监控系统项目开发文档

> 决策源于数据，而数据源于采集，采集源于规则梳理，规则梳理取决于业务的理解和架构经验积累。

项目背景：

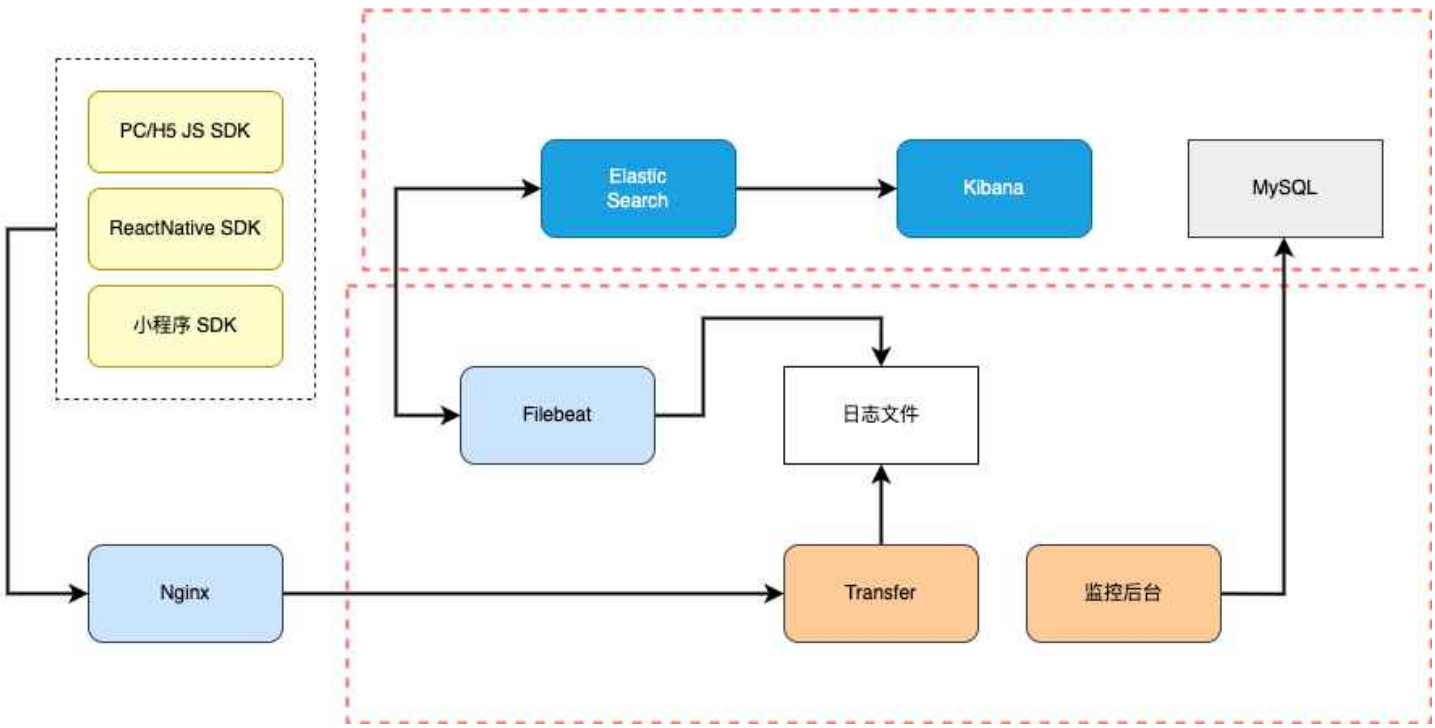
效率和质量是所有技术团队的核心基石。尽管效率容易量化和观察，然而质量管理往往被忽视。特别是在业务链路复杂、产品种类丰富的公司，例如拥有APP、小程序、大量中后台系统和前台电商网站的企业。产品上线后，多端全链路用户行为数据和线上页面异常数据存在许多盲区，如用户使用情况、线上异常情况、设备分布情况以及营销活动的PV/UV转化情况等。

作为前端工程师，我们需要对前端监控系统架构有一个清晰的认知。具体而言，如何设计功能、架构系统、编排代码，以及哪些功能该做、哪些不该做，以及不同选择的难点和成本考量，都是需要深思熟虑的。通过构建全面的前端监控系统，我们可以更好地掌握产品质量，填补数据盲区，提高整体用户体验。

系统架构

复杂度和难度是前端工程师成长为技术专家的必要条件。监控系统项目正好符合这一条件。例如，一个兼容多端的监控系统在生产环境中的架构就体现了这种复杂性和挑战性。通过设计和实现这样的系统，前端工程师可以有效提升自己的技术水平，迈向专家之路。

这个架构的实现成本相当高，体现在开发成本、服务器成本、运维人力成本等方面，在这门课程中，我们把最核心的部分，抽离出来用下面的架构来实施开发，大家在业务中落地可以从这个架构中往外做扩展。



项目介绍

实现一个前端错误监控系统，做到在搜集前端错误进行简单处理后上报到后端 (Egg.js)，后端再次进行解析以后保存到 Elasticsearch 中，后端对搜集到的错误数据进行抽象分类归纳并将归纳后的数据保存到 MySQL 中便于查看和处理。

整个系统包含 4 大块功能：

- JS SDK: 实现 PC/H5 JS 版本 SDK，进行前端错误搜集以及上报
- 日志处理：Egg.js 搭建日志处理和展现系统
- 日志保存：使用Elasticsearch 进行日志保存，其中 Filebeat 用于日志转发，Kibana 用于原始日志查看
    - 一般情况下 ES(Elasticsearch 简称) 是与 logstash/Fluentd 等日志过滤器加上 kafka/RabbitMQ/Redis 一类的消息队列（削峰）结合起来使用的，这里第一次实施直接使用 Filebeat 与 ES 进行连接，降低部署难度。
- 错误数据保存: 使用MySQL 对同类型错误数据进行保存，从而进行后续处理，包含反查和错误统计等，我们称之为 **issue**

里程碑 1：

开发环境准备和数据结构设计，这里除了 Node 环境配置以外还需要以下配置:

环境配置	描述	验收标准
Node 环境	v12.18.2	本地安装能运行
MySQL	v8.0.18	本地安装能运行
Nginx	v1.17.7	本地安装能运行
Docker 配置	v2.3.0.3	本地安装能运行
Elasticsearch	v7.8.0	本地安装能运行
Filebeat	v7.8.0	本地安装能运行
数据结构设计		
错误上报数据结构设计	比如设备表/应用表/访问日志表等	数据库的表结构描述文件
issue 管理的数据表设计	针对数据上报所整理出来的 issue	数据库的表结构描述文件

```
1   elk ls
2 elasticsearch filebeat
```

```
  elk ls elasticsearch
LICENSE.txt README.asciidoc config jdk.app logs plugins
NOTICE.txt bin data lib modules
  elk ls filebeat
LICENSE.txt README.md filebeat fileb eat.yml module
NOTICE.txt fields.yml filebeat.reference.yml kiban a modules.d
  elk node -v
v12.18.2
  elk mysql -V
mysql Ver 8.0.18 for osx10.14 on x86_64 (Homebrew)
  elk nginx -v
nginx version: nginx/1.17.7
  elk docker -v
Docker version 19.03.8, build afacb8b
  elk java --version
java 14.0.1 2020-04-14
Java(TM) SE Runtime Environment (build 14.0.1+7)
Java HotSpot(TM) 64-Bit Server VM (build 14.0.1+7, mixed mode, sharing)
```

里程碑 2：

1. Transfer Service 实现

即转发服务实现，我们将转发服务进行简化，与监控后台合并为一个服务，在项目中对两者职能进行切 割，而不是直接分为两个 Node 服务，实际生产过程中提倡将服务分开，同时为了上报性能考虑转发服务 可以做成多节点。

2. 配置 Elasticsearch 数据字段的存储规则

为了避免上报字段过多造成 ES 字段过多从而占用服务器过多内存，数据在存入 ES 前除了必要的解析处 理外，还需要字段过滤，这一功能为了灵活起见我们放在 ES 本身以及转发服务中去实现。

里程碑 3：

1. PC/H5 SDK 封装实现

根据里程碑 1 的上报数据结构设计，设计并实现 JS SDK 用于上报错误

里程碑 4：

1. Dashboard 管理后台
2. 监控后台/看板的实现

- 实现一个 egg-es 插件，用于处理 es 连接和查询 数据看板，
- 实现简单的数据查看功能
  - PV/UV/设备数

    - 错误信息展示
        - 实时错误信息展示
        - issue 展示和简单管理

里程碑 5：

- 线上环境配置
- 服务部署
- 配置域名