全链路监控那些事

黄杰@饿了么

大纲

- 介绍
- 整体架构
- 计算框架
- 存储方案
- Demo
- Q&A

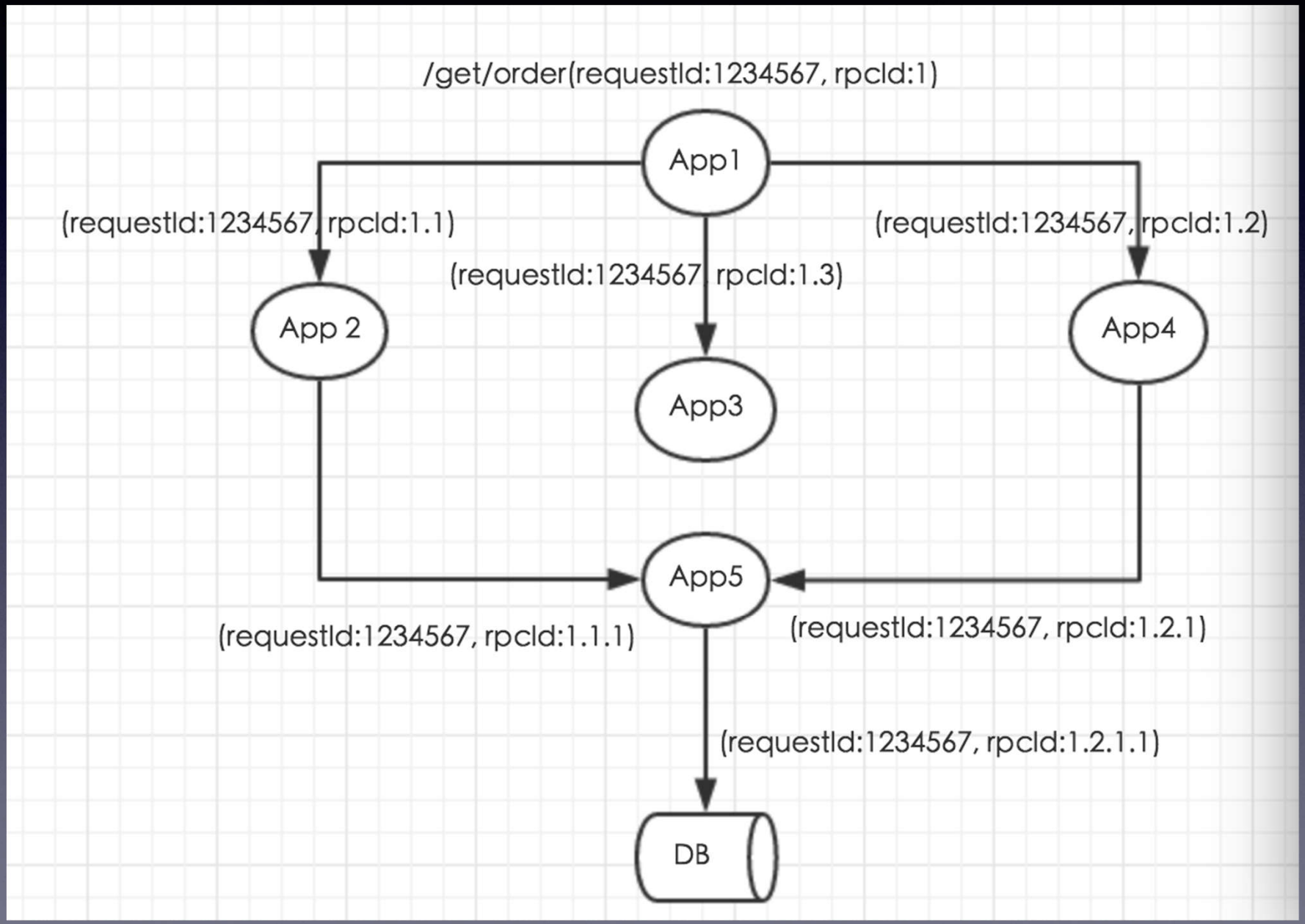
背景

- 从单应用到服务化之后，某个服务或几个服务出错时，不知道到底哪里出问题了；
  - 应用大量的报错很多时间可能都被忽略；
  - 有时应用出问题之后，需要花很长时间去定位问题；
- 服务之间的调用，数据库的访问等跨网络的IO操作，对我们都是黑盒子；
  - 访问好慢，但不知道是哪里导致访问慢了；
  - 对应具体某个应用内部也是黑盒的；

介绍

- 基于Java开发的轻量分布式实时监控平台，支持水平扩展，高吞吐；
- 由于饿了么开发语言的多样化，需要支持Java/Python/Go/Node.js；
- 支持跨IDC链路监控；
- 全量数据收集及处理；
- 由于是一个监控系统，所以整个系统不是一个高可靠的系统;

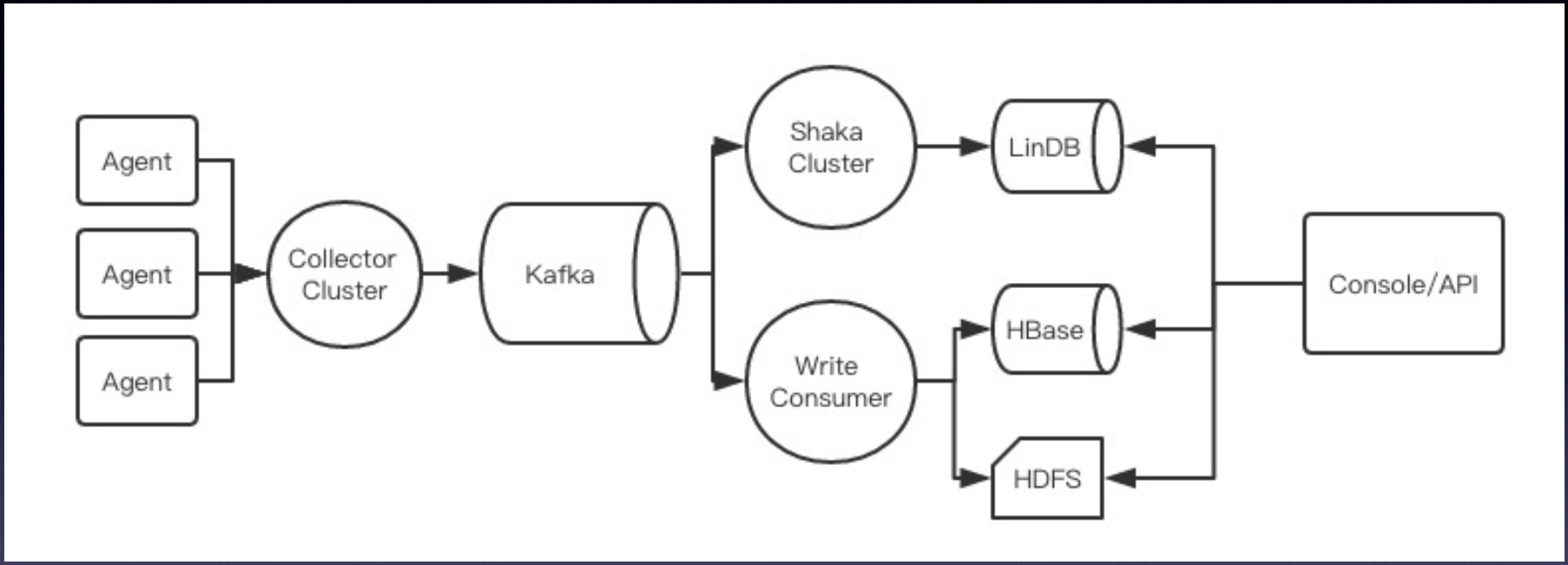
介绍


/get/order(requestId:1234567, rpcId:1)
App1
(requestId:1234567, rpcId:1.1)
(requestId:1234567, rpcId:1.3)
(requestId:1234567, rpcId:1.2)
App 2
App3
App4
App5
(requestId:1234567, rpcId:1.1.1)
(requestId:1234567, rpcId:1.2.1)
(requestId:1234567, rpcId:1.2.1.1)
DB

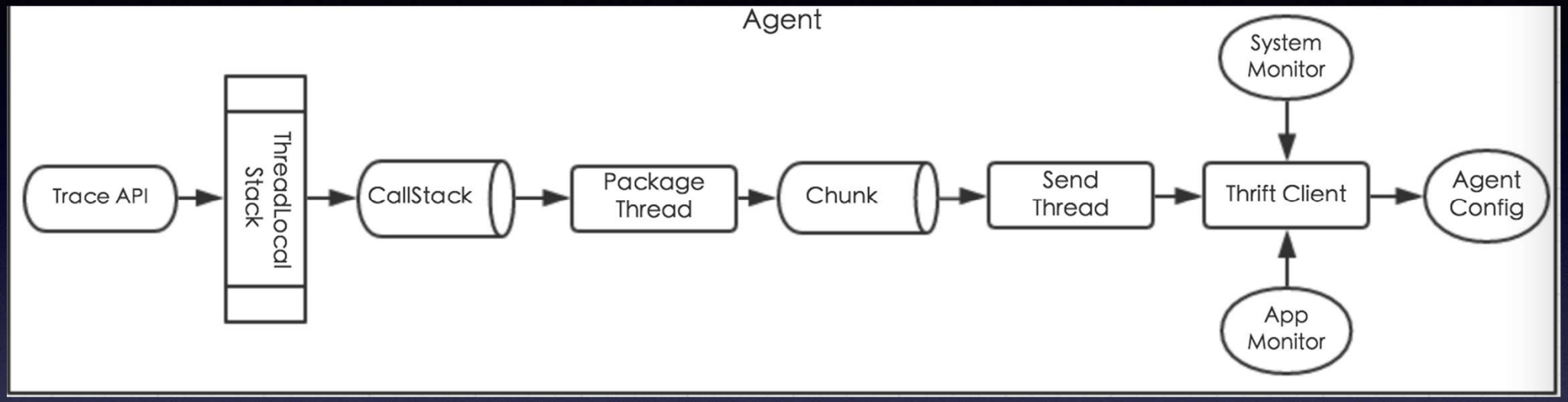
介绍

- 每天增量>200T，压缩之后>60T；
- 接入应用>1000；
- 接入服务器>12000台，不包括Docker；
- 消息处理延时<1s；
- 峰值消息处理QPS>2000W;

整体架构

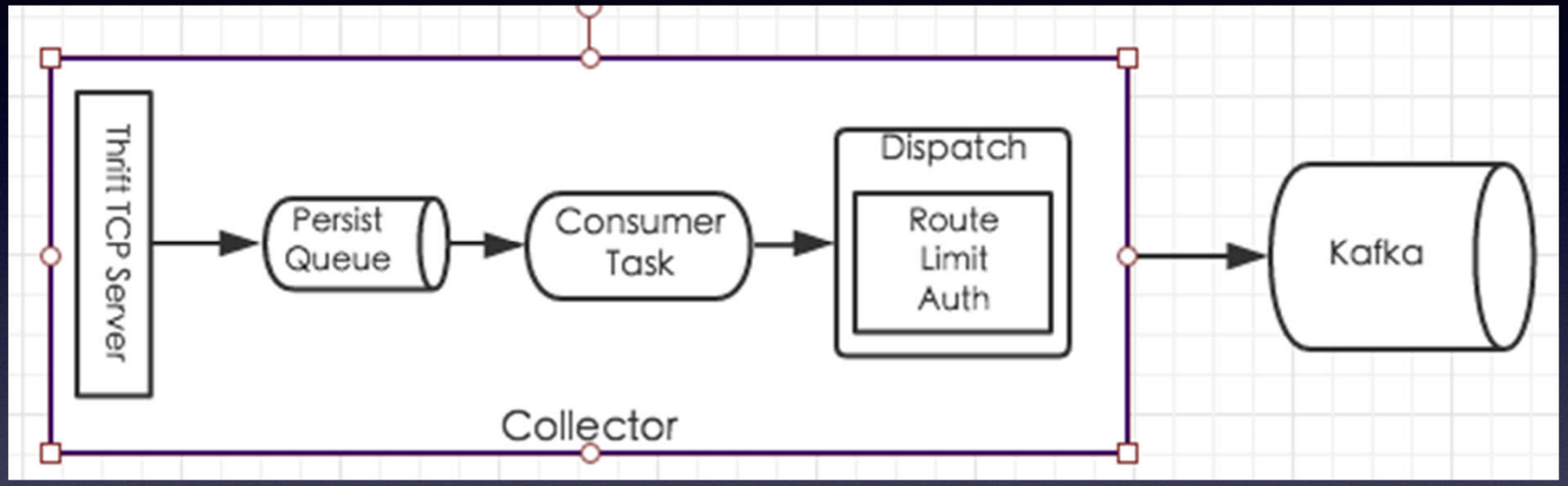

Agent
Agent
Agent
Collector Cluster
Kafka
Shaka Cluster
Write Consumer
LinDB
HBase
HDFS
Console/API

Agent



Duration	Duration(%)	Description(Type:Name)	ELog	EZone	Drill Up/Down	Timestamp
903.000ms	100.00%	SOAService:*ShoppingServic*	Elog	wg1	Drill Up	2017-12-18 11:29:47.569
1.000ms	0.11%	Client:Getclient				2017-12-18 11:29:47.573
188.000ms	20.82%	SOACall:*RankService.sear*			Drill Down	2017-12-18 11:29:47.575
1.000ms	0.11%	Client:Getclient				2017-12-18 11:29:47.773
13.000ms	1.44%	SOACall:*ShopBroadcastInfoSer*			Drill Down	2017-12-18 11:29:47.774
1.000ms	0.11%	Client:Getclient				2017-12-18 11:29:47.808
1.000ms	0.11%	SOACall:*ABTestService.getFactors*			Drill Down	2017-12-18 11:29:47.810
1.000ms	0.11%	Client:Getclient				2017-12-18 11:29:47.826
49.000ms	5.43%	SOACall:*ElemePromotionServic*			Drill Down	2017-12-18 11:29:47.828
1.000ms	0.11%	Client:Getclient				2017-12-18 11:29:47.878
336.000ms	37.21%	SOACall:*ElemePromotionS*			Drill Down	2017-12-18 11:29:47.879
1.000ms	0.11%	Client:Getclient				2017-12-18 11:29:48.215
2.000ms	0.22%	SOACall:*MemberSe*			Drill Down	2017-12-18 11:29:48.217
1.000ms	0.11%	Client:Getclient				2017-12-18 11:29:48.219
13.000ms	1.44%	SOACall:*ElemePromotionSe*			Drill Down	2017-12-18 11:29:48.220
11.000ms	1.22%	Client:Getclient				2017-12-18 11:29:48.233
158.000ms	17.50%	SOACall:*ElemePromotio*			Drill Down	2017-12-18 11:29:48.245
1.000ms	0.11%	Client:Getclient				2017-12-18 11:29:48.404
49.000ms	5.43%	SOACall:*ElemePromotionSer*			Drill Down	2017-12-18 11:29:48.406
9.000ms	1.00%	Redis:Stats				2017-12-18 11:29:48.472

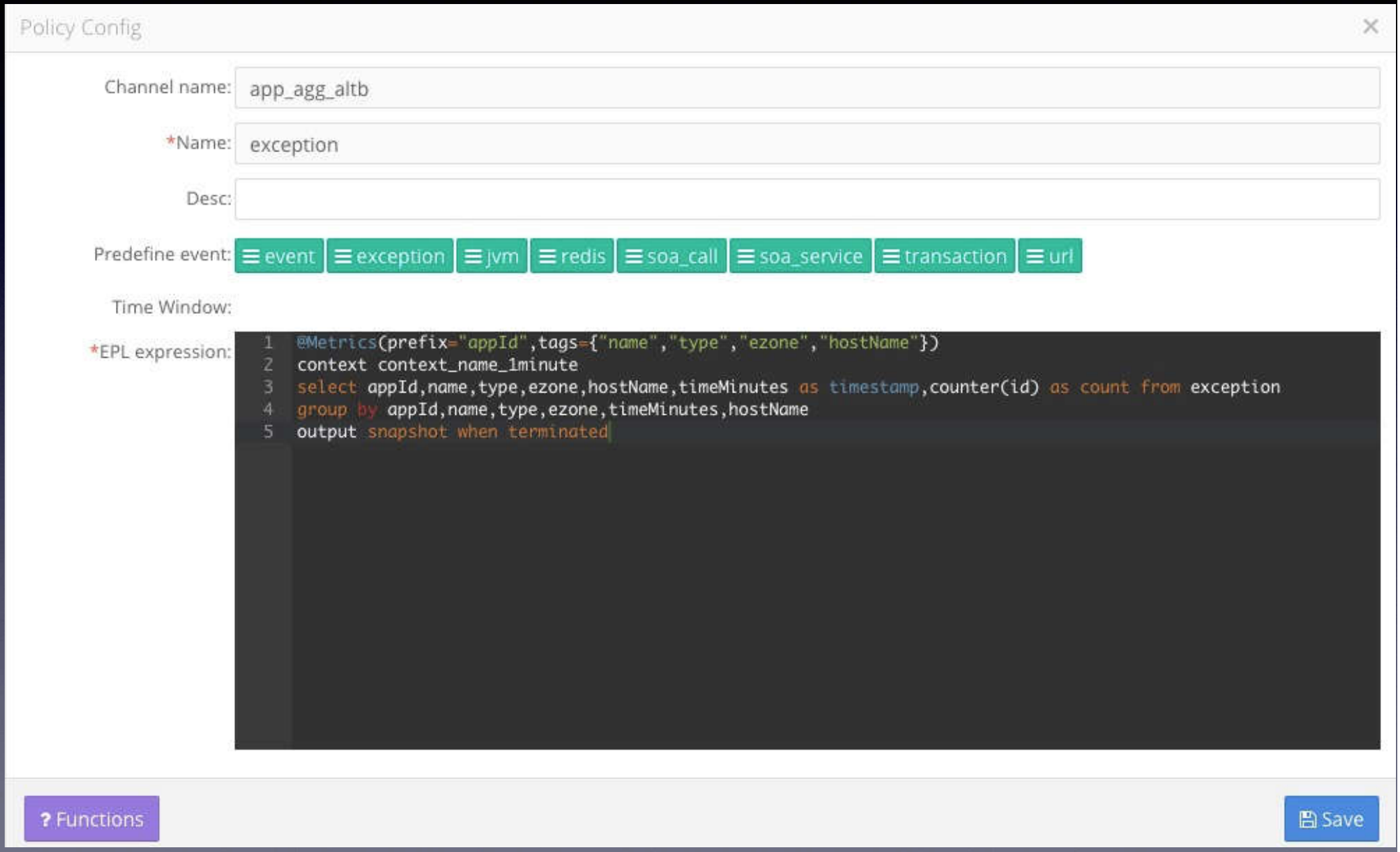
Collector


Thrift TCP Server
Persist Queue
Consumer Task
Dispatch
Route
Limit
Auth
Kafka
Collector

计算框架

- 目前的数据处理及计算是基本Esper自己实现的；
- 没有使用目前比较流行的Storm/Spark Streaming这个的架构；
- 本身的业务逻辑不是很复杂，引入Storm这样的组件同时也增加了运维成本，Storm之身的一些问题，如雪崩等；
- 网络资源比较浪费；

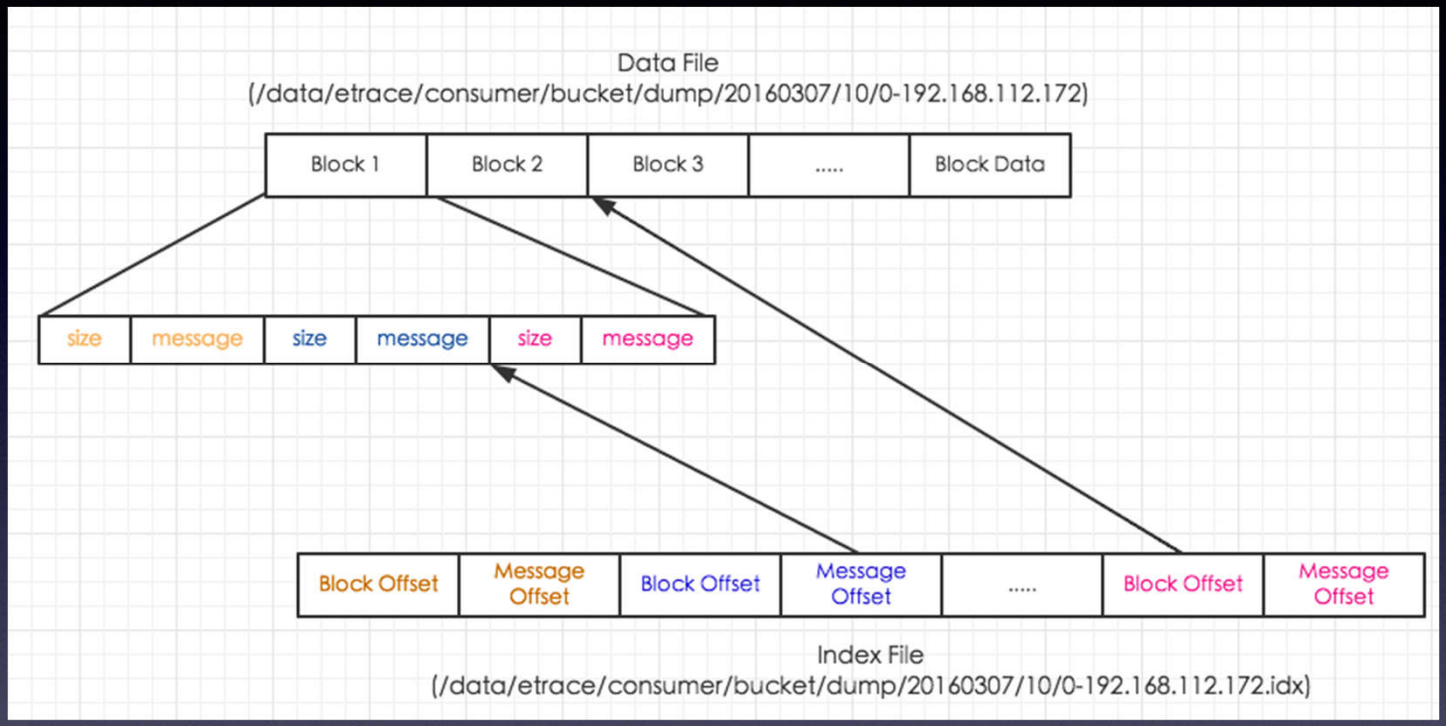
Shaka


Policy Config
Channel name:
app_agg_altb
*Name:
exception
Desc:
Predefine event:
event
exception
jvm
redis
soa_call
soa_service
transaction
url
Time Window:
*EPL expression:
@Metrics(prefix="appId",tags={"name","type","ezone","hostName"})
context context_name_1minute
select appId,name,type,ezone,hostName,timeMinutes as timestamp,counter(id) as count from exception
group by appId,name,type,ezone,timeMinutes,hostName
output snapshot when terminated
? Functions
Save

存储方案

- 存储使用HDFS及HBase，而不是目前比较火的ELK；
- HDFS存储Raw Data；
- HBase存储Index数据及一些采样数据；
- Raw Data按Block存储，每个Block通过Snappy压缩；
- 整个存储过程使用多个异步进行处理；
- 所有的指标目前存储在LinDB中；

File Format

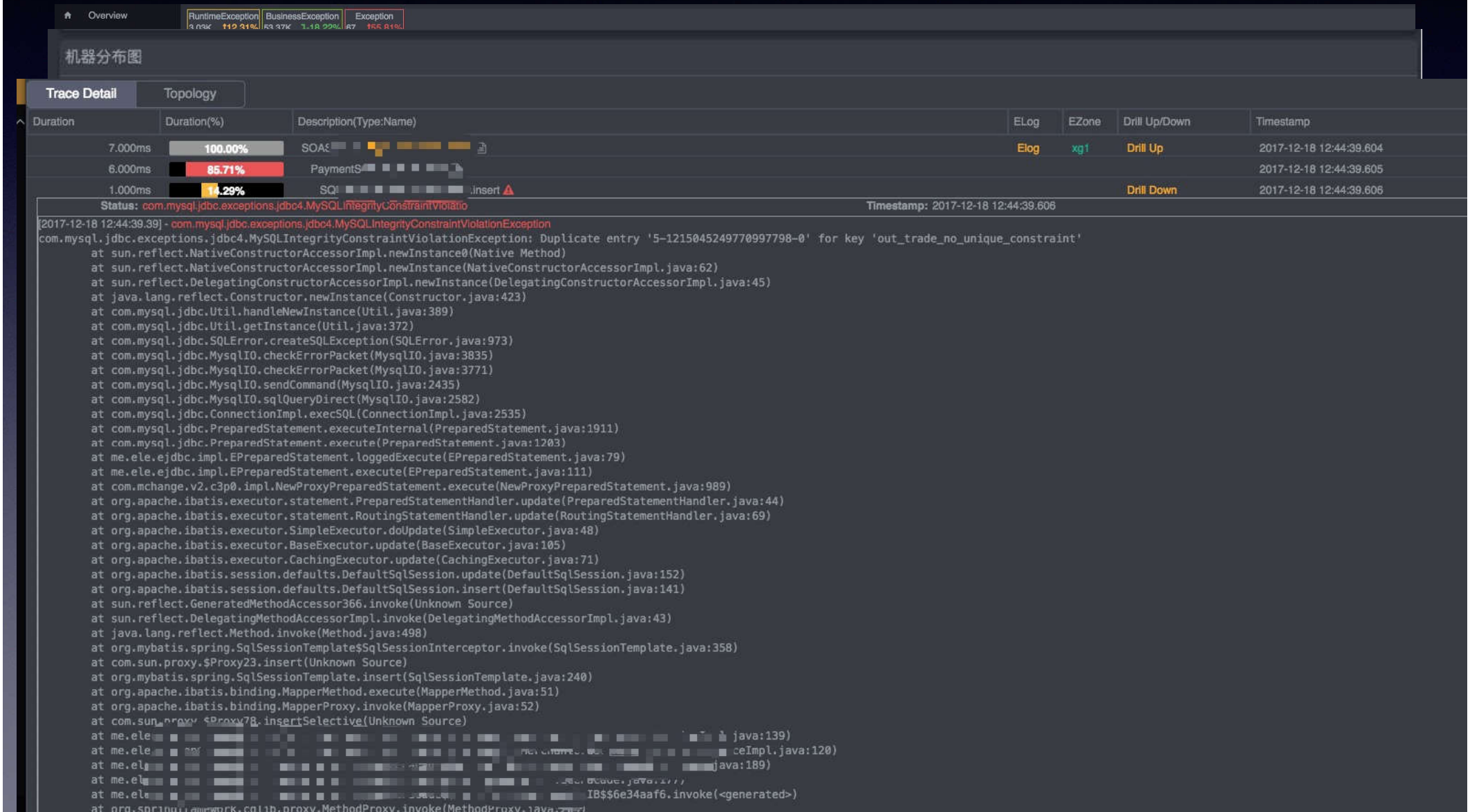


- 在Block写入的时候，为每个Block生成一个递增的Id；
- 每个Block 64K + Snappy Compress;
- Index File存储Block Offset(long)+Message Offset(long);

HBase

- 目前在HBase中存储index和采样数据；
- HBase主要解决热点问题，及Key的设计是否合理；
- 通过pre-split region，把region分散在每个Region Server中；
- 写入的时候根据Region把数据分散在每个Region上面；

Demo


Overview
RuntimeException
BusinessException
Exception
机器分布图
Trace Detail
Topology
Duration
Duration(%)
Description(Type:Name)
ELog
EZone
Drill Up/Down
Timestamp
7.000ms
100.00%
Elog
xg1
Drill Up
2017-12-18 12:44:39.604
6.000ms
85.71%
2017-12-18 12:44:39.605
1.000ms
14.29%
.insert
Drill Down
2017-12-18 12:44:39.606
Status: com.mysql.jdbc.exceptions.jdbc4.MySQLIntegrityConstraintViolatio
Timestamp: 2017-12-18 12:44:39.606
[2017-12-18 12:44:39.39] - com.mysql.jdbc.exceptions.jdbc4.MySQLIntegrityConstraintViolationException
com.mysql.jdbc.exceptions.jdbc4.MySQLIntegrityConstraintViolationException: Duplicate entry '5-1215045249770997798-0' for key 'out_trade_no_unique_constraint'
at sun.reflect.NativeConstructorAccessorImpl.newInstance0(Native Method)
at sun.reflect.NativeConstructorAccessorImpl.newInstance(NativeConstructorAccessorImpl.java:62)
at sun.reflect.DelegatingConstructorAccessorImpl.newInstance(DelegatingConstructorAccessorImpl.java:45)
at java.lang.reflect.Constructor.newInstance(Constructor.java:423)
at com.mysql.jdbc.Util.handleNewInstance(Util.java:389)
at com.mysql.jdbc.Util.getInstance(Util.java:372)
at com.mysql.jdbc.SQLError.createSQLException(SQLError.java:973)
at com.mysql.jdbc.MysqlIO.checkErrorPacket(MysqlIO.java:3835)
at com.mysql.jdbc.MysqlIO.checkErrorPacket(MysqlIO.java:3771)
at com.mysql.jdbc.MysqlIO.sendCommand(MysqlIO.java:2435)
at com.mysql.jdbc.MysqlIO.sqlQueryDirect(MysqlIO.java:2582)
at com.mysql.jdbc.ConnectionImpl.execSQL(ConnectionImpl.java:2535)
at com.mysql.jdbc.PreparedStatement.executeInternal(PreparedStatement.java:1911)
at com.mysql.jdbc.PreparedStatement.execute(PreparedStatement.java:1203)
at me.ele.ejdbc.impl.EPreparedStatement.loggedExecute(EPreparedStatement.java:79)
at me.ele.ejdbc.impl.EPreparedStatement.execute(EPreparedStatement.java:111)
at com.mchange.v2.c3p0.impl.NewProxyPreparedStatement.execute(NewProxyPreparedStatement.java:989)
at org.apache.ibatis.executor.statement.PreparedStatementHandler.update(PreparedStatementHandler.java:44)
at org.apache.ibatis.executor.statement.RoutingStatementHandler.update(RoutingStatementHandler.java:69)
at org.apache.ibatis.executor.SimpleExecutor.doUpdate(SimpleExecutor.java:48)
at org.apache.ibatis.executor.BaseExecutor.update(BaseExecutor.java:105)
at org.apache.ibatis.executor.CachingExecutor.update(CachingExecutor.java:71)
at org.apache.ibatis.session.defaults.DefaultSqlSession.update(DefaultSqlSession.java:152)
at org.apache.ibatis.session.defaults.DefaultSqlSession.insert(DefaultSqlSession.java:141)
at sun.reflect.GeneratedMethodAccessor366.invoke(Unknown Source)
at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:43)
at java.lang.reflect.Method.invoke(Method.java:498)
at org.mybatis.spring.SqlSessionTemplate$SqlSessionInterceptor.invoke(SqlSessionTemplate.java:358)
at com.sun.proxy.$Proxy23.insert(Unknown Source)
at org.mybatis.spring.SqlSessionTemplate.insert(SqlSessionTemplate.java:240)
at org.apache.ibatis.binding.MapperMethod.execute(MapperMethod.java:51)
at org.apache.ibatis.binding.MapperProxy.invoke(MapperProxy.java:52)
at com.sun.proxy.$Proxy78.insertSelective(Unknown Source)
java:139)
ceImpl.java:120)
java:189)
IB$$6e34aaf6.invoke(<generated>)

Thank You

Q&A