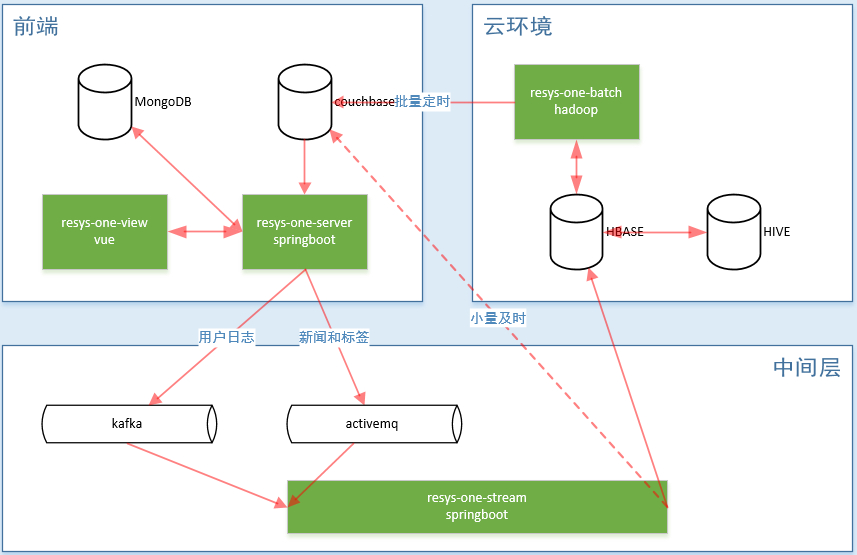
resys-one开发手册

# 概述

1. 本手册对应系统版本：
   1. 1.0
2. 所使用的组件：
   1. springboot
   2. iview-admin
   3. hbase
   4. hive
   5. hadoop
   6. kafka
   7. activemq
   8. couchbase
   9. mongodb
3. 需要掌握的基本知识：
   1. 流式数据编程基础
   2. MapReduce计算基础
   3. linux shell
4. 系统目标
   1. 如何隔离实时计算和离线计算
   2. 如何隔离业务工程和算法工程
5. 系统程序组成
   1. resys-one-batch
      1. 执行各个mapreduce任务以及存放脚本和shell运行环境
   2. resys-one-server
      1. 提供前端的接口服务
   3. resys-one-spring
      1. 所有基于springboot的服务的框架
   4. resys-one-stream
      1. 提供流式数据处理服务，处在server和云环境之间的一个服务
   5. resys-one-util
      1. 所有基于java的组件都需要继承的通用工具包，里面全部是静态方法
   6. resys-one-view
      1. 基于iview-admin的前端测试页面，用于系统管理及模拟数据结果

# 系统架构



如图，按照系统部署场景分，可以分为三块：前端、中间层和云环境。这样分割的原因：

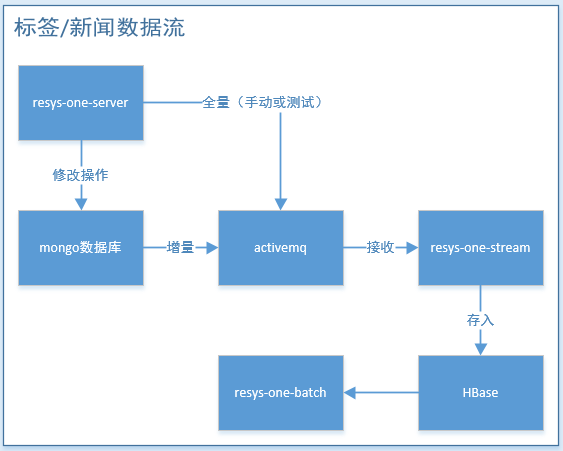
1. 前端和云环境部署环境不同，前端组件可以自由部署且没有环境限制，云环境下的组件和任务程序必须部署在hadoop集群或可以自由访问hadoop集群的服务器上。
2. 安全性不同，前端环境和云环境对开发人员、运维人员的安全要求不同，部署后，开发人员需要对系统进行持续的优化，这会与运维人员的工作产生冲突。

# 数据流

## 标签/新闻数据流

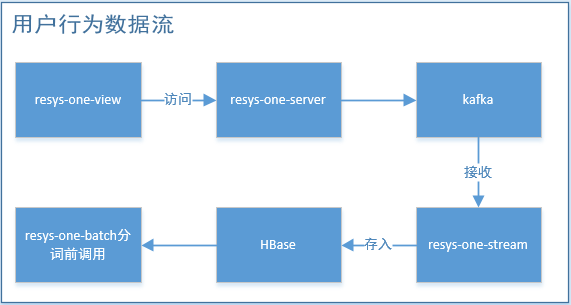
标签/新闻数据流原始内容保存在mongo中（实际运行中由业务组管理），但是比如新发送新闻或对原有新闻进行修改（错别字更正、降低新闻热度、下架某个新闻等）需要系统立刻做出相应。因此新闻数据由两条线更新：

1. 前端->全量导出为文件->resync或其它文件传递的方式->resys-one-batch将文件全量导入到hbase数据表news\_all
2. 前端->拦截到增删改操作->将增删改后的新闻送入activemq->resys-one-stream接收后送入HBase->resys-one-batch计算调用



## 用户行为日志数据流

用户行为日志因为实时性高，且数据量大，所以要走kafka。



# 词管理

系统内的词有两部分，一部分是前端所配置的标签，另一部分是通用标签。

## 标签词

配置词是在前端编辑或网站管理人员所配置的标签，用于对文章提供标签化。

一般来讲，大多数博客或新闻系统对新闻的管理分为标签管理和分类管理两种，作为推荐来说，两类信息都有利用价值，但本类下我们主要使用标签进行主要信息的计算。

标签词数量不需要太多，但重复性要高，无用的标签不要打，尽量打能反应文章有归类和特色的标签。建议整个系统标签不要超过300个。

## 全量词典

在计算相关新闻的时候，用到的词典是全量词典。这个词典由算法的同学管理，既需要管理词增减，也需要对词的nature和其它属性信息进行管理。这些词一般具有很强的通用性和召回性，和前端的标签无关。

全量词典要求尽可能充分，能够分出最多的词为佳，数量可以是几十万词。

# 运行步骤

## 启动前端工程

1. 确保下面的组件是正常运行：
   1. mongodb
   2. couchbase
   3. kafka
   4. activemq
2. 如果上述组件运行正常，启动下面两个工程（顺序不限）：
   1. resys-one-server
   2. resys-one-view

## 启动中间工程

1. 确保下面的组件是正常运行的：
   1. kafka
   2. activemq
2. 如果上述组件运行正常，启动下面这个工程：
   1. resys-one-stream

## 启动各个计算脚本

1. 确保下面的组件是正常运行的：
   1. hadoop
   2. zookeeper
   3. hbase
   4. couchbase
2. 全量导入所有新闻
   1. run\_import\_news\_all
3. 全量导入词典
   1. run\_import\_dictionary
4. 计算新闻热度
   1. run\_cal\_news\_click.sh
   2. run\_cal\_hot\_news\_one.sh
   3. run\_cal\_hot\_news\_two.sh
   4. run\_serve\_hot\_news.sh
5. 计算新闻相关性
   1. run\_cal\_news\_token.sh
   2. run\_cal\_news\_invert.sh
   3. run\_cal\_news\_correlation.sh
   4. run\_serve\_news\_correlation.sh