# 全链路监控项目

根据skywalking源码修改.

参考资料: skywalking、 opentracing标准、google dapper

存储结构使用dapper的结构， hbase

Rowkey: traceid, columnfamily: spans

一期功能

1. 系统调用链路
2. 系统指标监控统计: 访问量、响应时间、错误率。 好像未使用流式处理指标， 流仅用来落库了。 预研项目， 流式处理考虑延后

可规划指标 链路还原

项目架构

涉及到的关键技术点

**最核心的流程：监控数据的生成与采集**

监控数据的生成：java instrumentation java premain，java agent 插件增强、字节码增强技术(bytebuddy)。 代理模式，在被增强的接口调用前后生成监控数据。 此处未作改变

服务节点数据生成：修改为配置上传，本来为自动生成，不具有可读性。

监控数据传输： 原来为client推server， 修改为日志打印由行内日志filebeat采集推kafka

监控数据处理： 原来为server接收client端数据并缓存处理， 修改为spark streaming接收kafka数据解析预处理后存入hbase。

两类数据

Trace数据：用于生成各类指标统计及调用关系

Service数据：用于注册记录服务相关信息， 作为图节点展示信息。 仅在启动时发送。 原版skywalking是jvm插件增强，用来当前服务状态发送和心跳。

Opentracing

监控数据描述对象：

Span：

AbstractSpan

Entryspan，localSpan， exitSpan。

讲一下全流程：

插件定义->代理增强RPC调用接口->Trace信息的生成与传递 -> 数据采集->数据分析(接口调用统计，错误率，响应率； 链路还原， 单笔追踪等)