钉钉消息发送思考

一、背景

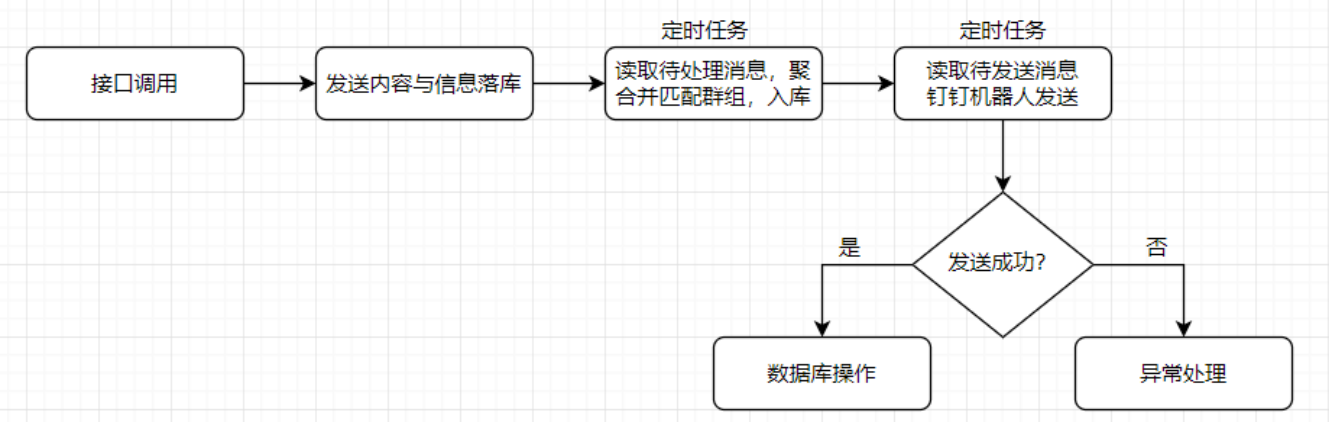
1. 钉钉消息机器人目前一分钟只能发送20条消息，超过该限制时，会被限流十分钟。使用机器人发送消息时，若一时间推给机器人发送消息的请求过多（每分钟超过二十条），会导致超过数量的消息发不出去，并且机器人被限流。
2. 当前消息发送时需要指定一个机器人，而一个群最多可以指定6个机器人。
3. 目前所有的异常都集中在部门兜底群中进行警报，网络抖动时产生的大量异常堆栈有时会淹没需要关注的错误异常。
4. 很多想要通过钉钉发送的消息通过error日志的形式报警到了兜底中。 兜底群中所有报警都集中在了部门兜底群中，比较杂乱。

二、目标

做一个钉钉群消息发送平台，对外提供接口，外部通过调用接口发送钉钉消息。

通过使用该系统进行发送，解决的问题：
1、消息聚合：某一时间段内，重复消息只发送一次。
2、机器人一定不被限流：避免一分钟内某个机器人发送消息过多导致被限流的场景
3、消息的自动分流：通过配置，可以将同一应用不同消息发送到不同的群中，也可以将不同应用的消息发送到同一群中。消息选择群的规则可配置。
4、消息的正常情况下的及时触达：保证消息一定会在对应的群中进行报警。
5、重要消息在大流量请求下的优先发送：某一群同时有大量消息待发送时，对消息进行优先级排序，优先发送重要消息。
6、消息发送失败有相应的异常处理与反馈逻辑：消息发送失败时，进行对应的逻辑判断，对于部分失败原因进行反馈与处理。

三、大致流程



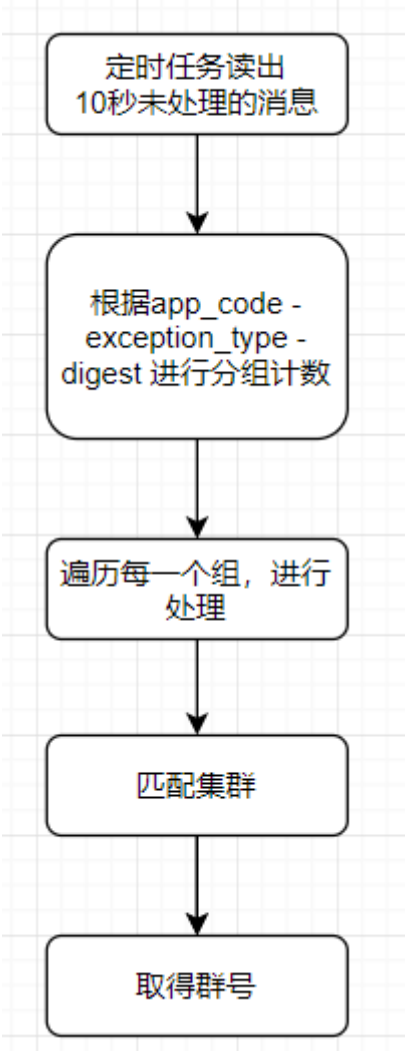
四、详细设计

1、消息聚合：

一是为了减少要发送的消息数量，二是太多内容重复的消息发送到群中不利于整体的阅读与反应，也容易看漏其他的消息。通过聚合重复消息，减少消息数量，使更多不同内容的消息发送到群中得到关注。

聚合有两处：

信息与相关数据落库待处理表后，定时从表示读出新增的数据，将相同的消息根据app_code-execption_type-digest进行聚合后，再按照匹配规则进行匹配，获得要发送的群号，放入到待发送的表中：



app_code表示应用名，exception_type代表异常类型，主要是为现有的兜底报警异常设计，为了将不同的异常类型能够区分开发到不同的群众，digest为信息摘要，异常报警调用时为原逻辑，消息发送接口调用时为调用处的堆栈信息。

定时从待发送的表中读取消息，此时仍有一处聚合逻辑，

通过这样的方式，一定程度上减少了消息重复的问题，避免了大量相同信息发送到同一钉钉群中浪费资源、不易阅读等情况。

2、保障机器人不被限流：

通过限制每分钟群内发送的数量从而保证机器人不被限流。根据定时任务可以将每分钟化为六个时间段，每个时间段的发送数量最大值为19*时间段-已发送的数量。

redis中维护每个群已发送消息数量的信息times，每分钟清0。

则每次可以发送的数量可以简单的表示为

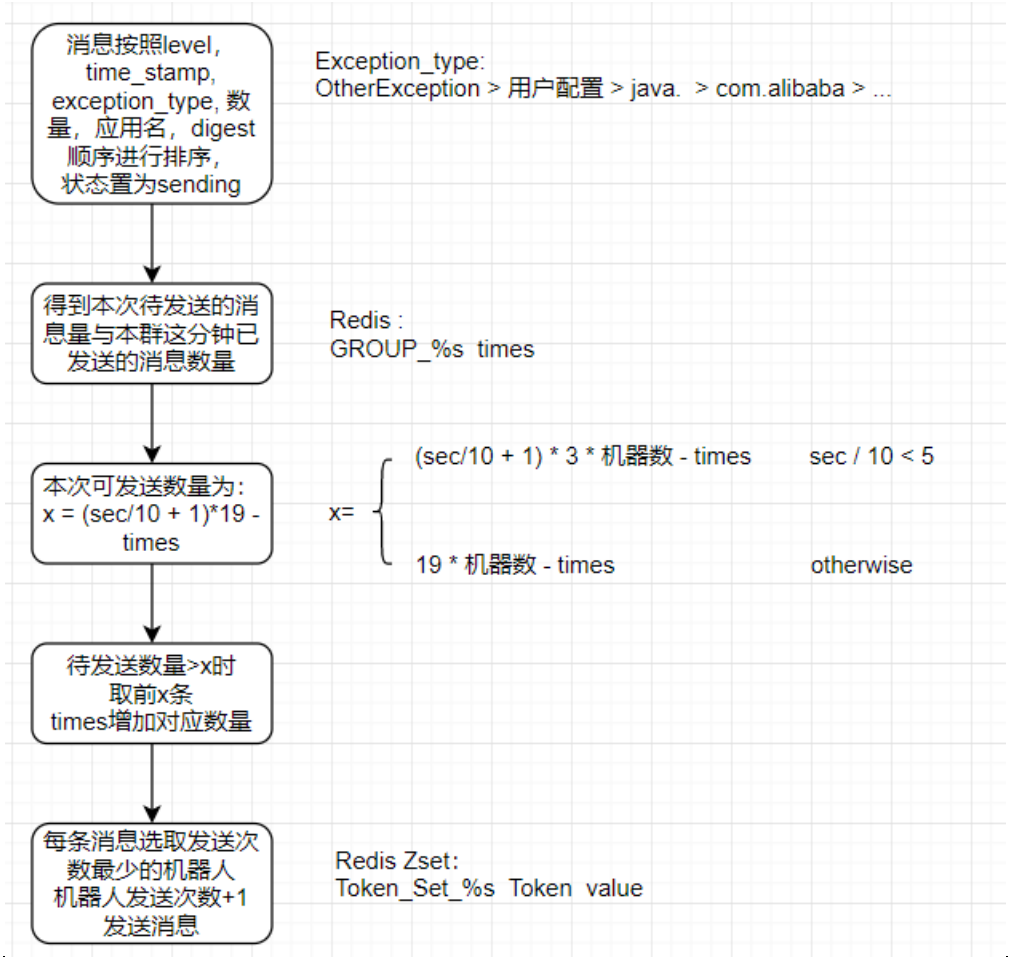
$$x=\begin{cases}(sec/10+1)*3*\text{机器数}-times & sec/10<5\\ 19*\text{机器数}-times & otherwise\end{cases}$$

这样子假设在50s前没有请求，50s时来了100个同一个群内容不同的消息也能够发送完全。

redis中还维护了每个群中机器人的发送次数，每次发送时选取当前发送次数最少的机器人进行发送，同样每分钟清零。

当此时待发送的数量超过本次可发送的数量时，依照优先级只选取x条消息。

通过保障每个群每个机器人每分钟发送消息的最多数量，保证机器人不会被限流。



若此时待发送的数量<本次可发送的数量，那么多余的名额会被本时段的重试任务占用。

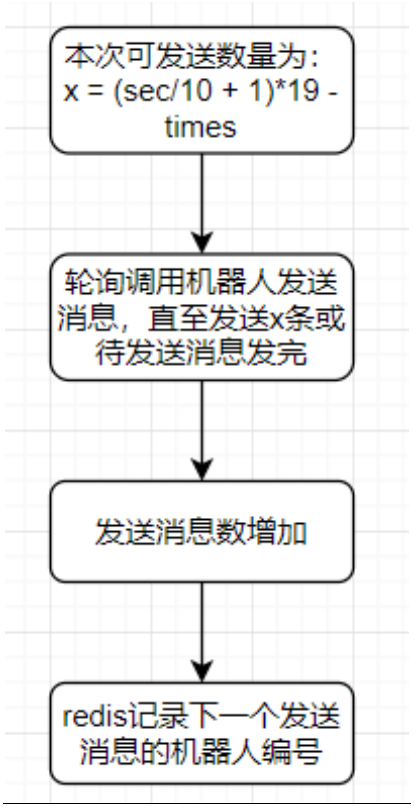
关于机器人的选择，目前考虑使用Redis维护Zset，里面分别维护机器人发送的次数。每次找到发送次数最少的。

可以按照机器人本次要发送的消息进行分组，通过多线性以机器人维度依次发送。相当于选取固定条数消息后按照机器人分组然后发送。

使用redis的原因是 服务部署在两台机器上，并且当前发送失败后不再往后延伸去新值。

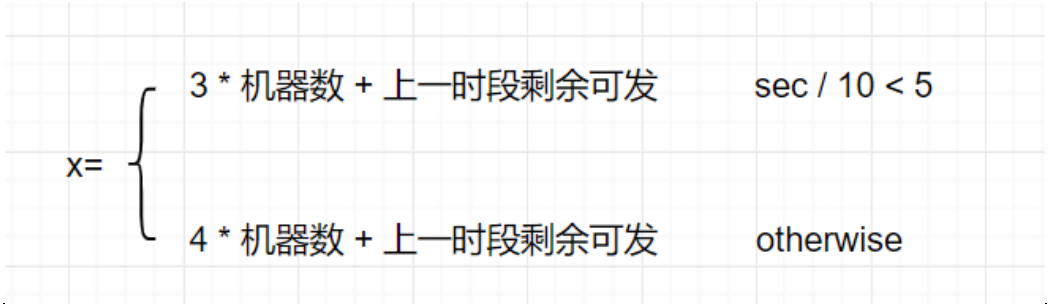
比如当前排序后30条消息，发送19条。若有 1条失败，会将机器人已发送消息与总发送消息减一，本时段的消息不会尝试抽出一个进行发送。

可以考虑进行同步发送，通过轮询的方式选取机器人。当该机器人发送失败时，继续使用该机器人。此时redis只需要维护当前机器人的编号。
``



~~当前排序后30条消息，发送19条。若有 1条失败（非最后一条），则继续使用该机器人发送下一条消息。这种情况下本次总共发送19条，而异步那里只发送18条。~~

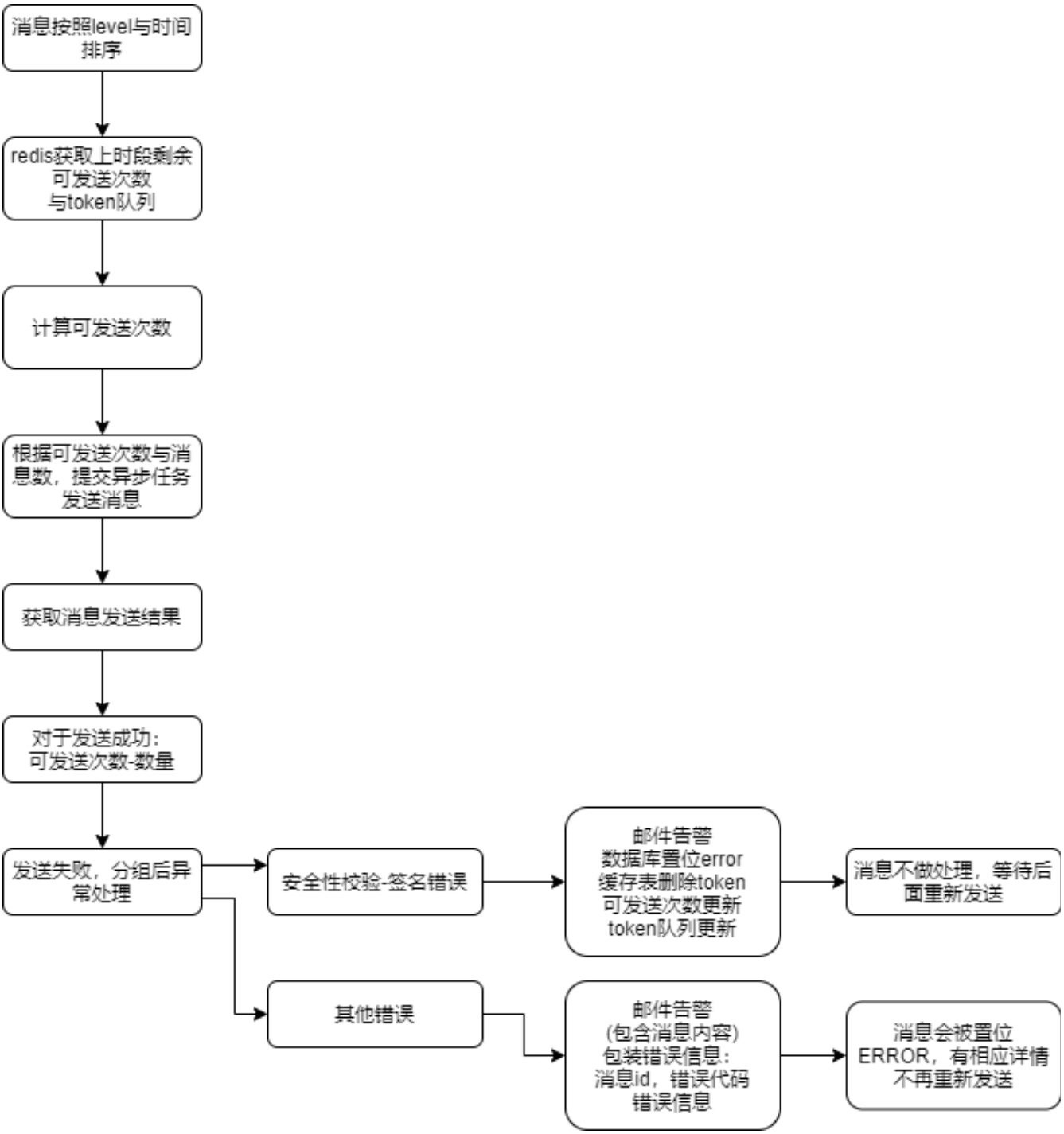
~~目前多线程发送，使用的方案是：~~



上面方案收到限流，托朋友在钉钉答疑群里面询问关于限流的原因，得到以下回答：

1. 网关限流统一通过tmd接入，但是tmd没人维护了，所以是不知道具体的限流规则，所以他也无法解释你测试的这个case
2. 后续要迁移到霸下，迁移后需要重新测试，暂时没有迁移计划
3. 对现在的限流建议仍然是每分钟不要超过20个，这个值是不是绝对安全的，答疑同学也没有明说，只是说通过内网访问的话很多都有白名单，公网限制比较严格。

于是关于发送次数，首先减少到单数轮发送2条，双数轮发送3条。每分钟单个机器人发送15条。经测试不会被限流。
对于突发的大量请求，对于不同的轮此设置相应的最大值。不知为何同样的策略两天测试中得到了不同的结果，于是目前保守设置每个机器人没轮次最多发送4条。

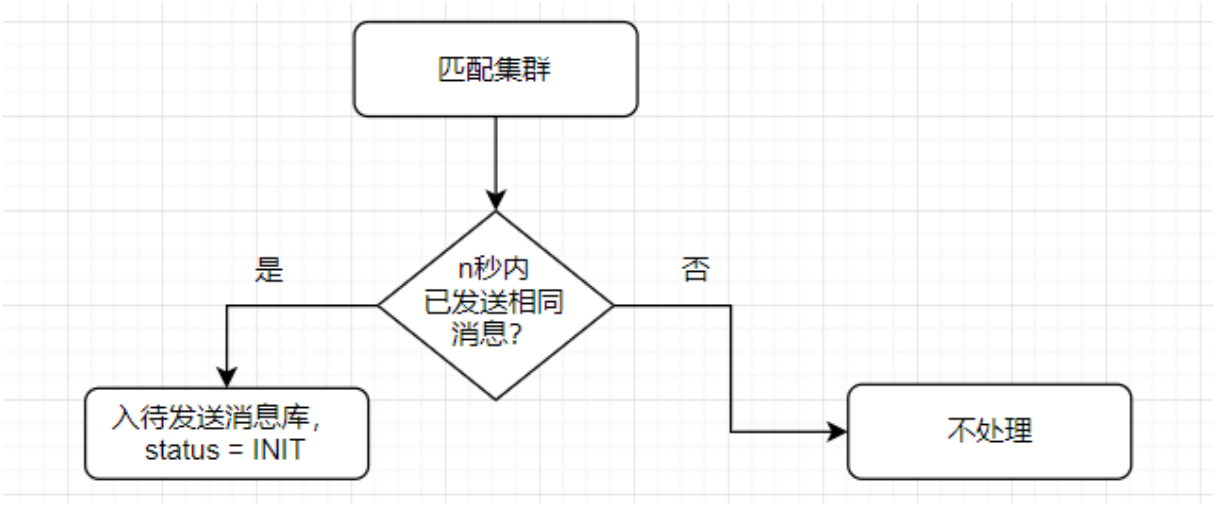


3、消息的自动分流：

目的是通过配置规则，使匹配的消息能够在期望的群中报出。

目前匹配规则考虑了两种形式：匹配规则

得到群号之后，会在待发送表中查找n秒（60s）内是否该群已经发过了相同的消息。如果已经发送过相同的消息，那么本次就不进行发送。



然后将相关信息包括群号放入待发送消息库，等待定时任务获取并调用机器人发送。

4、重要消息在大流量请求下的优先发送：

为消息设置了low、medium、high三个等级，不填写默认都是medium等级的消息.。

优先级越高，表示在大流量场景下越想及早的展示。

比如某一时段某一个群要发送100条不同的钉钉消息，而此时根据计算可发送的消息数量只有30。会按照优先级顺序进行排序，HIGH等级的消息将优先发送。

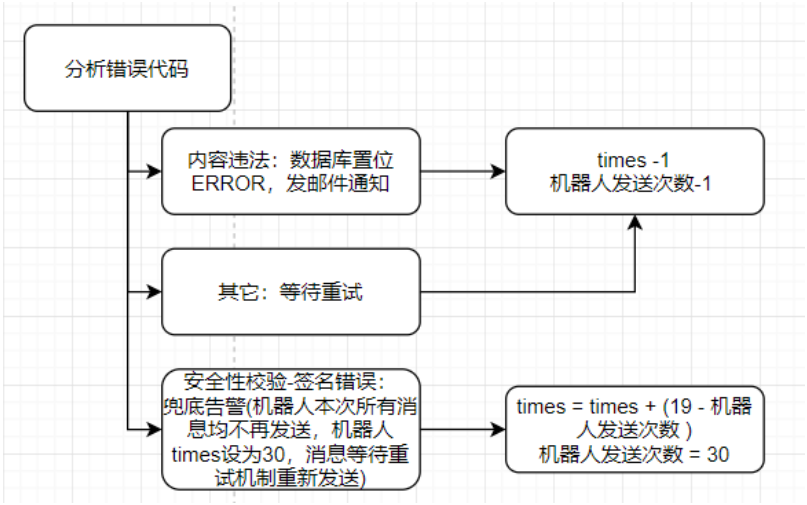
对于同等级的消息截取，按照异常类型与时间在进行进一步的排序。

5、发送失败时的处理：

发送消息可能会出现失败的情况，根据返回代码的不同做不同的处理：

errorCode	errorMsg	能否避免	动作
300001	参数错误		若是机器人配置问题， 那么Redis中将该机器人权重调整到20
101002	内容太长	通过前期校验避免	
130101	发送太快	通过发送测率避免	
300004	内容不合法		邮件告警，对应数据置位ERROR
1001	系统错误		邮件告警，消息等待重新发送
310000	安全校验错误（针对设置了安全设置的机器人）		签名错误： 邮件告警，通过设置机器人发送数量冻结机器人（可改进），消息等待重发

由于机器人发送失败，消息并没有发出去，并不影响发消息的次数，所以对应的已发送次数较少1。



使用redis维护主要原因担心服务起在多个机器上，如果只一个机器，可以在本地维护一张表就OK，机器人使用轮询策略。

目前配合多线程发送，异常流程在该图中有体现。

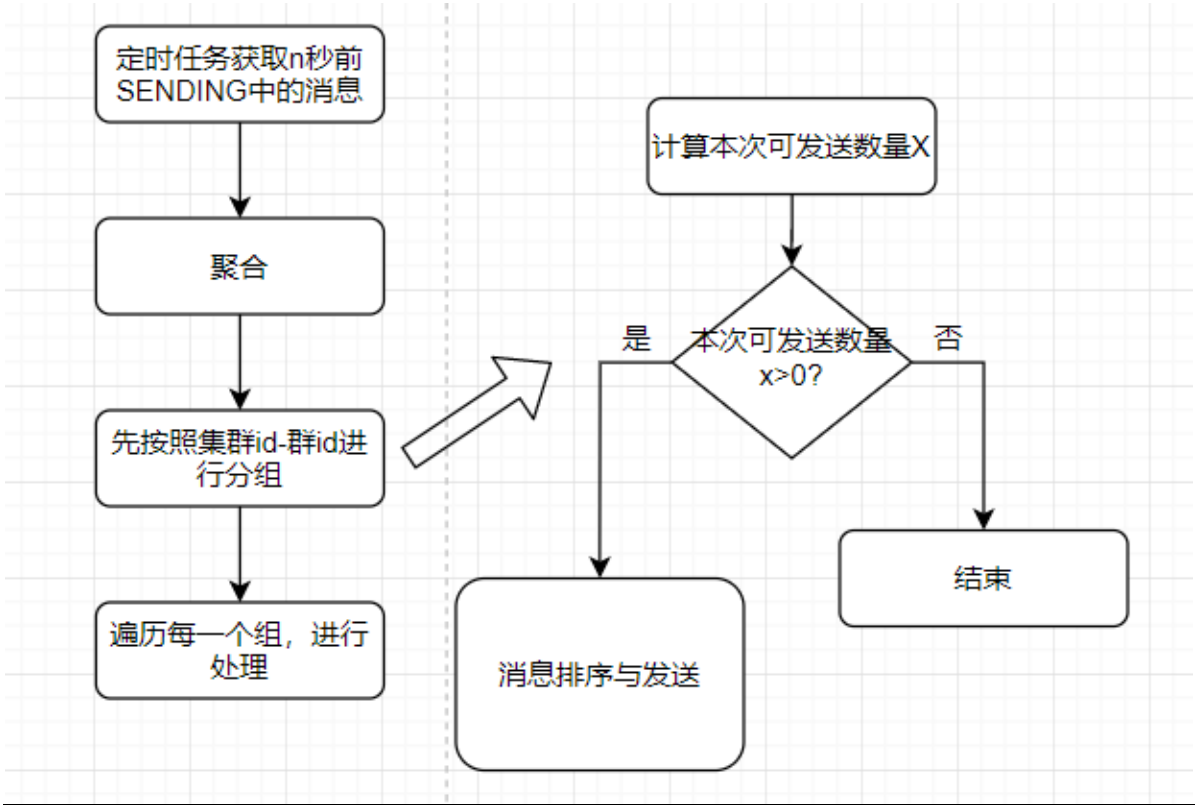
增加了对与限流的判断，如果机器人被限流，加入黑名单3分钟。（目前黑名单通过数据库字段实现）

6、重试机制：

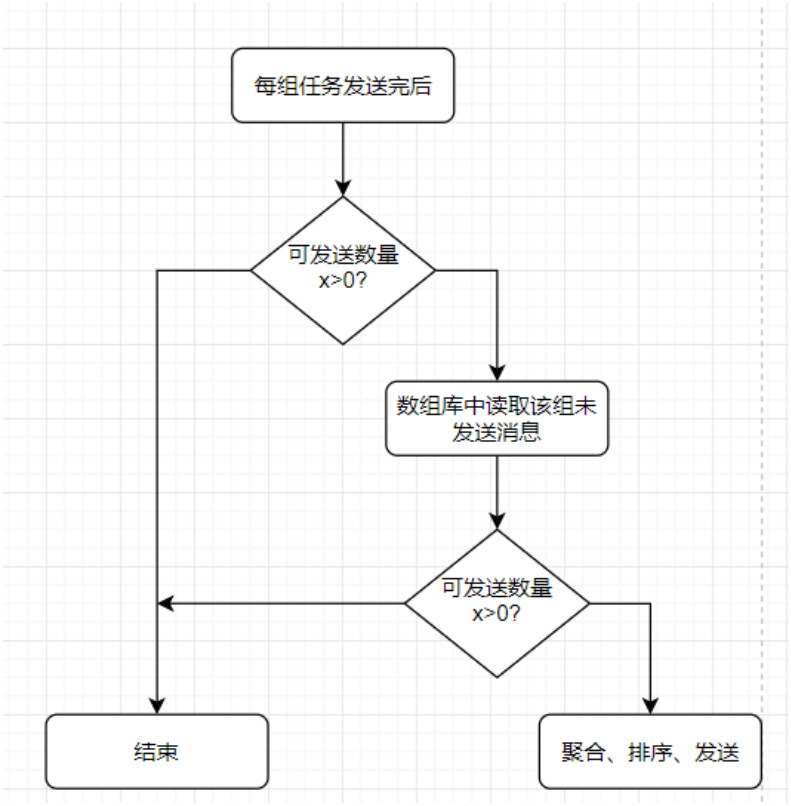
对于发送失败等待重新发送的消息、以及由于消息太多没有发送的消息，通过重发的形式尝试将消息发送出来。

比如5s时要发30条信息，根据选择只发发送了19条，剩余11条就会等待重新发送。在下个区间待发送4条信息，发送完毕后当前时段仍有15条可以发送的数量。

于是乎本时段的重试机制会找上个时段及之前没有发送出去的消息，同样地进行聚合与排序，选出一定数量的消息进行发送。发送逻辑与失败逻辑与之前所述相同。



重试可以考虑不单独写成定时任务，在正常发送流程结束后，该时段仍有可发送消息的名额，调用重试任务。

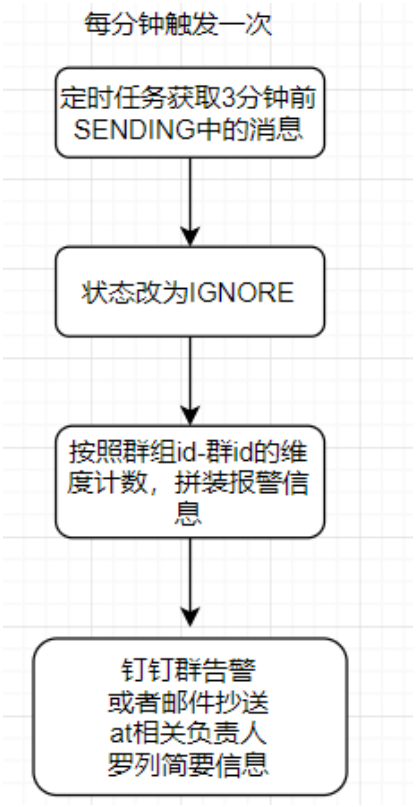


目前在发送流程结束后，如果存在可发送消息次数，触发重试机制。

7、忽略：

当一个群待消息过于饱和时，有一些消息可能经过很久也不会发出去。对于这类消息，存在一个忽略消息。

定时任务，查找三分钟前还未发送出的消息，根据群组进行分类，使用邮件或者钉钉群的方式进行报警，并将忽略的数量，以及忽略内容与相关信息做一个简要的处理合并发出来。



8、消息触达：

当调用接口发送消息时，经过匹配后最终决定将消息发送到某一群后，首先会通过该时段定时任务尝试发出。

1、正常情况下，该时段可发送消息数量>待发送消息数，消息会被及时发出。

2、当该时段可发送消息数<待发送消息数时，如若消息排序后排序在前，则会被发出。

3、本时间段未发送出时，之后的时段若重试任务可以发送消息且该消息排序靠前被选择到，则消息在重试中发出。

4、如果超过三分钟仍未发出，忽略动作中会将该消息与其它三分钟前未发出的消息进行一定的包装处理一并发出（邮件或者钉钉），此时消息内容会简略。

9、异常日志监控接入点：

现在异常日志监控流程可见：异常日志监控方案

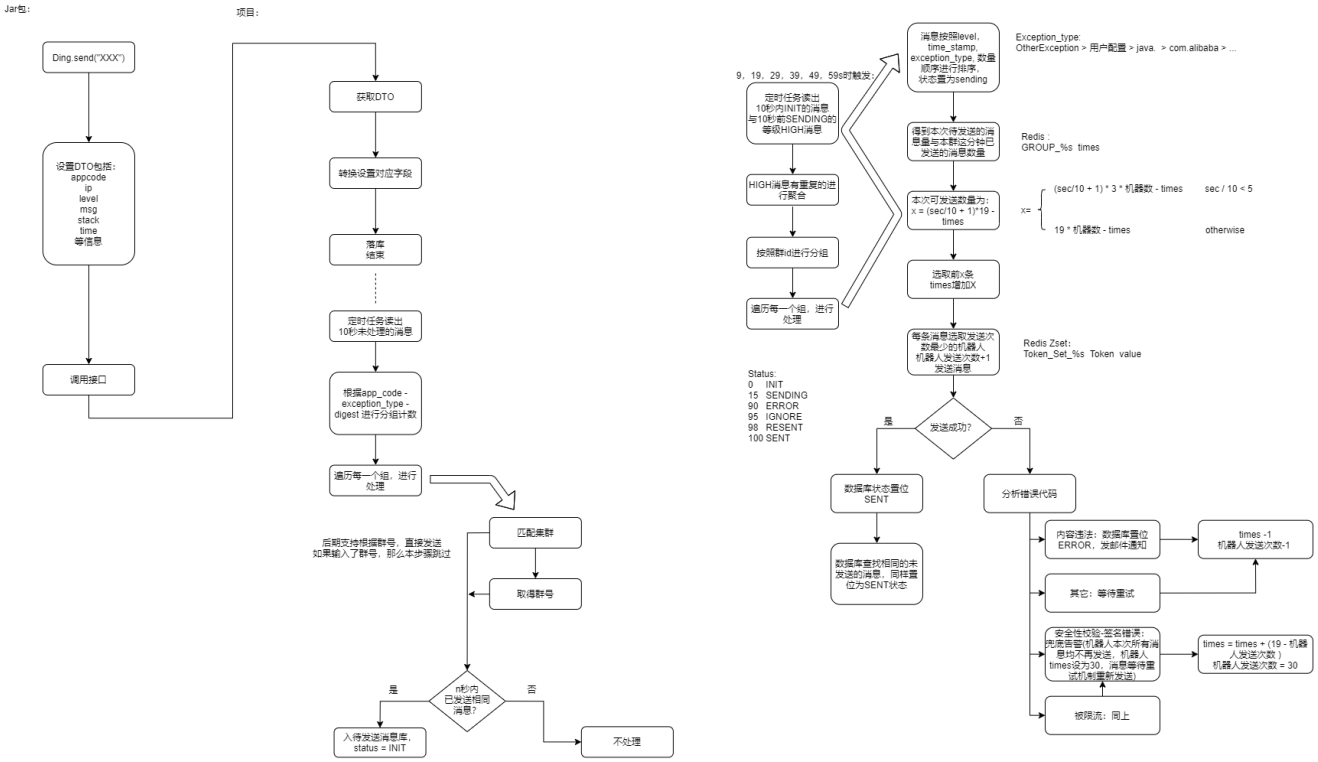
异常日志监控中生成告警信息后分组发送时调用接口。

接口将AlertRecord转变为对应的字段入待分组库，后续根据流程再进行群组的匹配、发送等后续动作。

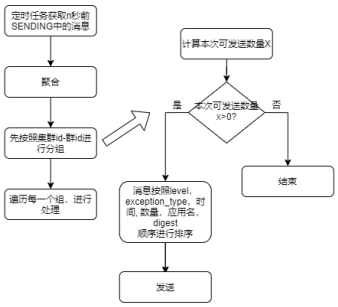
其中，当消息数量>20条时，异常日志监控系统会进行抛弃操作，只处理前5条消息。

修改后可以考虑去掉异常日志监控系统这条逻辑，因为在服务中也有类似的舍弃机制。

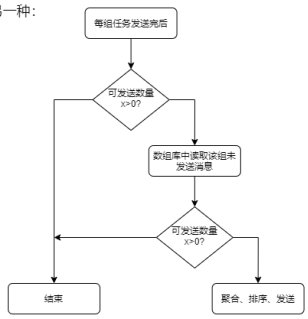
五、详细流程图：



重试



另一种：



舍弃：

统计：

每天一次定时任务，扫描并统计
消息重发次数
消息舍弃次数
消息错误次数与错误原因

思考：

1、有必要将聚合匹配与发送写成两个定时任务吗

目前每条消息匹配是多线程进行匹配、入库。发送时定时任务获取到信息后根据群号的维度进行处理。

如果写成一个任务，那就是多线程匹配→ 等待完成→ 所有消息按照群号分组→ 发送。 然后需要另一个重试的自动任务发送未发送的消息，仍然是两个定时任务。

2、使用部门作为群组分类时存在一个程序多个owner多个部门，以什么为主

默认群组流程：