Fault Detection and Diagnosis System

大型科学实验运行故障诊断系统设计与开发进展

张航畅，顾旻皓

中国科学院高能物理研究所

Outline

- 系统需求
- 系统设计与架构
  - 信息收集模块
  - 故障分析模块
- 应用场景
- 总结与展望

背景

LHAASO 高海拔宇宙线观测站

- 5195路电磁粒子探测器（ED）
- 1171路缪子探测器（MD）
- 3000路光电倍增管（PMT）
- 12台望远镜

JUNO 江门中微子观测站

- 数万个光电倍增管
- 160+台交换机和计算集群

大型科学实验：

- 规模庞大
- 设计复杂
- 无人值守
- 不停机运行

为什么需要FDD?

实验运行需求

- 收集在线系统运行状态与故障记录
- 快速检测实验硬件/软件状态
- 快速分析故障原因，通知值班人员

实验管理需求

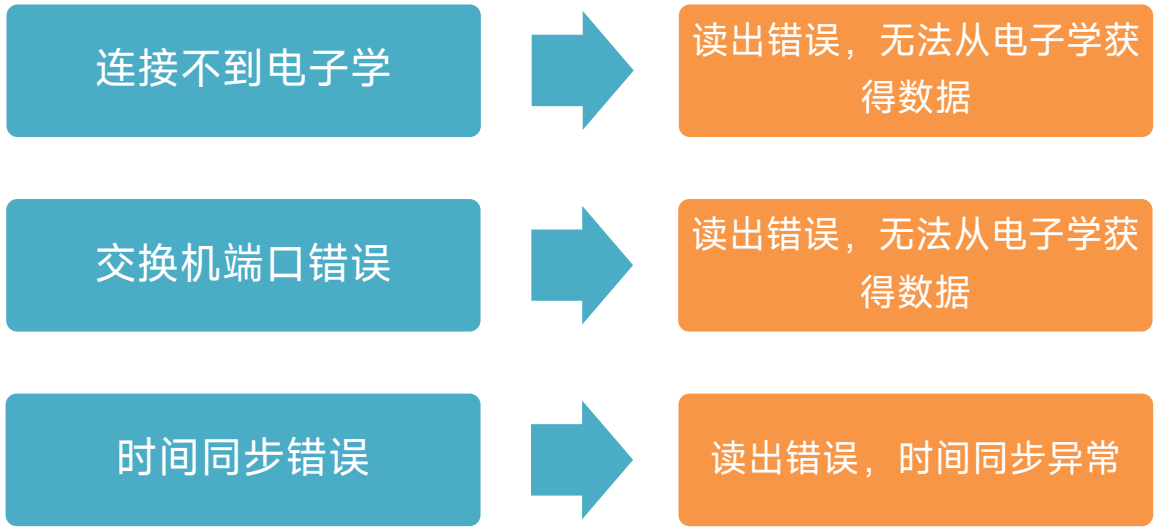
- 故障统计与回溯
- 运行与维护记录检索
- 生成报告

FDD的原理：实验系统中的故障因果关系

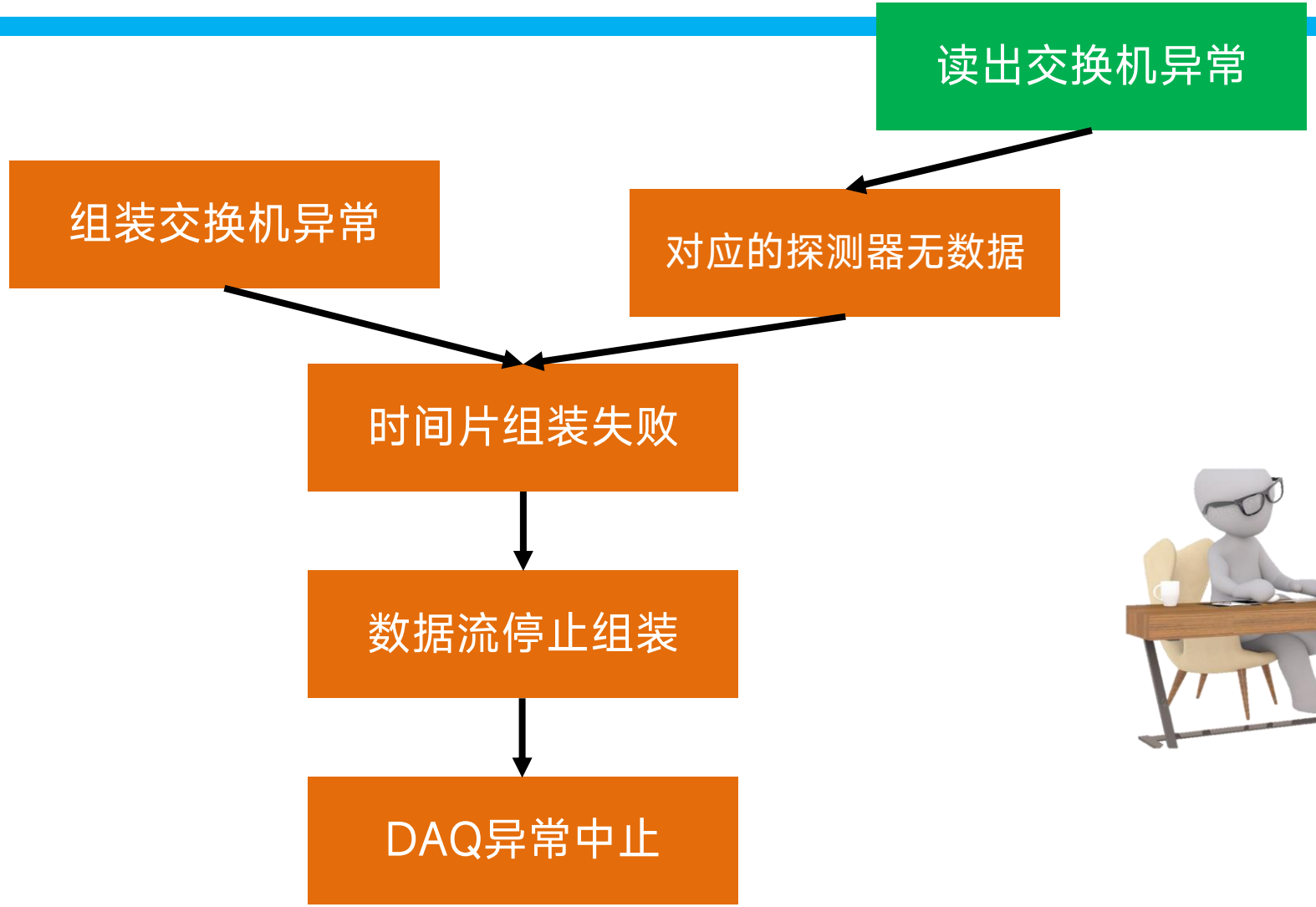
导致在线系统的故障因素：

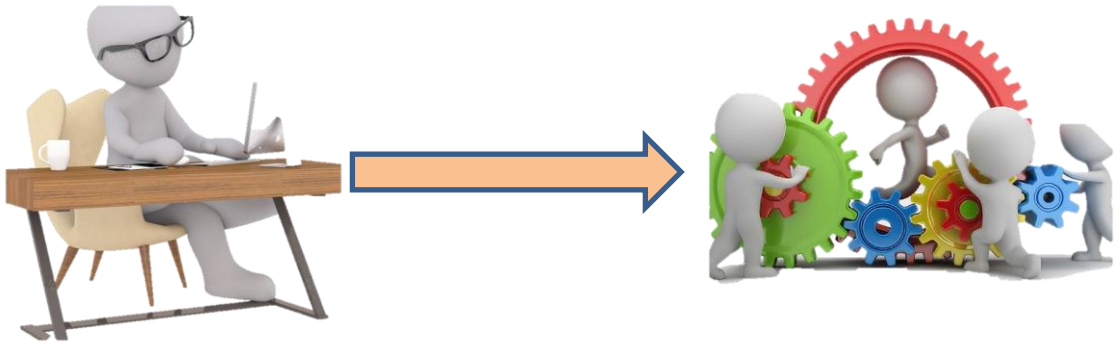
- 电子学连接错误;
- 电子学读出错误;
- 电子学同步错误;
- WR/DAQ 交换机错误;
- WR交换机同步错误;
- 计算节点错误;
- 软件进程错误;
- ...

Example:

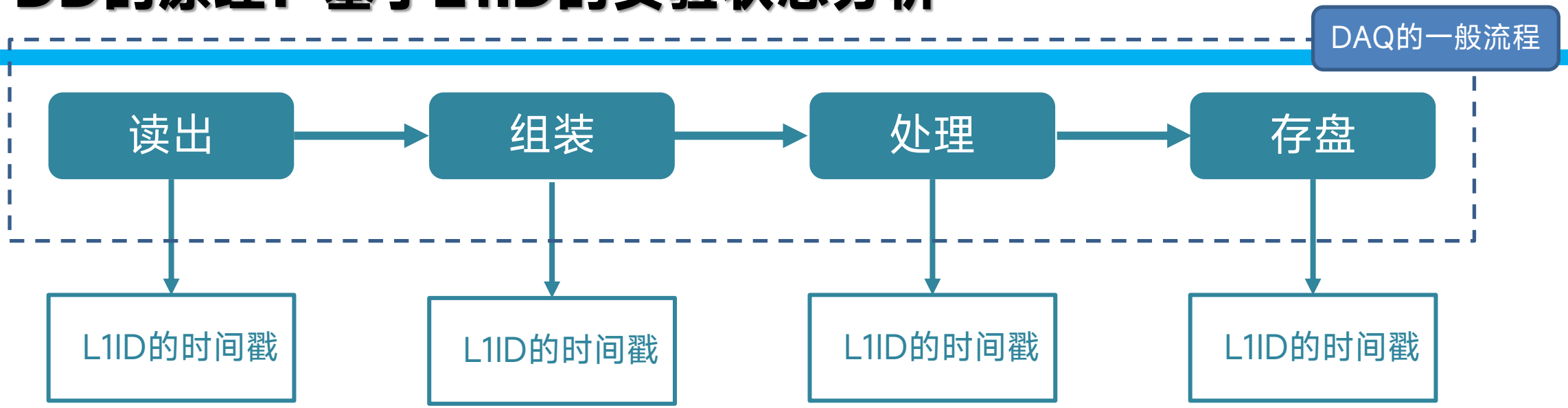


FDD的原理：故障因果树

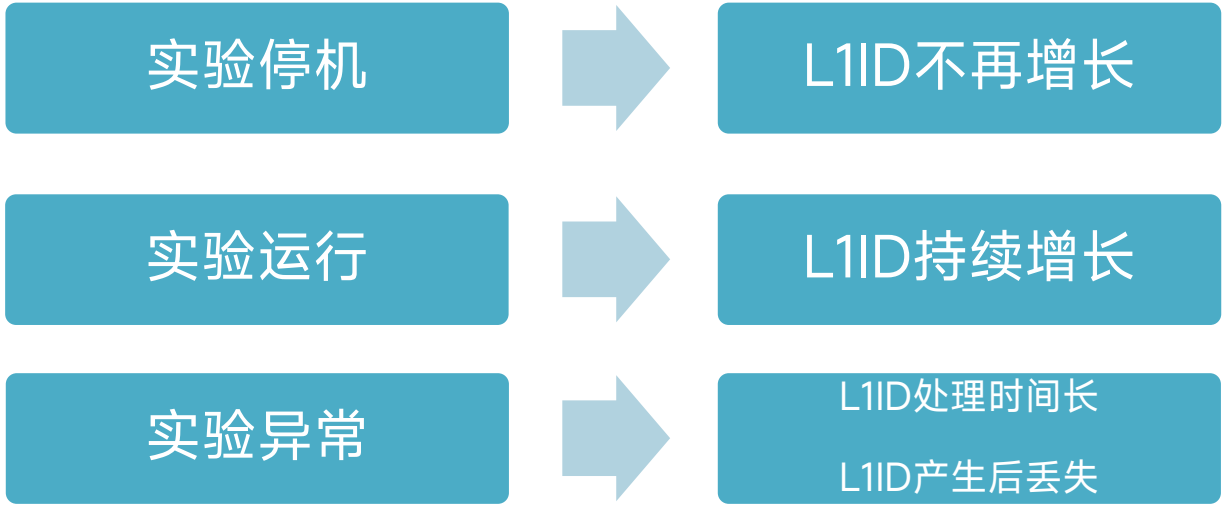


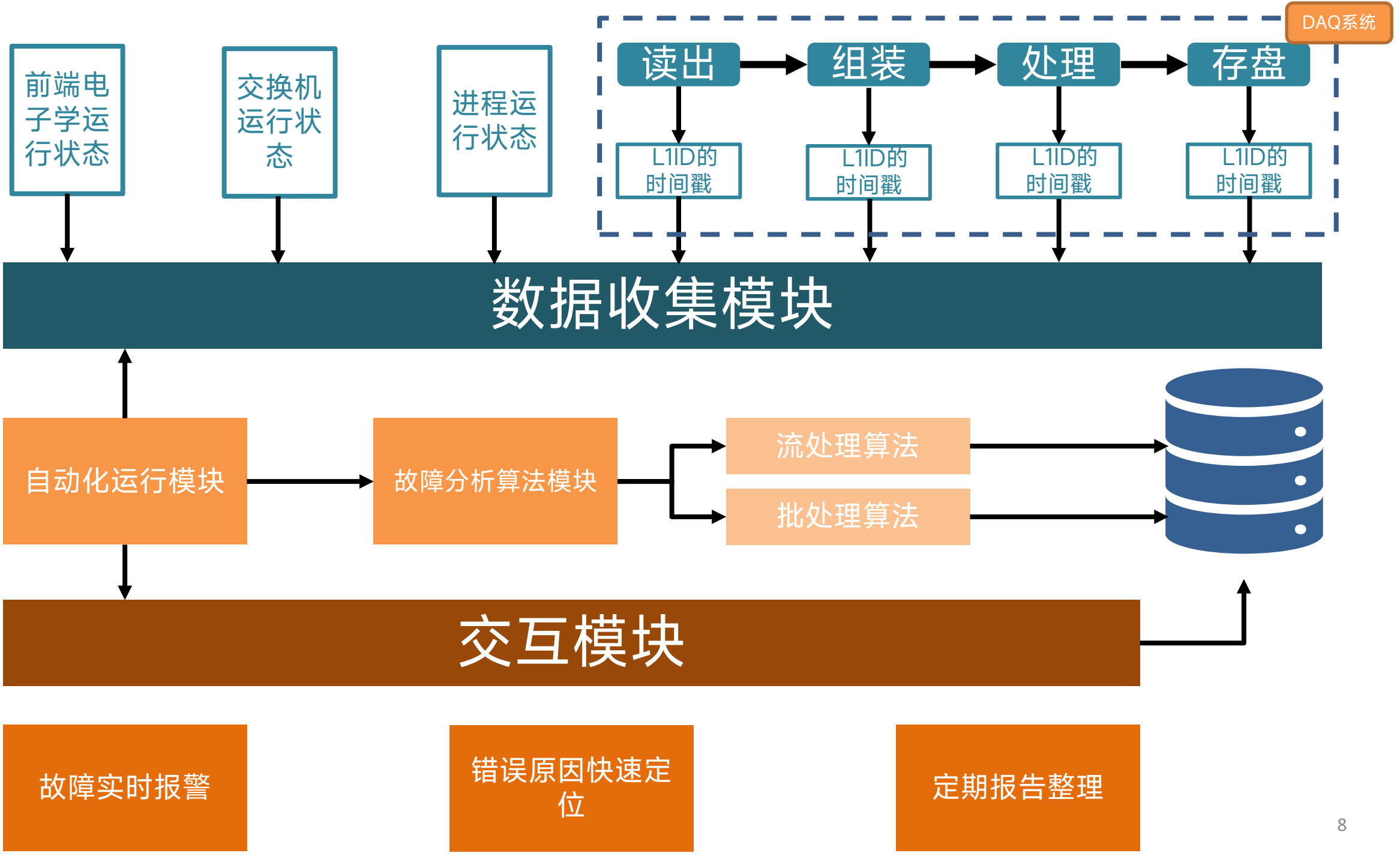


FDD的原理：基于L1ID的实验状态分析

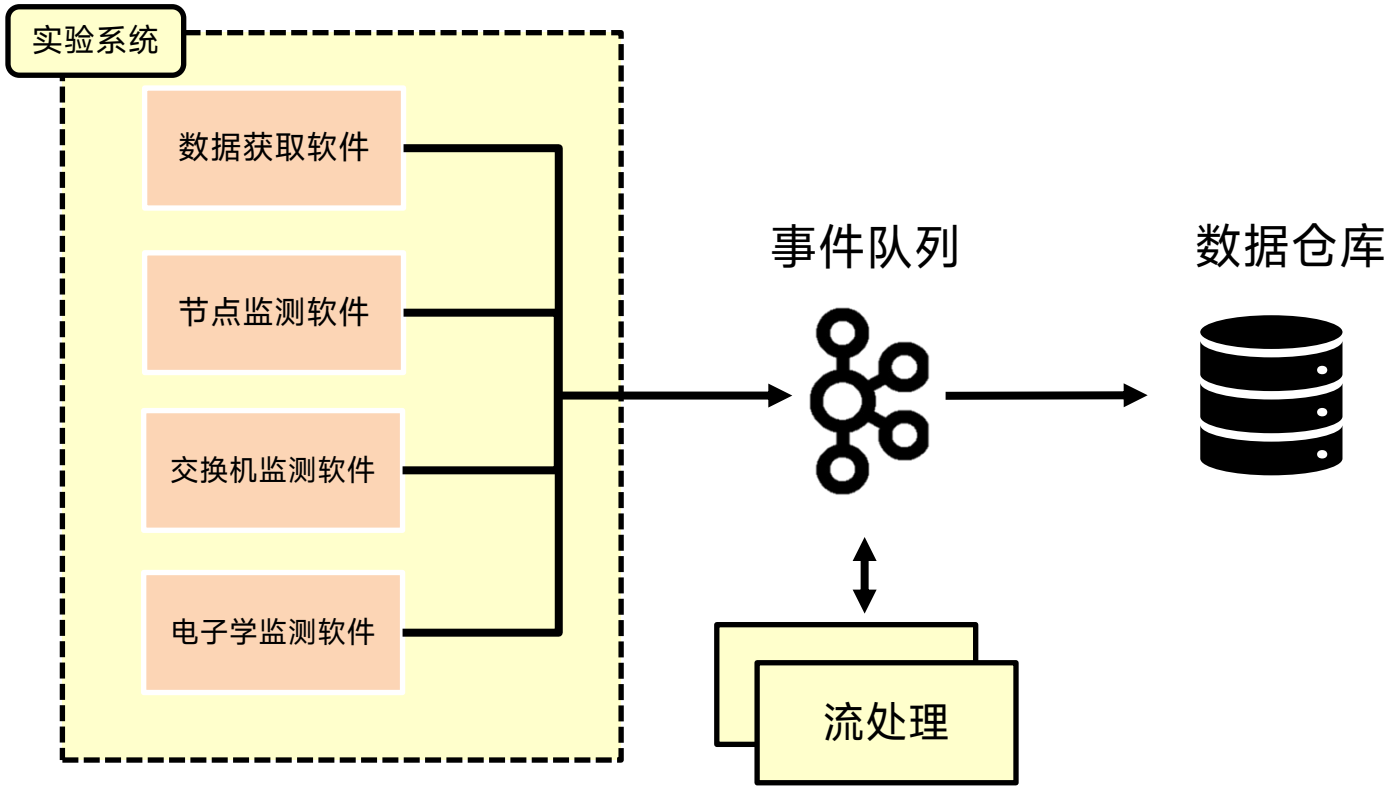


L1ID：DAQ系统为数据包产生的全局ID。




DAQ系统
前端电子学运行状态
交换机运行状态
进程运行状态
读出
组装
处理
存盘
L1ID的时间戳
L1ID的时间戳
L1ID的时间戳
L1ID的时间戳
数据收集模块
自动化运行模块
故障分析算法模块
流处理算法
批处理算法
交互模块
故障实时报警
错误原因快速定位
定期报告整理

系统架构：数据收集模块



技术栈:

- 消息中间件:
  Redpanda
- 数据库:
  Cassandra
- 流处理:
  Kafka Streams/Java

FDD系统架构: 三级消息结构

Monitoring Data

- Dataflow
- Node
- Network

- 原始消息流 100K/s：使用三级结构发送和保存
- 每条消息使用不同的 Topic 来区分不同的消息来源，进行分类处理。

Level 1


```
{
  "topic": "l1_df_ts",
  "key": "df|ros.tf-out.ros-4302.l1id",
  "value": "1228909090",
  "timestamp": 1675589248602,
  "partition": 0,
  "offset": 1807186207
}
```


L1id Msg Example

- Topic
- Key(event)
- Timestamp
- Data

Msg Format

- 聚合前的数据流信息

Stream

Save

Level 2


```
{
  "topic": "l1id-aggragate3-store1-changelog",
  "key": "df|ros.tf-gen.ros-3101.l1id\u0000\u0000\u0001\ufffd \ufffd(
    \u0000\u0000\u0000\u0000\u0000",
  "value": ",1228648704,1228648705,1228648706,1228648707,1228648708,1228648709
    ,1228648710,1228648711,1228648712,1228648713,1228648714,1228648715,1228648716
    ,1228648717,1228648718,1228648719,1228648720,1228648721,1228648722,1228648723
    ,1228648724,1228648725,1228648726,1228648727,1228648728,1228648729,1228648730
    ,1228648731,1228648732,1228648733,1228648734,1228648735,1228648736,1228648737
    ,1228648738,1228648739,1228648740,1228648741,1228648742,1228648743,1228648744
    ,1228648745,1228648746,1228648747,1228648748,1228648749,1228648750,1228648751
    ,1228648752,1228648753,1228648754,1228648755,1228648756,1228648757,1228648758
    ,1228648759,1228648760,1228648761,1228648762,1228648763,1228648764,1228648765
    ,1228648766,1228648767,1228648768,1228648769,1228648770,1228648771,1228648772
    ,1228648773,1228648774,1228648775,1228648776,1228648777,1228648778,1228648779
    ,1228648780,1228648781,1228648782,1228648783,1228648784,1228648785,1228648786
    ,1228648787,1228648788,1228648789,1228648790,1228648791,1228648792,1228648793
    ,1228648794,1228648795,1228648796,1228648797,1228648798,1228648799,1228648800
    ,1228648801,1228648802,1228648803",
  "timestamp": 1675586644996,
  "partition": 0,
  "offset": 14155761
}
```


Aggregated l1id Msg

- 聚合后的数据流信息
- 节点信息
- 网络信息

Batch/Stream

Save

Level 3

ev	tk	t	data
km2a|df-l1id-time|es-ack.error	2023-02-03	2023-02-03 07:11:41.000000+0000	0
km2a|df-l1id-time|es-ack.error	2023-02-03	2023-02-03 07:11:42.000000+0000	-2
km2a|df-l1id-time|es-ack.error	2023-02-03	2023-02-03 07:11:52.000000+0000	-2
km2a|df-l1id-time|es-ack.error	2023-02-03	2023-02-03 07:11:53.000000+0000	0
km2a|df-l1id-time|es-ack.error	2023-02-03	2023-02-03 08:45:54.000000+0000	0
km2a|df-l1id-time|es-ack.error	2023-02-03	2023-02-03 08:45:55.000000+0000	-2
km2a|df-l1id-time|es-ack.error	2023-02-03	2023-02-03 08:45:58.000000+0000	-2
km2a|df-l1id-time|es-ack.error	2023-02-03	2023-02-03 08:45:59.000000+0000	0
km2a|df-l1id-time|es-ack.error	2023-02-03	2023-02-03 08:46:00.000000+0000	-2
km2a|df-l1id-time|es-ack.error	2023-02-03	2023-02-03 08:46:20.000000+0000	-2
km2a|df-l1id-time|es-ack.error	2023-02-03	2023-02-03 08:46:21.000000+0000	0
km2a|df-l1id-time|es-ack.error	2023-02-03	2023-02-03 09:30:08.000000+0000	0
km2a|df-l1id-time|es-ack.error	2023-02-03	2023-02-03 09:30:09.000000+0000	-2
km2a|df-l1id-time|es-ack.error	2023-02-03	2023-02-03 09:30:51.000000+0000	-2

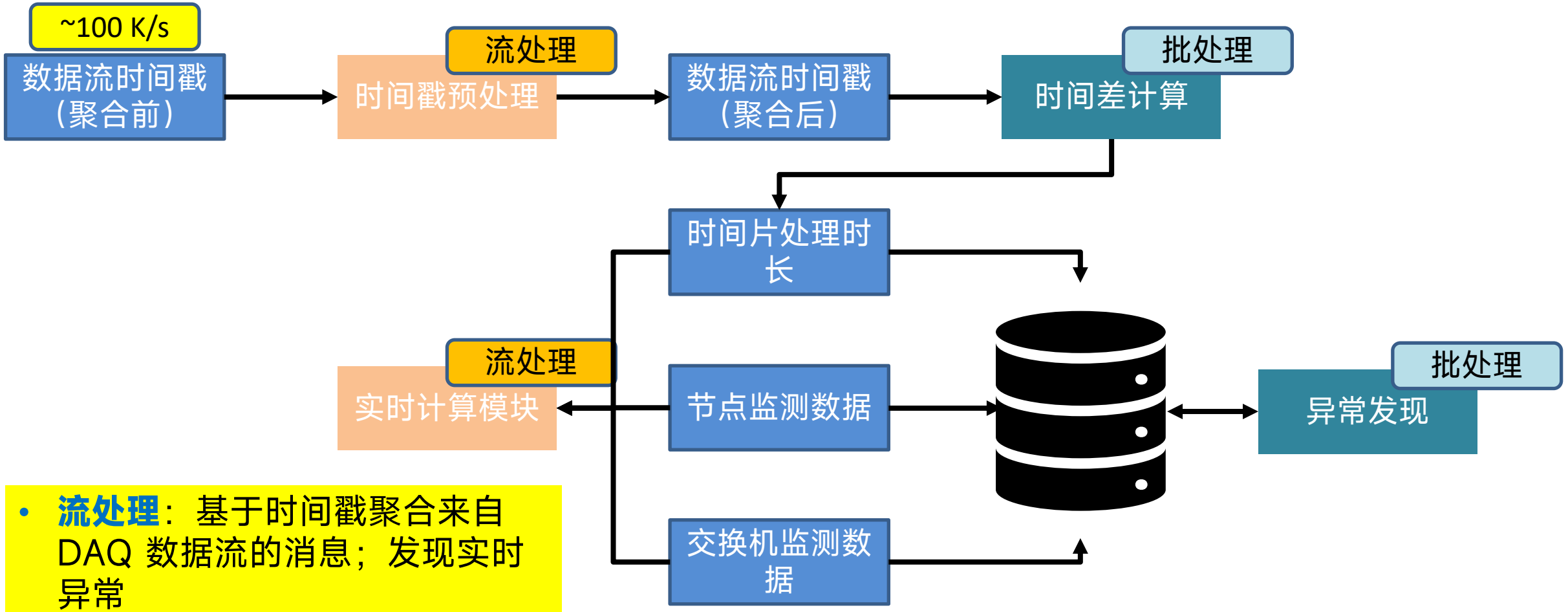
Data saved to the database

- 处理后的疑似故障信息

收集模块总结

- FDD 的消息系统基于Redpanda，基于C++的消息中间件。 Redpanda
- 为了满足原始消息率100K/s的要求，设计了一个三级消息结构，根据不同的来源对消息进行分类处理。
- 使用流处理（Kafka Streams/JAVA）来聚合数据。
- （目标）实验系统的几乎所有运行状态数据都会被采集并保存到数据库中。

FDD系统架构：故障分析模块

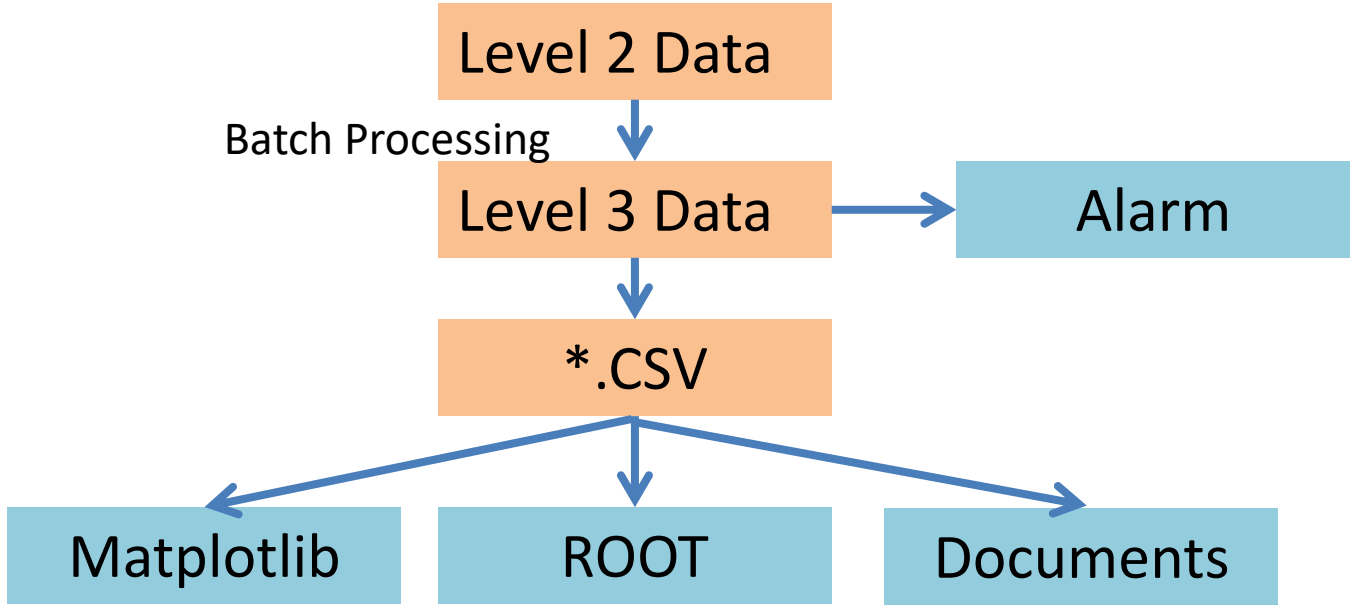


- **流处理**：基于时间戳聚合来自 DAQ 数据流的消息；发现实时异常
- **批处理**：从数据库发现异常，保存至另一个表中

故障分析算法

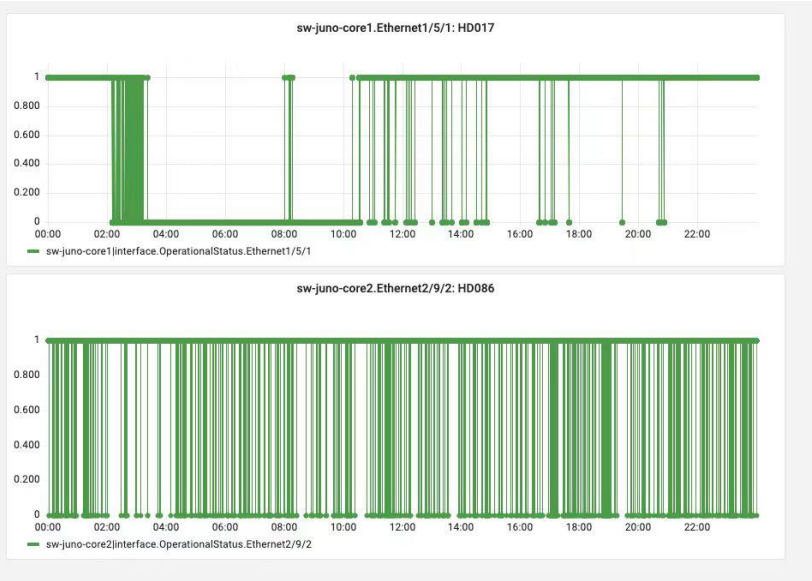
- process_l1id_rates: 利用L2级的消息计算每个L1ID处理时间
- error_detect_based_on_threshold: 利用阈值判断是否故障
- fault_cause_based_on_switch_time: 当发生DAQ故障时判断是否是哪台交换机故障

```
config = {
    "km2a|df-l1id-time|es-ack": {
        "upper": 15000,
        "lower": 0
    },
    "km2a|df-l1id-time|ps-done": {
        "upper": 8000,
        "lower": 0
    },
    ".*NodeStatus.NetworkPortStatus.*": {
        "upper": 1,
        "lower": 0.5
    },
    "sw-km2A|interface.OperationalStatus.*":{
        "upper": 1,
        "lower": 0.5
    },
    "df-l1id-time|ros-all-ready": {
        "upper": 20000,
        "lower": 10
    }
}
```

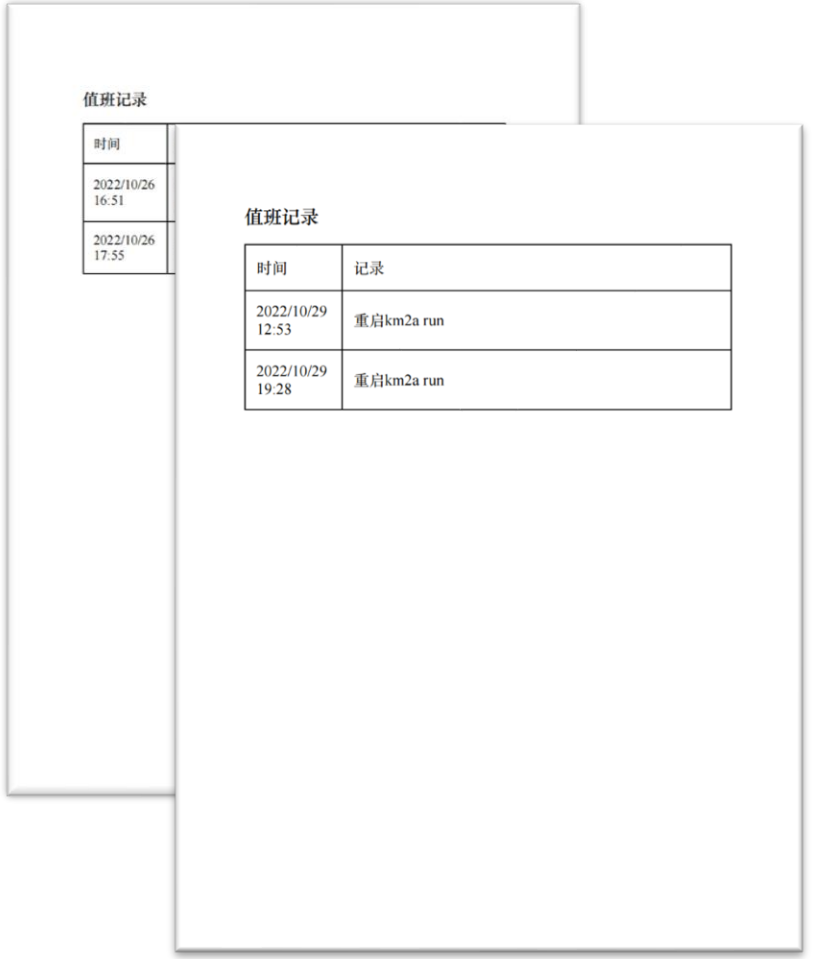


应用场景（1）

交换机故障统计



生成值班记录



值班记录

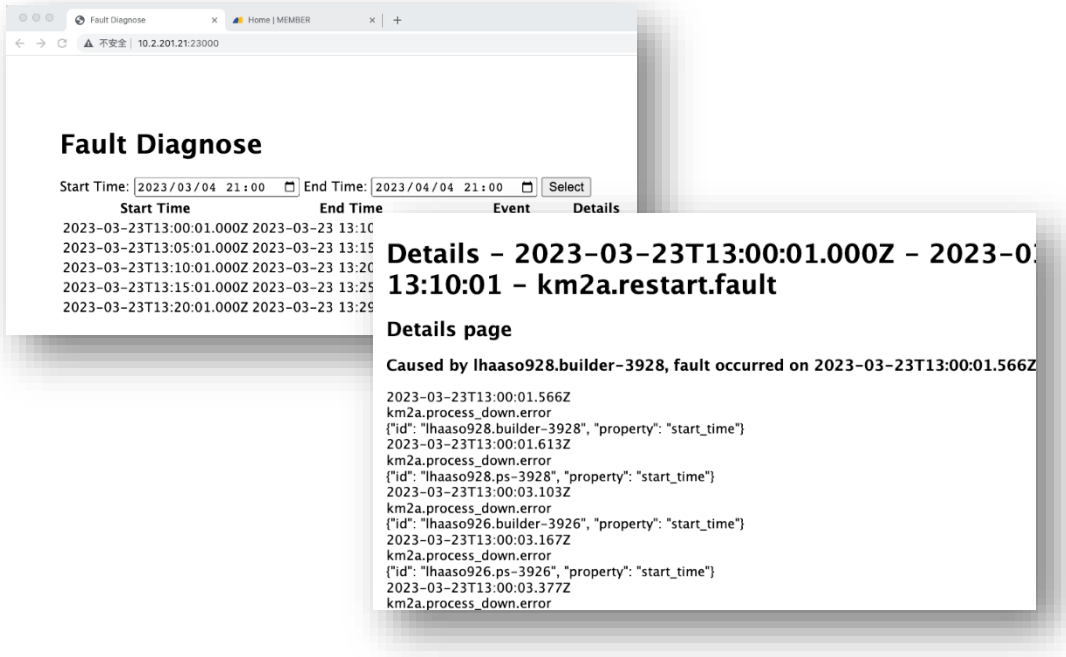
时间
2022/10/26 16:51
2022/10/26 17:55

值班记录

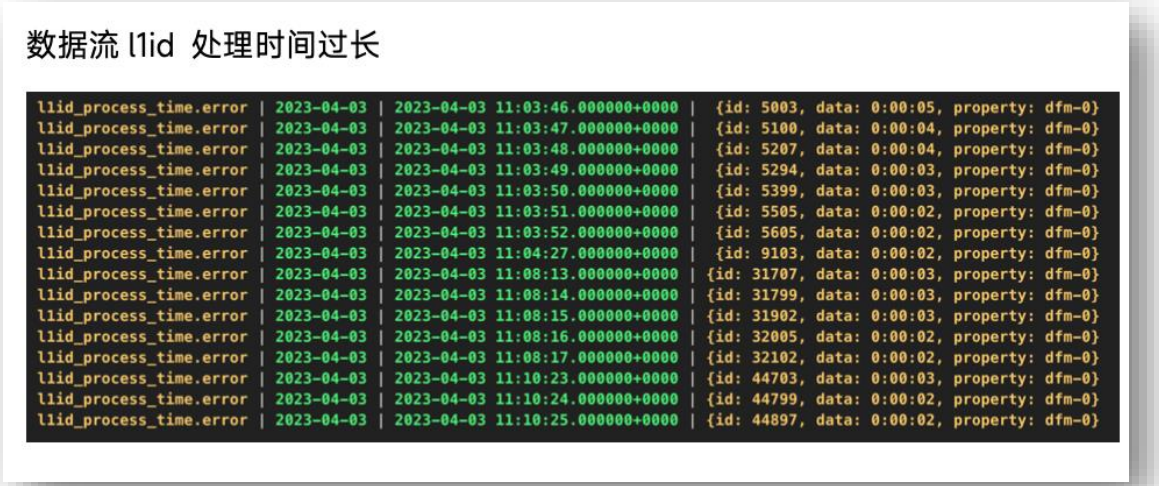
时间	记录
2022/10/29 12:53	重启km2a run
2022/10/29 19:28	重启km2a run

应用场景（2）

接入值班页面，直接给出故障原因



数据流质量测试



总结与展望

- 已完成结构设计和方案验证
- 完成了三级消息结构、流处理、批处理框架
- 原型系统已在LHAASO部署，开始部分信息收集和对计算节点、DAQ读出交换机、DAQ软件的故障分析
- 为JUNO DAQ开发阶段定制了一个版本，用于DAQ数据流质量分析

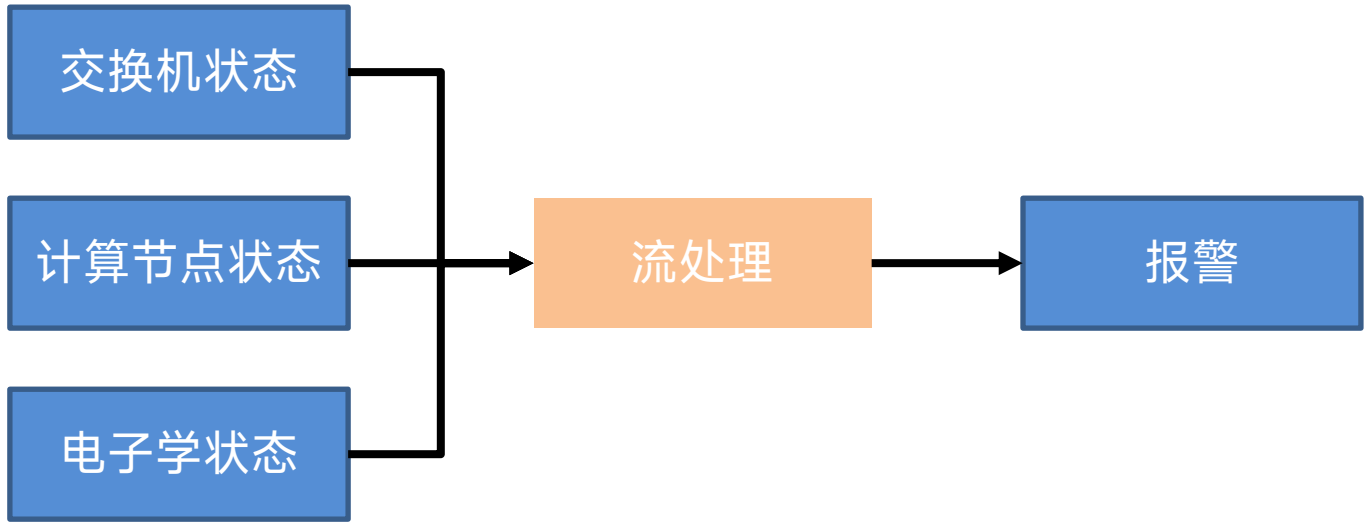
Next:

- 开发自动化运行模块
- 设计更全面的故障因果树
- 提供用户友好的前端页面
- 收集更多实验系统的运行信息
- 提供更丰富的故障分析，为实验运行服务

Thanks

自动化运行模块

- 流处理实时报警



- 批处理自动分析

Backup: Kafka Vs Redpanda

Workload	Target P99.9 Latency	Kafka Infra Requirement		Redpanda Infra Requirement	
		Nodes	P99.9 Latency	Nodes	P99.9 Latency
50 MB/sec	< 20ms	3 (i3en.xlarge)	16.123ms	3 (is4gen.medium)	6.363ms
500 MB/sec	< 20ms	9 (i3en.3xlarge	73.61ms	3 (i3en.3xlarge)	10.571ms
1 GB/sec	< 20ms	9 (i3en.9xlarge)	271.47ms	3 (i3en.6xlarge)	16.216ms

P99.9: 服务响应时间与分布指标,99.9%的用户耗时

Backup：Cassandra Throughput