Engineering Requirements Document

[开发设计文档]

**项目名称: 推送平台**

**创建者：**

**文档修改纪录**

|  |  |  |  |
| --- | --- | --- | --- |
| **版本** | **日期** | **变化描述** | **修改人** |
| 1.0.0 | 2017-10-17 | 初始版本 | 仇加龙 |
| 1.0.1 | 2017-10-18 | 补充限流、断路器、手动推送功能 | 仇加龙 |
|  |  |  |  |

# 概述

## 产品目标

构建通用、高效、稳定、易管控的推送服务，以专注业务，简化开发，避免重复造轮子

## 产品概述

* 提供接入功能解耦业务与推送以提供可复用的推送服务
* 支持消息推送拉取
* 支持多平台、多协议推送
* 易于监控管理，如管理接收方优先级、消息优先级，开关、消息转换器，监控发送量等
* 易于横向扩展，提供高性能、高可用、易伸缩的推送服务

其中多平台、多协议推送对APP、TCP的支持为二期规划，其余为一期规划

# 设计

## 对现有设计分析

现存在多套消息推送系统且与业务耦合程度较高，难以复用，多个团队各自维护，反复踩坑，开发维护成本较高，且采用系统通过消息存储到数据库及定时任务读取发送消息模式，存在消息丢失、难以策略化推送，管控缺失，伸缩性较差，难以扩展

## 新需求相关的设计

### 业务流程图

****

### 数据库E-R设计





### 部署结构



### 性能方面的考虑

1. Scheduler、Pusher、Admin支持分布式及横向扩展，瓶颈在数据库存取上，考虑按消息类型分表、缓冲池批量加载待发送消息、本地缓存接收方及消息类型配置信息以削减数据库压力及提高数据库横向扩展性
2. 当前分销产品数量9033，分销商数量11724，整个分销消息日吞吐量为600W，根据产品人员提供的分销未来三年数据量，估算的消息量如下：

|  |  |  |  |
| --- | --- | --- | --- |
| 年份 | 分销产品数量 | 分销商数量 | 消息日吞吐量 |
| 2018 | 10387 | 14068 | 827W |
| 2019 | 11742 | 16413 | 1091W |
| 2020 | 13097 | 18758 | 1392W |

现有推送系统主要采用消息存储数据库🡪定时任务取消息数据🡪消息推送到接收方的模式，通过消息数据预加载、慢消息隔离、均衡负载等优化措施预期性能方面有部分提升，后期需通过压测评估是否满足未来的业务增长量

### 异常处理

1. 基于zookeeper的节点注册监控功能管理集群节点的上下线
2. 对于网络问题导致数据发送失败问题，采用数据暂存本地磁盘，定时重试的处理方式

# 数据存储

## 关系存储结构图

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 接入表 | | | | | |
| 字段 | 类型 | 是否主键 | 是否唯一 | 是否可为空 | 说明 |
| token | char(128) | 是 | 是 | 否 | 接收方标识 |
| name | varchar(200) | 否 | 否 | 否 | 接收方名称 |
| receive\_type | tinyInt | 否 | 否 | 否 | 接收类型，1：推送，2：拉取 |
| push\_type | tinyInt | 否 | 否 | 否 | 推送类型，1：HTTP，2：TCP，3：JAR |
| push\_url | varchar(200) | 否 | 否 | 否 | 推送接口地址 |
| is\_open | tinyInt | 否 | 否 | 否 | 开关，1：开启，0：关闭 |
| converter\_type | tinyInt | 否 | 否 | 否 | 转换器，1：json，2：xml，3：form |
| priority | tinyInt | 否 | 否 | 否 | 优先级 |
| threshold | int | 否 | 否 | 否 | 拉取频率或发送频率限制 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 消息表 | | | | | |
| 字段 | 类型 | 是否主键 | 是否唯一 | 是否为空 | 说明 |
| id | bigInt | 是 | 是 | 否 | 自增主键 |
| type\_id | int | 否 | 否 | 否 | 消息类型 |
| content | varchar(1000) | 否 | 否 | 否 | 消息内容 |
| status | tinyInt | 否 | 否 | 否 | 消息状态，0：待发送，1：发送中，2：已发送 |
| receive\_time | datetime | 否 | 否 | 否 | 接收时间 |
| modify\_time | timestamp | 否 | 否 | 否 | 状态变更时间 |
| push\_id | char(100) | 否 | 否 | 否 | 发送器标识 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 发送表 | | | | | |
| 字段 | 类型 | 是否主键 | 是否唯一 | 是否为空 | 说明 |
| msg\_id | bigInt | 否 | 否 | 否 | 消息id |
| token | varchar(128) | 否 | 否 | 否 | 接收方标识 |
| status | tinyInt | 否 | 否 | 否 | 发送状态，0：待发送，1：发送中，3：已发送 |
| modify\_time | timestamp | 否 | 否 | 否 | 状态变更时间 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 消息配置表 | | | | | |
| 字段 | 类型 | 是否主键 | 是否唯一 | 是否为空 | 说明 |
| id | int | 是 | 是 | 是 | 主键 |
| type | varchar(100) | 是 | 否 | 否 | 消息类型 |
| priority | tinyInt | 否 | 否 | 否 | 优先级 |
| is\_open | tinyInt | 否 | 否 | 否 | 开关，1：开启，0：关闭 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 全局统计表 | | | | | |
| 字段 | 类型 | 是否主键 | 是否唯一 | 是否为空 | 说明 |
| [reach](file:///C:\Users\Administrator\AppData\Local\youdao\dict\Application\7.3.0.0807\resultui\dict\?keyword=reach)\_count | int | 否 | 否 | 否 | 到达量 |
| receive\_count | int | 否 | 否 | 否 | 接收量 |
| fail\_count | int | 否 | 否 | 否 | 失败量 |
| send\_count | int | 否 | 否 | 否 | 发送量 |
| timestamp | int | 否 | 否 | 否 | 统计时间 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 消息类型统计表 | | | | | |
| 字段 | 类型 | 是否主键 | 是否唯一 | 是否为空 | 说明 |
| type\_id | varchar(100) | 是 | 是 | 是 | 消息类型id |
| [reach](file:///C:\Users\Administrator\AppData\Local\youdao\dict\Application\7.3.0.0807\resultui\dict\?keyword=reach)\_count | int | 否 | 否 | 否 | 到达量 |
| receive\_count | int | 否 | 否 | 否 | 接收量 |
| fail\_count | int | 否 | 否 | 否 | 失败量 |
| send\_count | int | 否 | 否 | 否 | 发送量 |
| timestamp | int | 否 | 否 | 否 | 统计时间 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 消息发送进度监控 | | | | | |
| 字段 | 类型 | 是否主键 | 是否唯一 | 是否为空 | 说明 |
| msg\_id | bigInt | 是 | 是 | 是 | 消息id |
| [reach](file:///C:\Users\Administrator\AppData\Local\youdao\dict\Application\7.3.0.0807\resultui\dict\?keyword=reach)\_count | int | 否 | 否 | 否 | 到达量 |
| send\_count | int | 否 | 否 | 否 | 发送量 |
| fail\_count | int | 否 | 否 | 否 | 失败量 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 接收方发送进度监控 | | | | | |
| 字段 | 类型 | 是否主键 | 是否唯一 | 是否为空 | 说明 |
| token | varchar(128) | 是 | 是 | 是 | 接收方token |
| [reach](file:///C:\Users\Administrator\AppData\Local\youdao\dict\Application\7.3.0.0807\resultui\dict\?keyword=reach)\_count | int | 否 | 否 | 否 | 到达量 |
| receive\_count | int | 否 | 否 | 否 | 接收量 |
| fail\_count | int | 否 | 否 | 否 | 失败量 |
| send\_count | int | 否 | 否 | 否 | 发送量 |
| timestamp | int | 否 | 否 | 否 | 统计时间 |

# 功能设计

## 接入管理

1. 提供管理界面及接口录入接收方标识、接收方式(推送\拉取)、推送方式(TCP\HTTP\JAR)、开关、优先级、出口消息转换器等信息，生成token，供发送方发送消息使用
2. 提供分组管理功能以支持接入信息的批量管理

## 消息类型管理

管理消息类别、优先级、开关等信息

## 监控预警

* 全局统计：按时间段统计接收量、发送量、到达量、失败量
* 消息类别统计：按时间段统计某类别消息接收量、发送量、到达量、失败量
* 消息发送进度统计：统计某消息发送量、到达量、失败量、已发送接收方、未发送接收方
* 预警：对发送失败率>阈值等情况预警

## 消息投递

规范消息格式，提供HTTP接口及TCP客户端按Msg,TokenA,B,C….方式发送消息

## 消息分发

根据消息优先级(接收方优先级、消息优先级、发送时间)、推送开关、流量控制等信息派发消息，并提供相应的保障措施：

* + 消息隔离，避免不同类型的消息相互影响及慢消息拖慢整个消息系统，隔离方式：

1. 数据库层面按消息类型分表存储
2. scheduler模块通过统计消息发送耗时，动态识别慢消息，并按消息类别及慢消息两个维度分不同队列预加载消息，并通过加权随机或加权轮训等方式均衡多个队列的消息发送
3. pusher模块根据scheduler模块识别出的消息快慢分发到快慢线程池
   * 推送断路器：熔断及恢复某接收方的消息推送，避免由于接收方的系统不稳定影响整个推送服务
   * 限流：控制某接收方的消息推送频率，避免压垮接收方系统，导致业务流程受影响
   * 预警：消息推送接收方失败率达到阈值，进行预警，并提供推送开关以开启或关闭某接收方的推送

## 消息推送

按照配置的消息转换器Json\Xml\Form及配置的推送方式TCP\HTTP\JAR等方式推送消息

## 消息拉取

提供Http消息拉取接口供接收方主动拉取消息，提供限流配置

## 消息手动推送

提供界面手动立即推送消息

# 集中配置

## 支持集中配置点

* 接收方配置：消息接收方式、优先级、开关、消息转换器，配置信息存储于接入表
* 消息配置：优先级、开关，配置信息存储于消息配置表
* 系统运行参数：如Scheduler、Pusher池资源大小等对查询功能要求不高的配置信息存储于统一配置中心sweet或zookeeper中

# 监控

## 业务监控点

* 全局数据监控：到达量、接受量、发送量、失败量
* 某类型消息监控：到达量、接受量、发送量、失败量
* 某消息发送进度监控：发送量、失败量、已发送接受方、未发送接受方

## 系统监控点

系统各节点的CPU、内存、网络、硬盘等资源信息接入zabbix

# 现有系统参考

目前规划的推送平台与分销在用的LTS架构类似，经分析讨论，LTS通信模块、可用性模块等存在较多复用点：

1. LTS通信模块是基于Netty开发的一套私有协议，封装了一套RPC方法，更接近应用层，提供了同步、异步调用远程服务等方法，并具有较好的扩展性，基于LTS通信模块做扩展开发相对基于Netty等开源NIO框架重新封装私有协议成本更低
2. LTS针对可用性问题实现了一套高可用方案，基于zookeeper监控集群节点上下线及节点主从切换，并针对由于网络问题导致数据发送失败等问题提供了数据暂存本地磁盘，定时重试的解决方案

基于以上考虑，后续推送平台基于LTS做改造开发