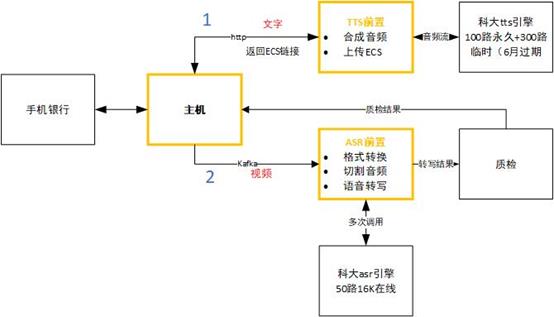
1. 上游发起TTS请求，将文本话术转语音后向客户播报。
2. 收到完整语音记录后， 调用ASR将录音转化为文字向下游质检模块发送。

整个流程是事后质检，即所有交互动作完成之后进行质检。



原调用链路，Client -> tts frontier -> 科大TTS（500路）

kafka -> asr frontier -> 科大ASR

存在问题：

1. TTS前置机会OOM
2. ASR前置机部署60 pod只有一个pod有流量
3. TTS OOM问题， 由于时间紧迫，改为kafka异步，并限制内部向科大请求的线程数来完成。 由于认为是向科大发起请求时占用的内存过多， 所以思路为限制单机同时调用科大tts服务的线程数量。 后证明问题不在这里。
4. 当过了抢购时间后，TTS前置机服务移交给我， 使用Jmeter压测并使用VisualVM监控内存情况， 发现有大量byte[]内存占用， 而tts中正需要大量byte[]来缓存音频文件， 再进一步确认文件大小和压测线程后，发现byte[]数量远远超过预估数量。 寻找其他原因， 在抽样器中发现tomcat http-nio线程单个可达100M-400M，内存占用极大。 检查应用配置，发现有server.maxHttpHeaderSize=102400000, server.maxHttpPostSize=102400000。 取值远远大于需求文本长度(约千字左右)。 将其值改小， 并在下一迭代中将请求文本放入body中而不再使用requestParam的方式。
5. 对于ASR流量不均的问题， 首先怀疑是kafka partion数量过少， 经检查申请的资源状态后发现只有一个partition，并且消费逻辑中单笔请求asr（一个kafka record可能会调用多次asr服务）。 切换到更多partition的topic，并将接收kafka数据与请求逻辑解耦完成并发改造。(consumer接收到数据后提交到线程池去请求asr服务)

改造后瓶颈在于科大资源数, tts tps 68, asr tps 10，看后续购买资源补充或用自研TTS/ASR替代。

由于购买流程在用户端也是异步的， 购买提交后在页面刷新查看结果，该处理速度已经能满足业务需求。

由于有许多模板话术， 所以在tts前置机增加了redis缓存， 可以命中大量不包含个人信息的文本， 如大量关于基金产品的介绍。 但个人觉得这个缓存与业务强相关， 应当由上游来做。

1. Tts asr改造

并发数相对来说并不高， 但由于下游处理时间较长， 所以可能存在消息堆积。 并且是IO请求阻塞，对于前置机来说CPU的使用率并不高， 所以改为异步处理，线程池线程数量总数与科大资源数量对齐， 上游增加超时机制来进行了进一步的改造。