趣讯一起充

消息中心概要设计

R1.0

目录

1 简介 1

1.1 目的 1

1.2 文档范围 1

1.3 参考 1

2 系统概述 2

2.1 说明 2

2.2 消息处理流程与概念 2

2.3 配置信息 3

2.4 消息处理实例说明 3

2.5 业务限制 4

3 系统架构 5

3.1 系统间关系 5

3.2 系统内模块关系 5

4 系统代码 6

4.1 MessageCenter 6

5 系统接口 8

5.1 MessageCenter 8

6 数据模型(MongoDB) 9

6.1 原始消息message 9

6.2 用户消息user\_message 9

6.3 订阅关系user\_subscribe 9

7 修改记录 10

# 简介

## 目的

本文描述了消息中心模块的整体方案。

## 文档范围

本文档阅读范围

1. 消息中心开发人员

## 参考

# 系统概述

## 说明

消息中心是独立模块，主要功能是收集各个模块生成的系统和业务消息，并根据规则进行预处理，并将消息送达用户。

## 消息处理流程与概念

消息收件箱

inbox

消息

合并/过滤/

/压缩/持久化

消息预处理

原始消息

订阅处理

持久化

短信通道SMS

微信通道WECHAT

邮件通道EMAIL

用户消息

消息分发

### 收件箱

消息中心对外接收消息的入口，

收件箱可配置单位时间消息接收量和最大消息量，超过此限额则丢弃消息。

收件箱获取消息后，根据消息类型调用不同的消息预处理模块。

### 消息预处理模块

系统可配置的消息预处理模块，可根据消息类型配置。

消息预处理可对不重要的消息进行消息合并，并最终将消息持久化到数据库中。

### 订阅处理模块

订阅处理模块根据系统配置的用户-消息关系，将原始消息关联到用户消息，并持久化到数据库中。

### 消息分发

定期扫描用户消息，根据订阅情况，将消息通过各种消息通道推送给用户。

## 配置信息

config需要配置一下关系

### 类型消息模板，根据消息类型配置信息模板

### 消息类型-预处理模块 关联关系

### 消息类型-订阅处理 关联关系

## 消息处理实例说明

### 停电报警 type=system.power\_failure

停电报警属于紧急情况，消息以最快的速度发送给用户，消息不需要合并

机房发生停电，电源监控程序检测到停电，按分钟发送报警

type=system.power\_failure

payload={ups信息}

通过接口发送至inbox，消息被缓存至Redis

system.power\_failure对应的预处理程序为紧急处理程序，操作方式为立即持久化到数据库。

用户订阅处理模块检测到新的消息，取出新消息，根据type=system.power\_failure以及订阅关系，查找关心的用户，并生成用户消息。

消息分发模块获取新增加的用户消息，找到处理方法，根据此方法发送消息。

### 卡单报警type=madeira.order\_timeout

系统卡单是常见的业务消息，不需要非常高的时效性，而且一般情况下，系统一直存在卡单，消息需要做合并处理，否则会收到过多的消息。

madeira运维程序定时扫描系统卡单，对应需要客服介入的卡单，将订单列表发送到消息中心

type=madeira.order\_timeout

palyload={order\_list: [‘Q11111111’, ‘Q222222’}

通过接口发送至inbox，消息被缓存至Redis

madeira.order\_timeout对应的预处理程序为缓存合并程序，系统保存一个

## 业务限制

# 系统架构

## 系统间关系

madeira

purus

hardware

agent

mc

## 系统内模块关系

消息收件箱

inbox

消息

合并/过滤/

/压缩/持久化

消息预处理

原始消息

订阅处理

持久化

短信通道SMS

邮件通道EMAIL

用户消息

消息分发

MongoDB

Task

# 系统代码

## MessageCenter

代码结构（一级目录）

|  |  |
| --- | --- |
| handlers/ | HTTP接口Handlers |
| core/ | 核心处理器 |
| task/ | 定时任务 |

### handlers/inbox.py

#### InboxHandler

1. def post(self, path)

对传入的消息报文验证合法性

通过 redis中的id:msg自增，得到消息的id

将消息缓存到Redis

将消息id LPUSH 到 list:ready

调用 core.preprocessing

### core/preprocessing.py

#### def preprocessing(msg\_id, type)

预处理消息入口

从list:ready RPOP弹出一个消息id

从Redis获取消息类型

根据type，从config读取对应的预处理函数，通过动态调用方式调用

如果没有定义，则设置一个default的预处理函数

madeira.order\_timeout : direct

#### def preprocessing\_direct(msg\_id, type)

直接将消息持久化到MongoDB

将消息类型放到 set:types

### task/subscribe.py

#### SubscribeTask

def run():

从 set:types 取出当前系统中待的消息类型

根据type从配置充读取对应的订阅处理

调用 core.subcribe\_ 订阅处理模块

subscribe:

madeira.order\_timeout : direct

### core/subscribe.py

#### def subscrible\_direct(msg\_type)

从mongodb根据type获取未处理的消息

根据type获取订阅关系

直接生成用户消息，存入mongodb

根据订阅关系中定义的处理方式，调用相应的处理代码

### core/send.py

#### def send\_by\_sms(msg, handler)

调用蝶信接口，发送短信

#### def send\_by\_email(msg, handler)

调用邮件发送接口，发送邮件通知

# 系统接口

## MessageCenter

### /inbox

1. 输入参数：

{

“msg”: “超时订单”,

“type”: “madeira.order\_timeout“,

“payload”: { “order\_list”: [‘Q10000000000’] }

}

1. 输出参数：

{“status”: “ok”}

# 数据模型(MongoDB)

## 原始消息message

|  |  |  |  |
| --- | --- | --- | --- |
| 属性 | 类型 | 说明 | 例子 |
| \_id | string | 消息唯一标识，通过Redis生成 | 1000001 |
| msg | string | 文字类说明 | 订单超过2小时无回调 |
| type | string | 系统约定的消息类型 | madeira.timeout |
| payload | object | 基于JSON的格式化消息体 | {‘order\_id’: ‘Q123123213213’} |
| create\_time | date | 创建时间 |  |
| processing\_time | date | 处理时间，无此属性表示待处理 |  |

## 用户消息user\_message

|  |  |  |  |
| --- | --- | --- | --- |
| 属性 | 类型 | 说明 | 例子 |
| \_id | string | 唯一标识，通过Redis生成 |  |
| operator\_id | string | 用户（操作员）标识 | 2 |
| msg\_list | list | 对应的原始消息\_id列表 | [‘1’,’2’,’3’] |
| read\_time | date | 阅读时间 | 无此属性表示未阅读 |
| send\_time | date | 发送时间 | 无此属性表示未发送 |
| sub\_id | string | 订阅关系ID | 发送时根据此字段找到处理方法 |

## 订阅关系user\_subscribe

|  |  |  |  |
| --- | --- | --- | --- |
| 属性 | 类型 | 说明 | 例子 |
| \_id | string | 主键 |  |
| operator\_id | string | 用户（操作员） | 2 |
| type | string | 类型 | madeira.order\_timeout |
| handling | list | 处理方式列表 | [{‘handler’: ‘sms’, mobile: ‘139xxxxxxxx’},  {‘ handler’: ‘email’, address: ‘abc@example.com’}] |

# 修改记录

|  |  |  |  |
| --- | --- | --- | --- |
| 日期 | **版本** | **作者** | **说明** |
| 2016/1/20 | Rev 1 | 羊笑佺 | 初始版本 |
|  |  |  |  |