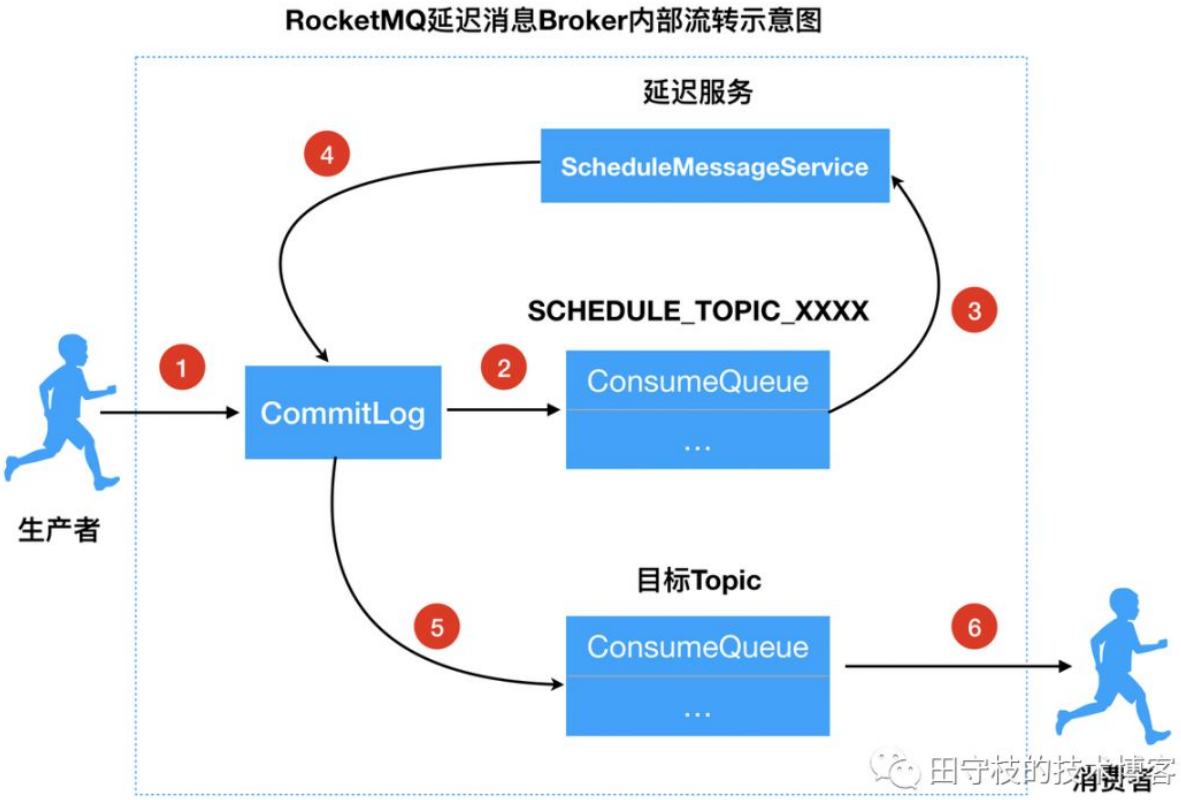
RocketMQ发送消息之延迟消息

延迟消息

延迟消息是指消息发送后，消费者要在一定时间后，或者指定某个时间点才可以消费。在没有延迟消息时，基本的做法是基于定时计划任务调度，定时发送消息。在 RocketMQ中只需要在发送消息时设置延迟级别即可实现。

```
1s 5s 10s 30s 1m 2m 3m 4m 5m 6m 7m 8m 9m 10m 20m 30m 1h 2h
```

Broker在启动时，内部会创建一个内部主题：SCHEDULE_TOPIC_XXXX，根据延迟level的个数，创建对应数量的队列，也就是说18个level对应了18个队列。注意，这并不是说这个内部主题只会有18个队列，因为Broker通常是集群模式部署的，因此每个节点都有18个队列。



1. 修改消息Topic名称和队列信息

RocketMQ Broker端在存储生产者写入的消息时，首先都会将其写入到CommitLog中。之后将Topic的名称修改为SCHEDULE_TOPIC_XXXX，并根据延迟级别确定要投递到哪个队列下。

2. 转发消息到延迟主题的CosumeQueue中

CommitLog中的消息转发到CosumeQueue中是异步进行的。在转发过程中，会对延迟消息进行特殊处理，主要是计算这条延迟消息需要在什么时候进行投递。

3. 延迟服务消费SCHEDULE_TOPIC_XXXX消息

Broker内部有一个ScheduleMessageService类，其充当延迟服务，消费SCHEDULE_TOPIC_XXXX中的消息，并投递到目标Topic中。ScheduleMessageService在启动

时，其会创建一个定时器Timer，并根据延迟级别的个数，启动对应数量的TimerTask，每个TimerTask负责一个延迟级别的消费与投递。

4. 将信息重新存储到CommitLog中

在将消息到期后，需要投递到目标Topic。由于在第一步已经记录了原来的Topic和队列信息，因此这里重新设置，再存储到CommitLog即可。此外，由于之前Message Tag HashCode字段存储的是消息的投递时间，这里需要重新计算tag的哈希值后再存储

5. 将消息投递到目标Topic中

这一步与第二步类似，不过由于消息的Topic名称已经改为了目标Topic。因此消息会直接投递到目标Topic的ConsumeQueue中，之后消费者即消费到这条消息。

6. 消费者消费目标topic中的数据

```
public class MessageType3 {
    public static void main(String[] args) {
        DefaultMQProducer producer = new
DefaultMQProducer("ProducerGroupName");
        producer.setNamesrvAddr("192.168.31.103:9876");
        producer.setRetryTimesWhenSendAsyncFailed(2);
        try {
            producer.start();
            long id = 4466l;

            String data = "{\"id\":" + id+" , + \"title\":\"X市2021年度第四季度税务汇总数据
\"}";
            Message message = new Message("tax-data", "2021S4",
data.getBytes(RemotingHelper.DEFAULT_CHARSET));
            //1s 5s 10s 30s 1m 2m 3m 4m 5m 6m 7m 8m 9m 10m 20m 30m 1h 2h
            message.setDelayTimeLevel(5);
            SendResult result = producer.send(message);
            System.out.println("消息已发送：MsgId:" + result.getMsgId() + "，发送状态:"
+ result.getSendStatus());
        }catch (Exception e){
            e.printStackTrace();
        }finally {
            try {
                producer.shutdown();
                System.out.println("连接已关闭");
            }catch (Exception e){
                e.printStackTrace();
            }
        }
    }
}
```

修改延时配置

single-master.properties

#集群名称，同一个集群下的broker要求统一
brokerClusterName=DefaultCluster
#broker名称
brokerName=broker-a
...
messageDelayLevel=90s 5s 10s 30s 1m 2m 3m 4m 5m 6m 7m 8m 9m 10m 20m 30m 1h 2h