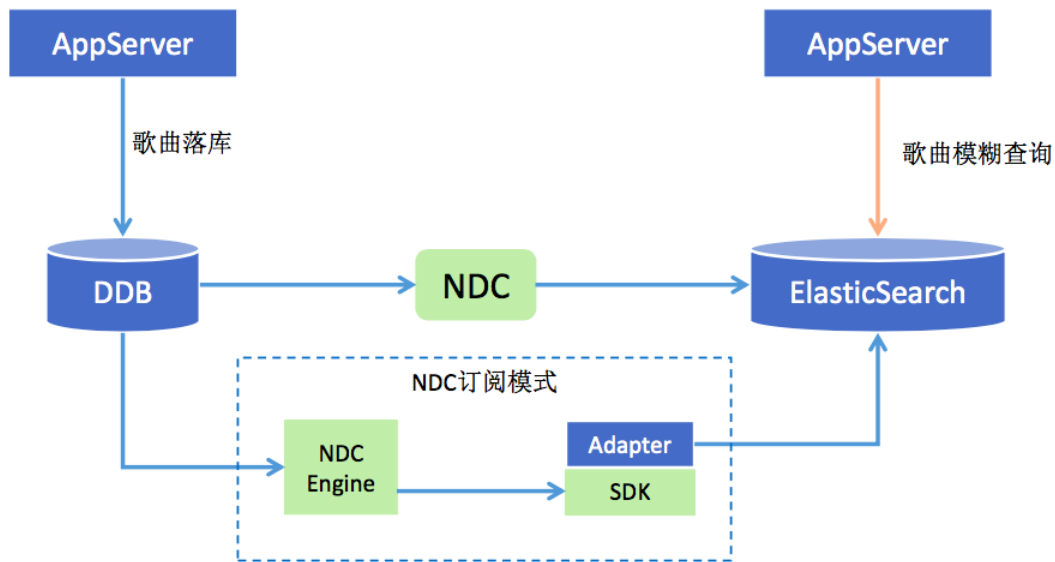
# 应用场景

- DDB在线数据迁移
  - 在线扩容，机器迁移，更改均衡字段等



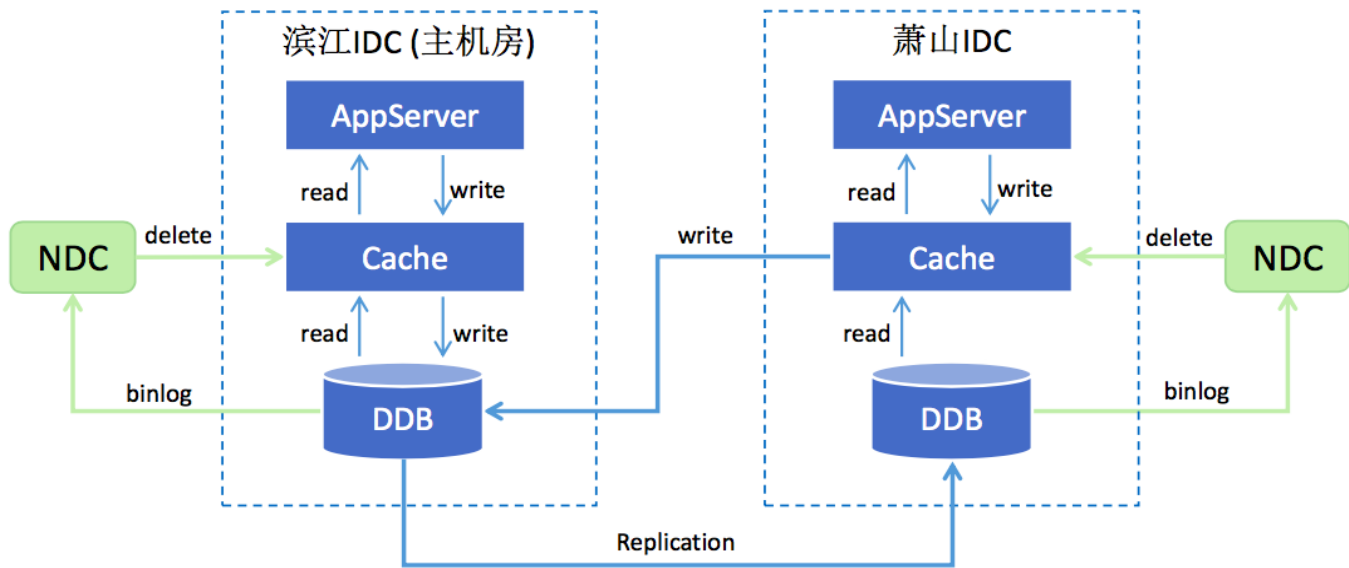
# 应用场景

- 数据库第三方索引更新
  - 全文索引、地图索引等



# 应用场景

- 多机房缓存淘汰
  - 原则：同步优先于淘汰



# 应用场景

	需求
应用方视角	数据迁移：在线扩缩容，异构数据库在线迁移
	数据同步：同构、异构的跨机房、跨域、跨国的实时数据同步
	数据订阅：数据驱动业务，业务间解耦
大数据视角	数据整合：OLTP到OLAP的数据整合
	数据治理：动态数据实时发布到治理系统

- 核心

- 实时获取数据
- 向外发布数据



# 产品形态

- 平台化


- WEB管理工具
- 资源管理和调度
- 报警监控

- 插件化

- 不同数据源extractor
- 不同目标库applier
- 账号系统

# 产品形态

NDC Dashboard



admin

数据迁移

数据订阅

订阅节点

执行节点

用户列表

迁移任务列表

Q 请输入任务名,回车搜索

启动

暂停

停止

释放

修改

详情

刷新

数据同步

数据迁移

<input type="checkbox"/>	任务名称	源类型	任务状态	创建时间	全量迁移	增量延迟
<input type="checkbox"/>	music_playlist141718	MySQL	运行中	2017/06/08 11:46:45	<div><div></div></div> 100%	0 ms
<input type="checkbox"/>	▼ music_purchase	DDB	运行中	2017/06/06 19:19:53	<div><div></div></div> 100%	0 ms
<input type="checkbox"/>	music_purchase.50-topic1	MySQL	运行中	2017/06/06 19:19:53	<div><div></div></div> 100%	0 ms
<input type="checkbox"/>	music_purchase.5-music_platform5	MySQL	运行中	2017/06/06 19:19:53	<div><div></div></div> 100%	0 ms
<input checked="" type="checkbox"/>	music_purchase.4-music_platform4	MySQL	运行中	2017/06/06 19:19:53	<div><div></div></div> 100%	0 ms
<input type="checkbox"/>	music_purchase.2-music_platform2	MySQL	运行中	2017/06/06 19:19:53	<div><div></div></div> 100%	0 ms
<input type="checkbox"/>	music_purchase.15-music_platform7	MySQL	运行中	2017/06/06 19:19:53	<div><div></div></div> 100%	0 ms
<input type="checkbox"/>	music_purchase.1-music_platform1	MySQL	运行中	2017/06/06 19:19:53	<div><div></div></div> 100%	0 ms
<input type="checkbox"/>	▶ yidun-statistic	DDB	运行中	2017/06/05 17:07:52	<div><div></div></div> 100%	0 ms
<input type="checkbox"/>	▶ lofter-timeline-2	DDB	失败	2017/06/05 16:42:14	<div><div></div></div> 100%	14 s
<input type="checkbox"/>	▶ music_kudu_musician_1	DDB	运行中	2017/05/19 17:04:11	<div><div></div></div> 100%	0 ms
<input type="checkbox"/>	▶ music_kudu_store_1	DDB	运行中	2017/05/19 16:49:12	<div><div></div></div> 100%	0 ms

任务数据

全量迁移

增量迁移

执行节点地址

10.120.196.84:7101

数据源端地址

10.160.247.73:4331

任务ID

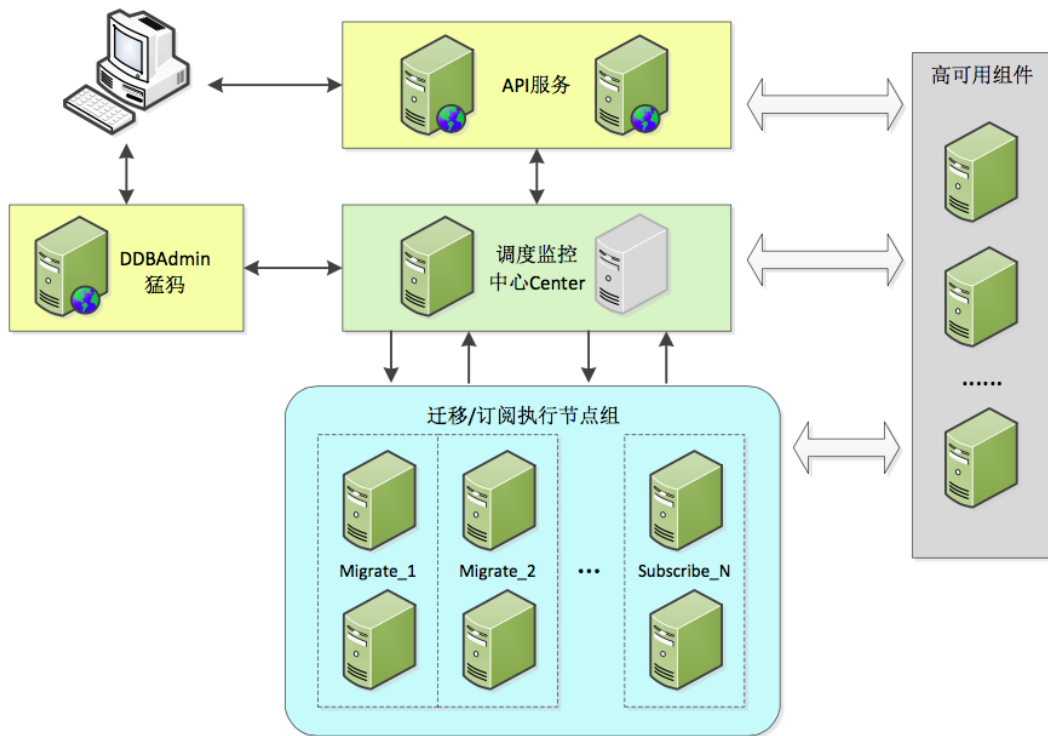
417

上一页

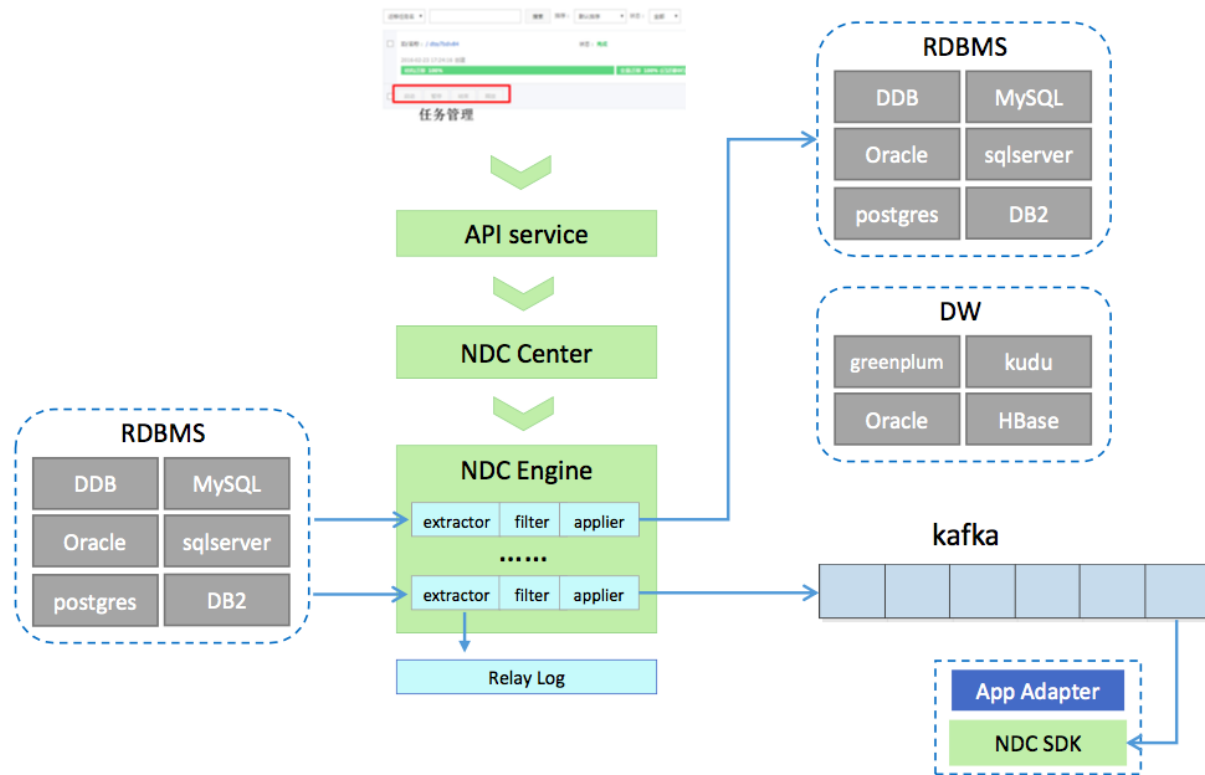
第1页 共3页

下一页

# 系统架构

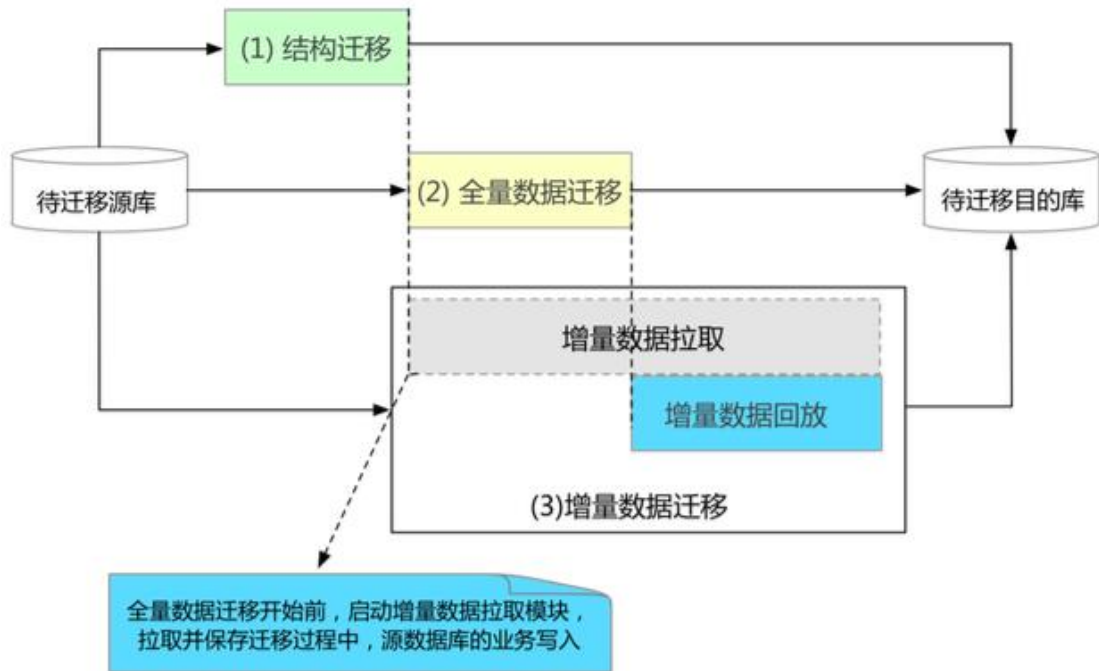


# 系统架构



# 核心实现

- 数据迁移
  - 结构迁移
  - 全量迁移
  - 增量迁移



源			目标				
MySQL	DDB	Oracle	MySQL	DDB	Oracle	Kudu	GreenPlum

# 核心实现

- 全量迁移
  - 特性
    - 不锁表，对线上压力小
    - Insert ignore into
    - 并行复制（目标端并行插入）
  - 原理
    - 计算当前源的最大主键max
    - 源端Select ... where pk <= max ... limit n order by pk，结果并行插入目标端
    - select 返回结果数 < n，全量数据拉取完成，等待apply完成

# 核心实现

- 增量迁移
  - 特性
    - 行级并行复制
    - 数据一致性保证
    - 断点续传
  - 原理
    - Tungsten Replicator
    - Row binlog ( MySQL ), 物化视图 ( Oracle )
    - 唯一性索引构建事件依赖图, 幂等性apply
    - 定时更新同步点 ( binlog Pos , 物化视图的自增id )

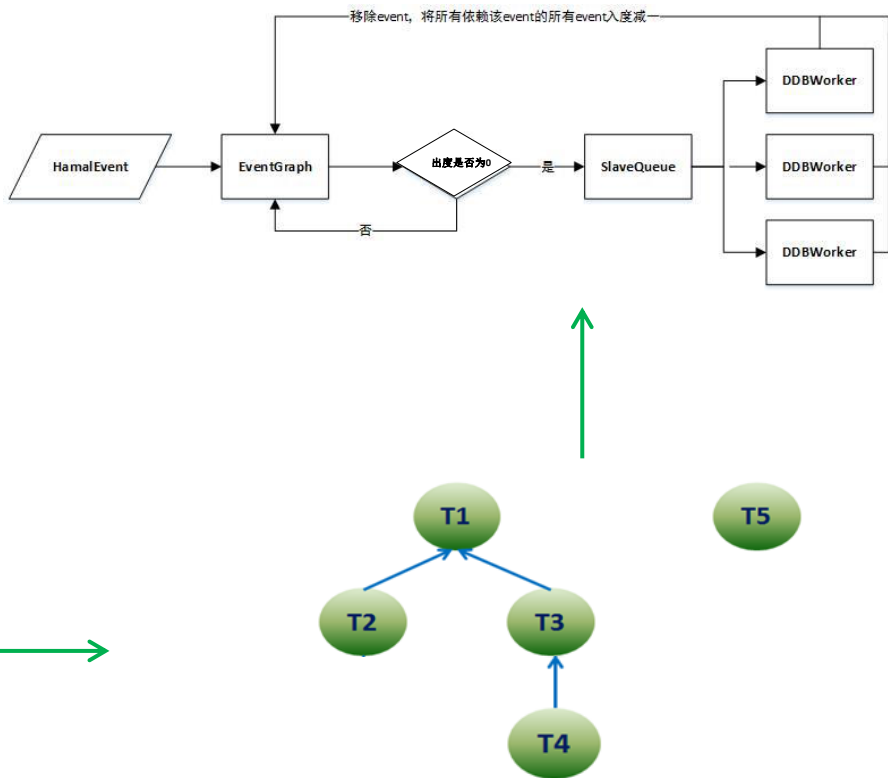
# 核心实现

## 增量迁移-并行复制

事务	修改id行为
T1	Update 1 to 3
T2	Insert 1
T3	Update 3 to 4
T4	Delete 4
T5	Insert 2



Hash值	事件依赖链
1	T1 T2
2	T5
3	T1 T3
4	T3 T4





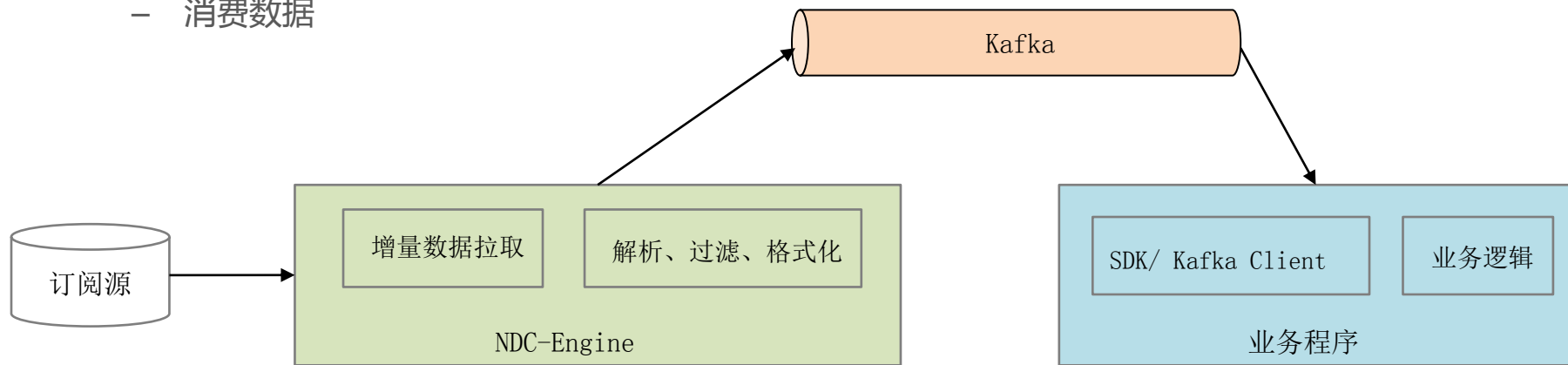
# 核心实现

- 增量迁移-并行复制
  - 事件依赖图
    - 唯一性索引（包括前项、后项）构建依赖图
    - 无依赖->并行，有依赖->串行
    - 每个表对应一个事件依赖图

# 核心实现

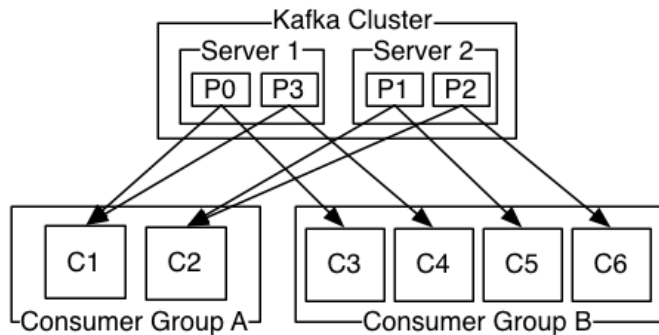
- 数据订阅

- 拉取数据 ( MySQL, DDB, Oracle )
- 发布数据 ( Kafka )
- 消费数据



# 核心实现

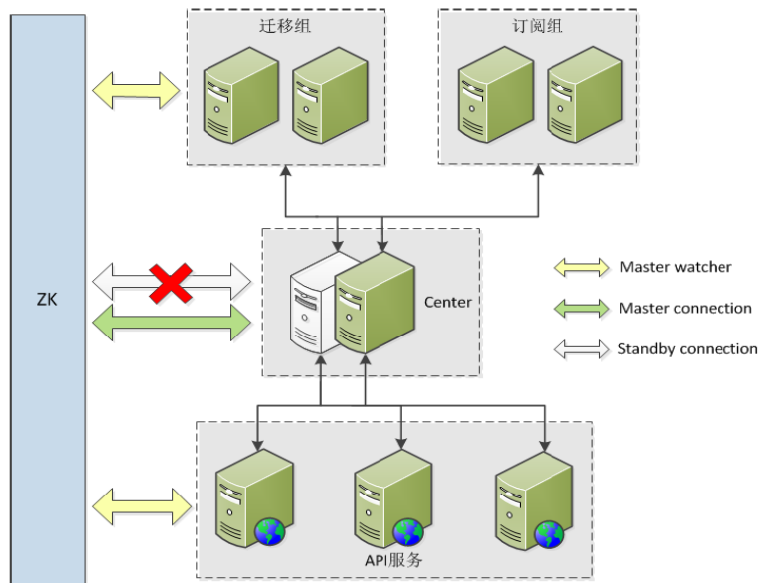
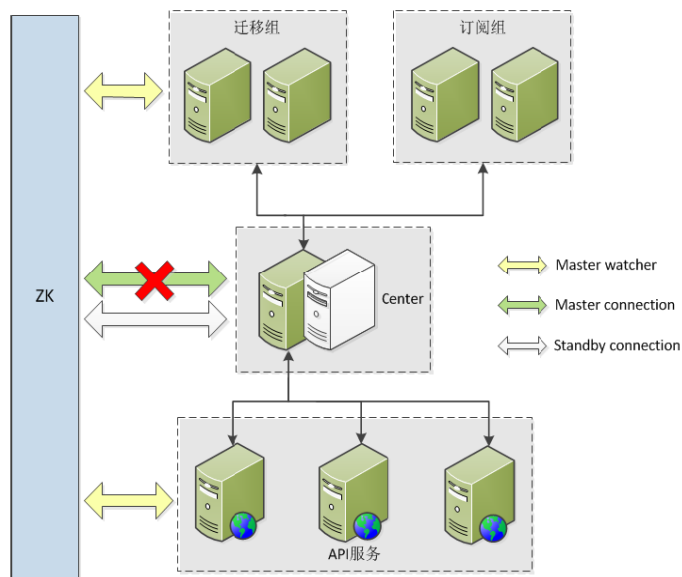
- 数据订阅-SDK
  - SubscribeEvent
  - 数据有序
  - 高可用
  - 消息可能重复消费



```
public class SubscribeEvent {  
    private List<OneRowChange> rowChanges;  
    private long seqno;  
    private boolean lastEvent;  
    private long timestamp;  
}  
  
public class OneRowChange {  
    private List<OneColumnChange> columnChanges;  
    private String schemaName;  
    private String tableName;  
    private RowChangeType type;  
}  
  
public class OneColumnChange {  
    private String columnName;  
    private ColumnType type;  
    private Object oldValue;  
    private Object newValue;  
}
```

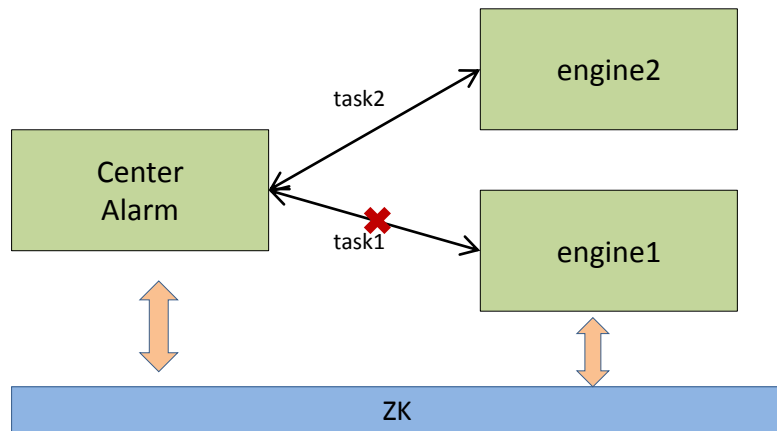
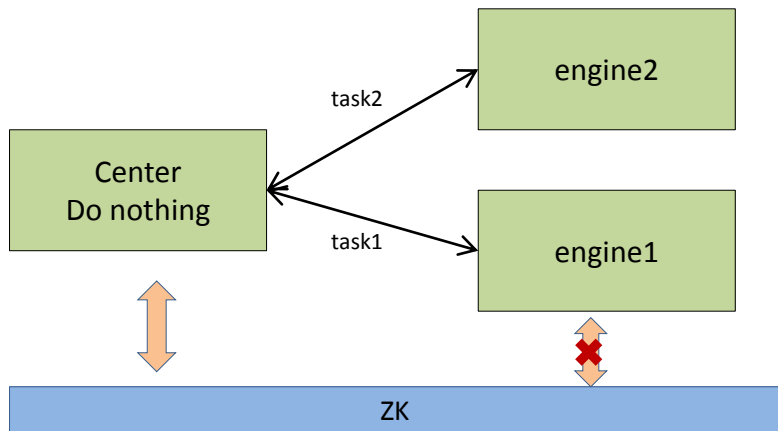
# 核心实现

- Center高可用
  - Zookeeper + Cruator



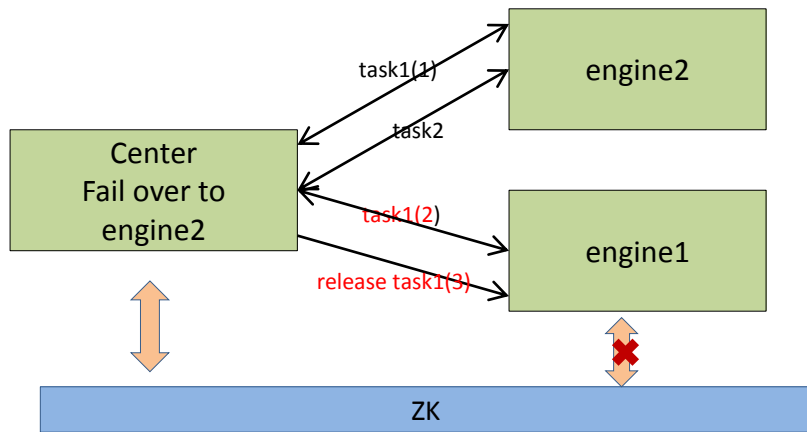
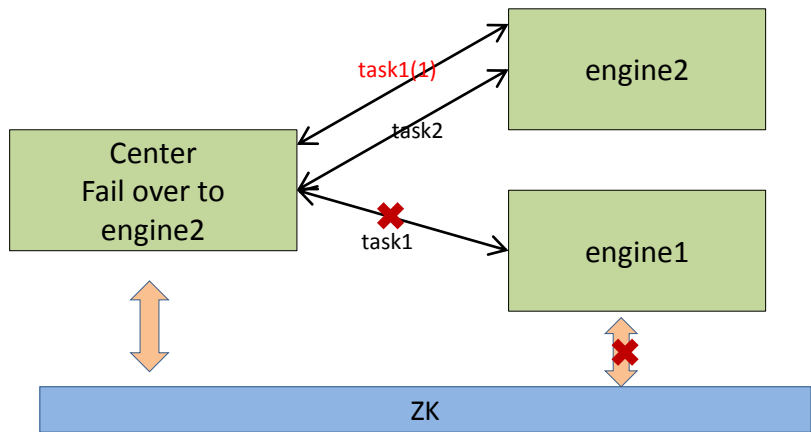
# 核心实现

- Engine高可用
  - 租约与ZK双重验证



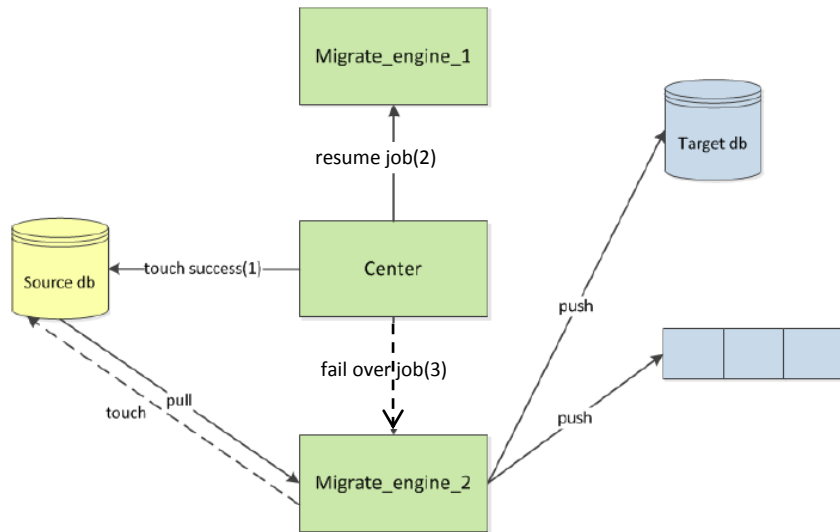
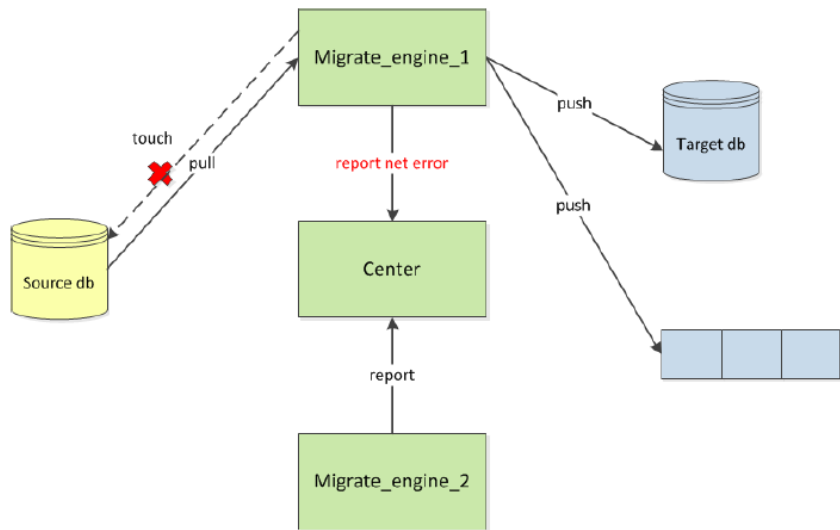
# 核心实现

- Engine高可用
  - 租约与ZK双重验证



# 核心实现

- 任务高可用
  - Center探活源端成功
  - 任务断点续传



網易 NETEASE

Thank you