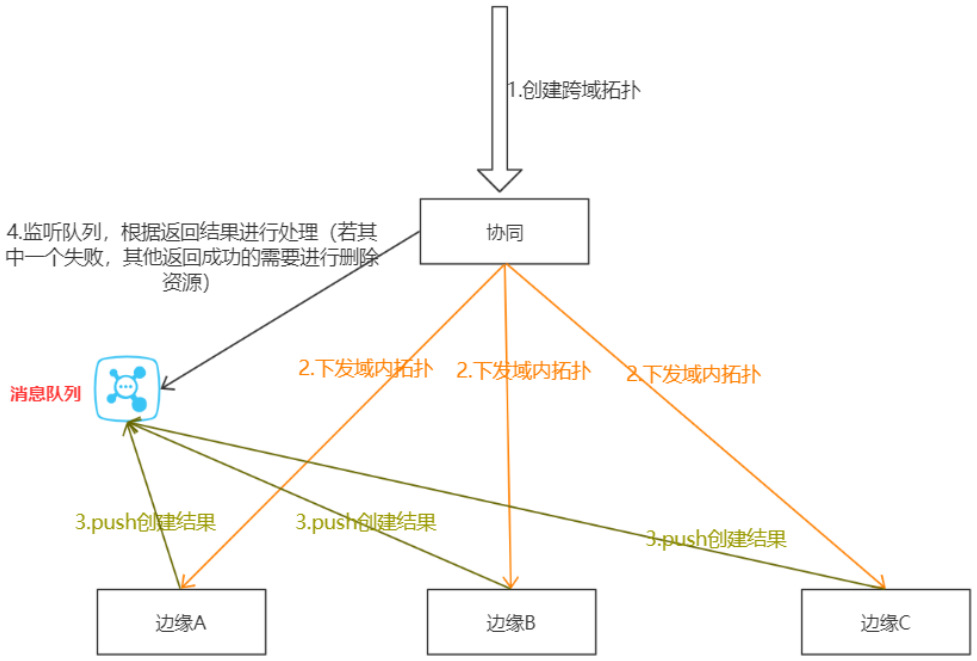
消息队列引入后的消息机制（以创建为例）

192.168.(



消息队列引入可能遇到的问题?

1.创建失败怎么办?

A成功B成功C成功：拓扑成功

A失败B成功C成功：拓扑置为创建失败，收到B、C成功后即刻删除B、C（删除失败记录告警）

A成功B失败C成功：拓扑置为创建失败，立即删除C，收到B成功后即刻删除B（删除失败记录告警）

A成功B成功C失败：拓扑置为创建失败，立即删除B、C（删除失败记录告警）

2.创建失败，上层用户立马删除怎么办?

如果置为创建失败，用户可能会立即删除，打断上述任意一步：

处理思路例如：A失败（用户删除）B成功C成功：拓扑置为删除中，收到B、C成功后即刻删除B、C（删除失败记录告警），拓扑置为删除成功

3.协同迟迟拉取不到边缘创建/删除的结果信息怎么办?

边缘和协同均设置拓扑超时机制，确保拓扑不会总是处于创建中/删除中的状态

4.消息队列挂掉怎么办?

协同采取类似心跳机制来保证与消息队列的通信，在创建和删除之前，均会进行检查，如果消息队列挂掉，则采用轮询边缘的方式进行，边缘创建/删除完成不会发往消息队列，防止轮询线程和消息队列监听线程同时拿到边缘创建的结果进行处理。

引入消息队列的优缺点?

优点

1.不需要每创建或删除拓扑都开启一个线程，如果是批量场景下，会造成资源耗费过大，严重甚至会导致线程耗尽

2.边缘侧减少了重复轮询接口调用的压力

3.拓扑能够进一步精细化处理，便于实现分布式事务

缺点

1.增加了系统的复杂性

2.降低了系统的可用性

3.需要考虑消息队列的持久化存储、集群部署（k8s环境下）