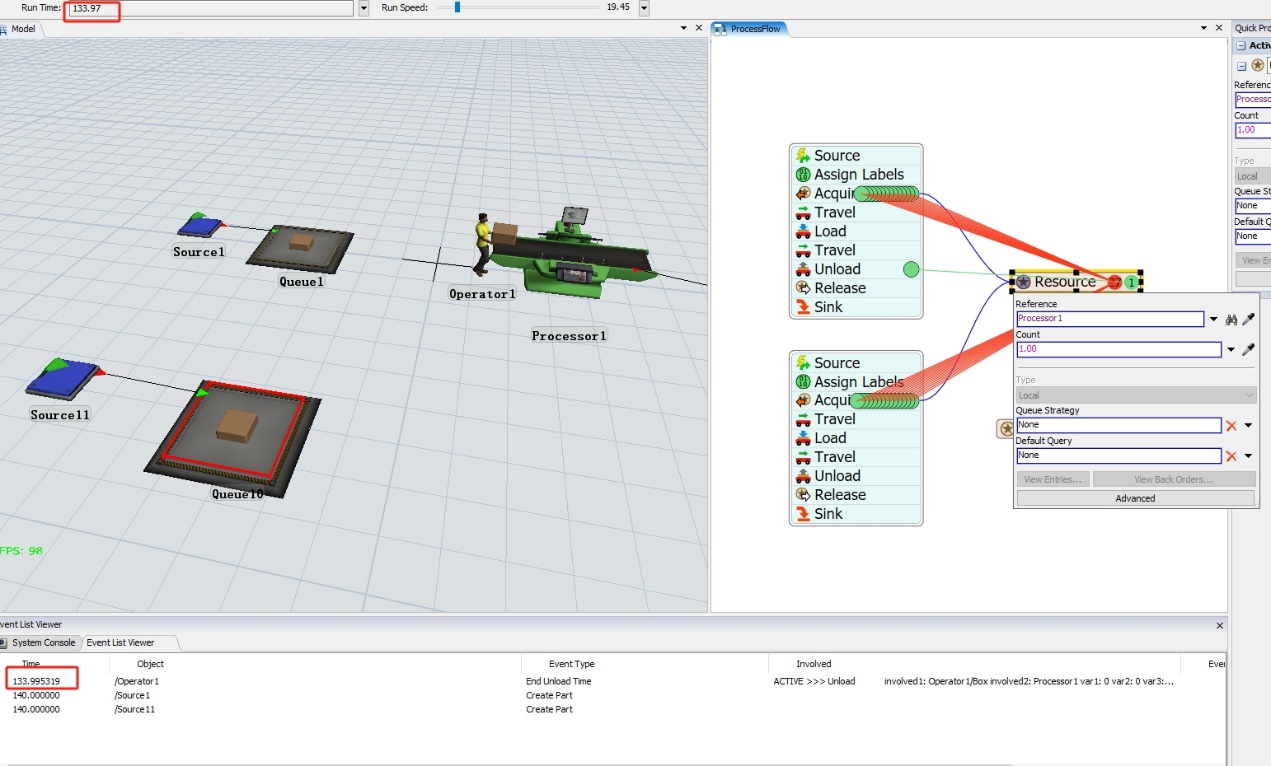
对有限资源获取时的逻辑总结

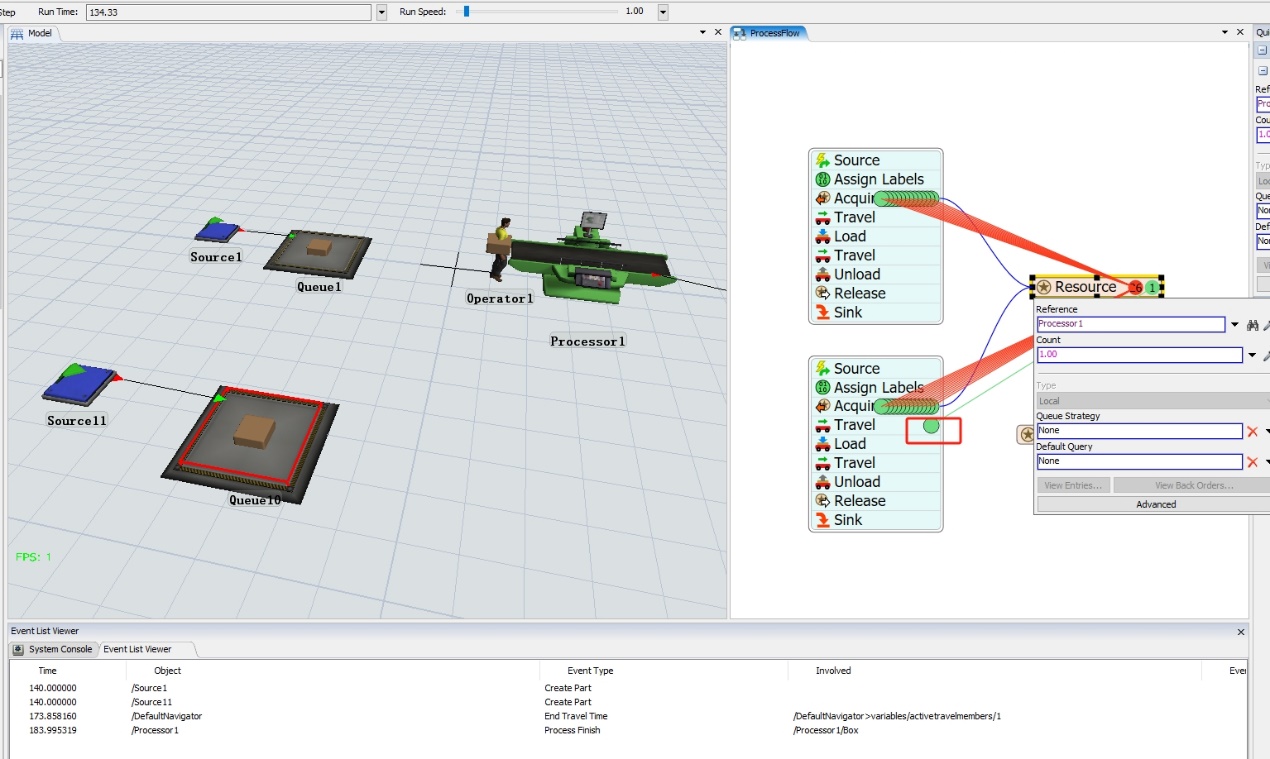
结论1： 当带条件查询时， 只会在事件触发下查询单次

结论2： 当查询时涉及计算distance时，只能使用Pull from list,这样才能传递Puller 对象才会让distance 计算成功

使用 Resource 没有query， 会一直查询, 有就给



在系统时间133.995319时刻，Operator结束Unload任务（而后释放资源Processor），

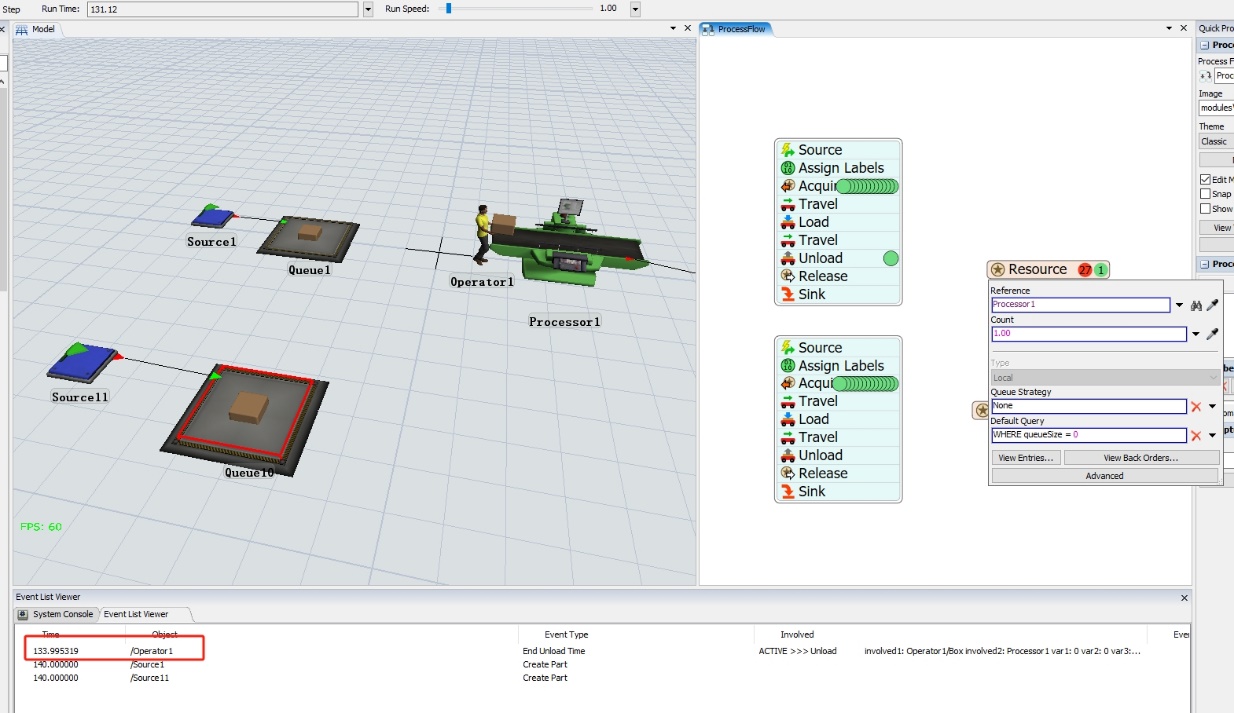


在系统时间134.33时刻，观测到 Processor资源被其他token获取到

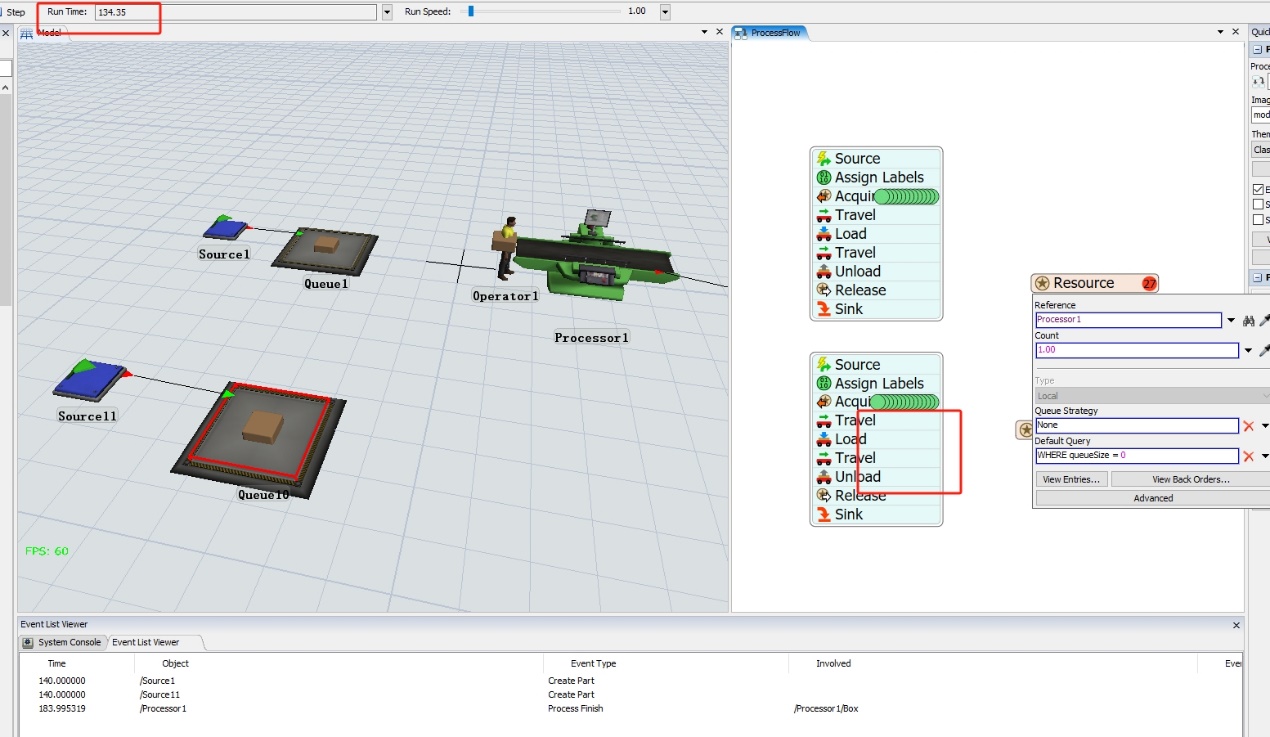
在133.995319 到140.之间没有其他事件发生

总结： 没有query，token一直在查询， 有一个给一个

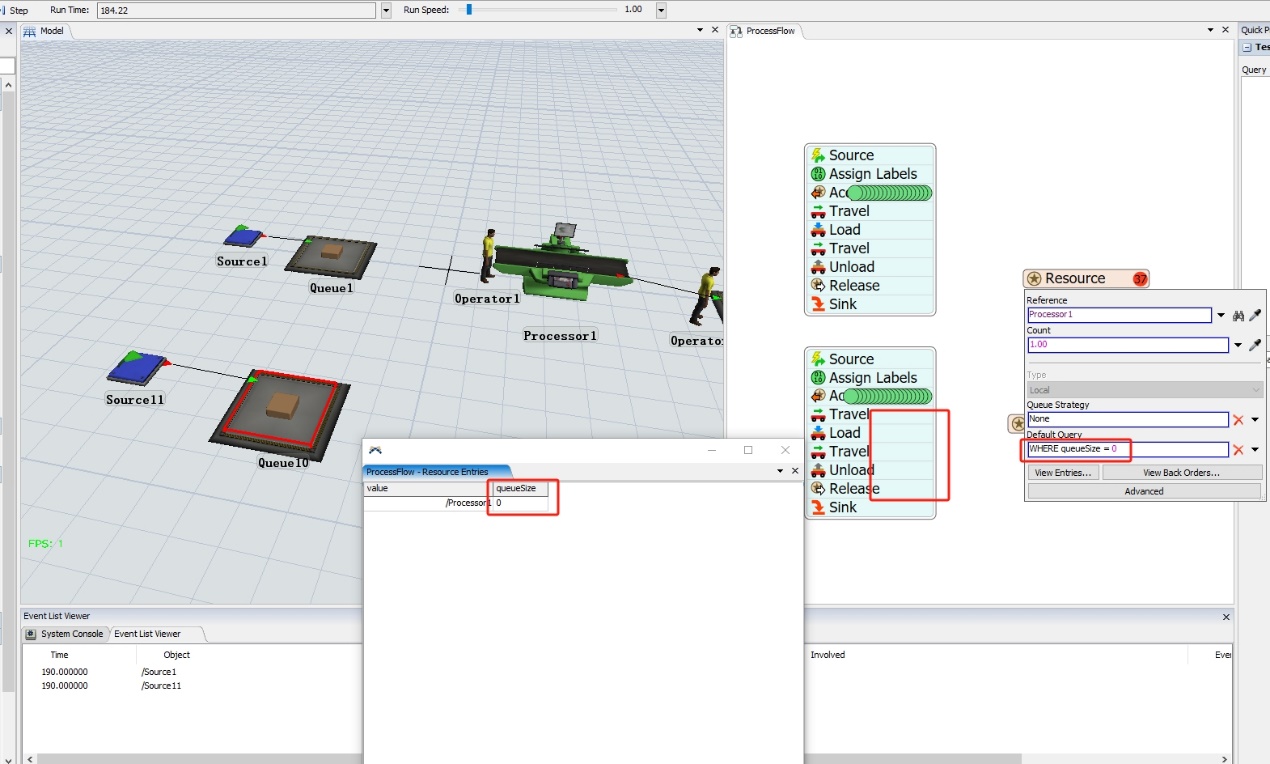
有query，token只会在事件触发下查询一次， 其余事件不会，条件满足也不会查询



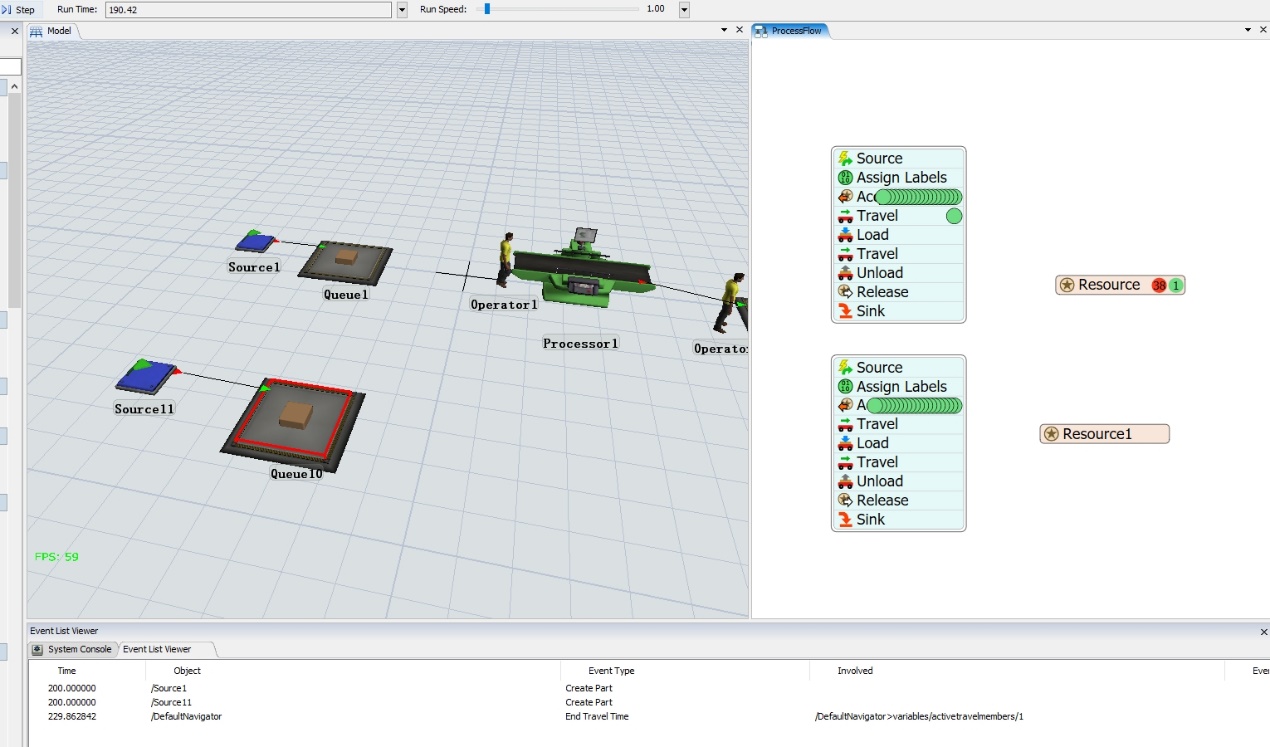
在系统时间133.995319时刻，Operator结束Unload任务（而后释放资源Processor），



在134.35， token没有资源被获取到，符合预期（query条件不满足，当前queue Size= 1）



184.22观察，在条件满足且有资源后没有给，

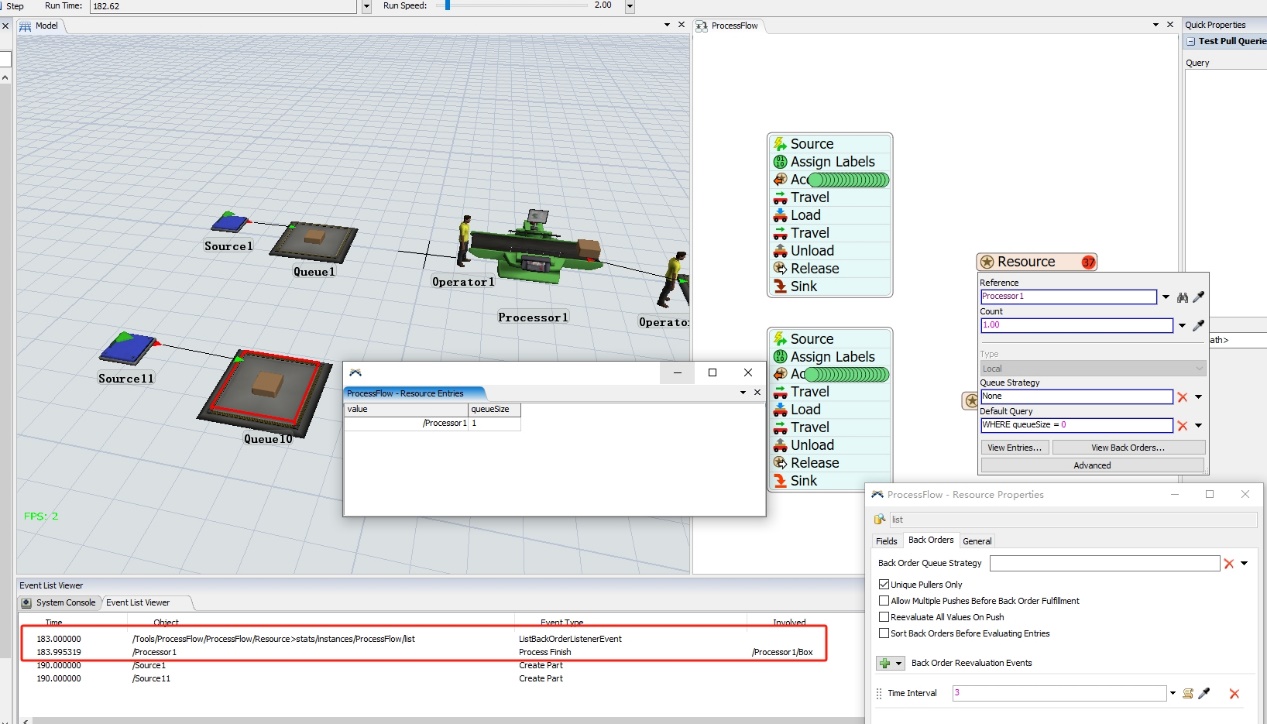


等待了一段时间

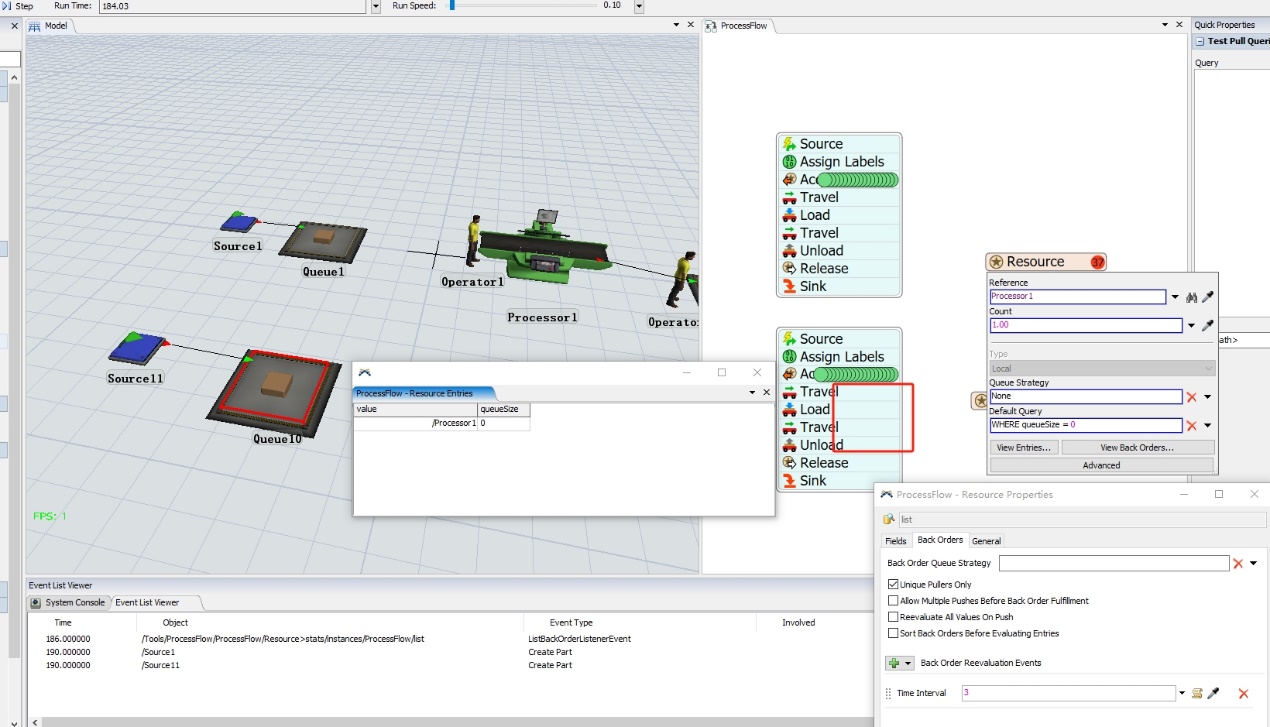
在190.42观测到，给上一个查询的

可能原因：下一个查询的token到达activety（时间：190.00000），让所有的token都刷新又查询了一次

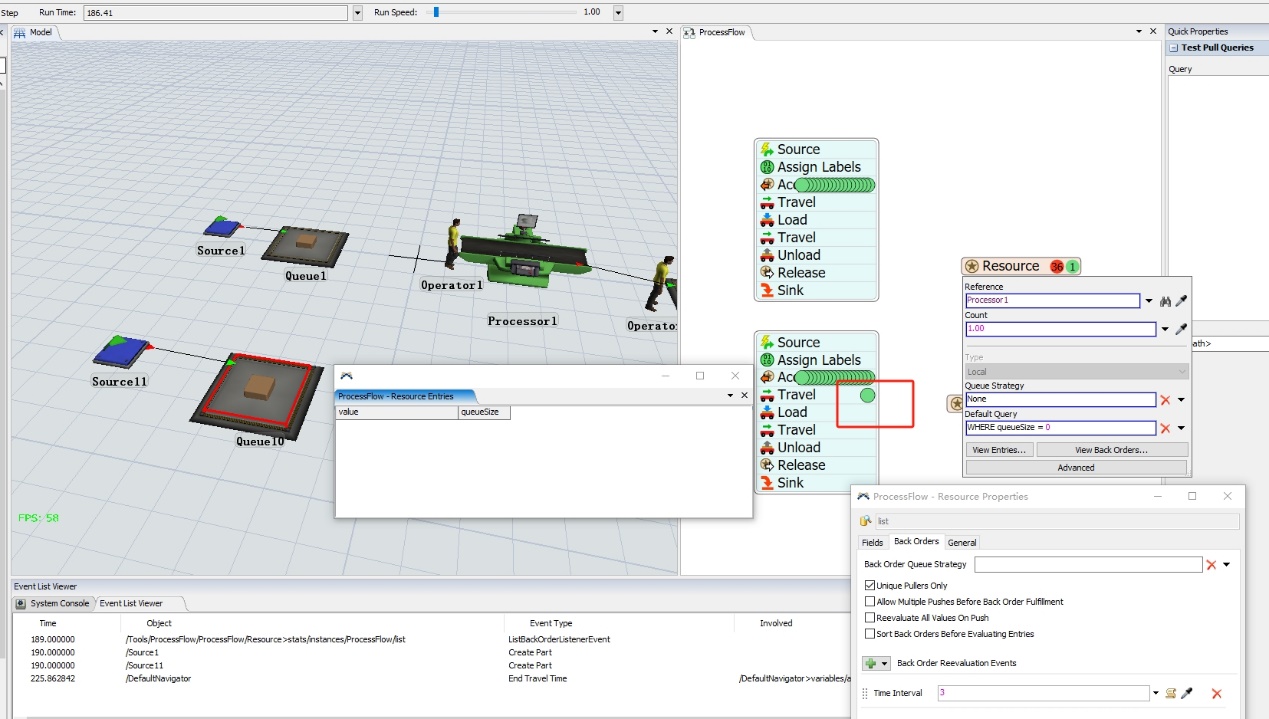
带query申请，在back orders设置定时刷新事件，



系统将在183.99 处理器加工完成， query条件将会满足



在183.03观测到 条件满足，有资源还是没给



在186.41观测到 条件满足，资源给了后续token， 设置的定时事件在186.00触发， 让所有token都重新查询了一边

总结： 带query申请时， resource内部处理成List， 不会持续查询， 只会在事件（进入、离开、定时等）下查询一次，

对fulfilled 字段理解

TODO

勾选 Allow Multiple Pushes Before Back Order Fulfillment

对上述的影响

带query 无定时 还是后一个token进去才触发