web IM涉及的种类、功能、样式及技术

**1、什么是IM**

IM（Instant Messaging）即时通信，是一种通过网络进行实时通信的系统，允许两人或多人使用网络即时的传递文字消息、文件、语音与视频交流，通常以网站、软件或者移动app的方式提供服务。自从互联网的兴起，IM就一直和我们的生活息息相关，日常聊天、工作、打车、外卖、购物等等，可以说，我们现在的生活已经几乎离不开IM。

IM历史

最早人们的通信靠的是邮件，需要人去邮局寄信，然后邮递员再经过漫长的旅程送达对方。从邮件到传呼机再到有线电话，无线电话，最后随着互联网的发展IM迎来了它的新生。

最早的即时通信软件叫做ICQ，他是四名以色列青年于1996年7月成立的Mirabilis公司推出的产品。然后腾讯接着推出了OICQ。

**2、主流的IM应用**

Whatsapp 美国

Line 日本

Kakao Talk 韩国

WeChat、QQ、飞书 中国

Facebook Messenger 美国

**3、特性**

IM的四大特性，有效性、实时性、一致性以及安全性，那么如何实现一个可靠的Web IM应用呢？业界其实已有很成熟的IM的方案，我们会从通信协议、应用层、通信数据格式、以及相应策略方面给大家阐述一个可靠的IM应用都需要哪些东西。

**4、IM通信协议**

早期，IM采用的是http短轮询的模式，定期、高频地轮询服务器端信息。

它的缺点也很明显，会有大量无用的请求，用户端也会非常耗电耗流量，而服务端面对高频QPS，内存资源压力也会非常大

对于短轮询的优化，就出现了长轮询。相对短轮询，它大幅降低了无用轮询导致的网络与功耗开销，但是服务端悬挂住请求，只是降低了入口请求的QPS，并没有降低服务器的资源开销，假如有1000个请求在等待，那就意味着有1000个线程挂起，被轮询占用消息存储资源。

为了更好的解决实时性问题，IM领域经历过几次技术的迭代升级，从简单、低效的短轮询逐步升级到相对效率可控的长轮询，然后随着h5的出现，全双工（Full-duplex）的websocket彻底解决了服务端推送的问题。用户侧和服务端利用websocket建立长连接后，双方就可以同时进行双向的数据传输了。服务器的压力也不再是连接数，而是每一条消息事物。

**5、应用层可靠**

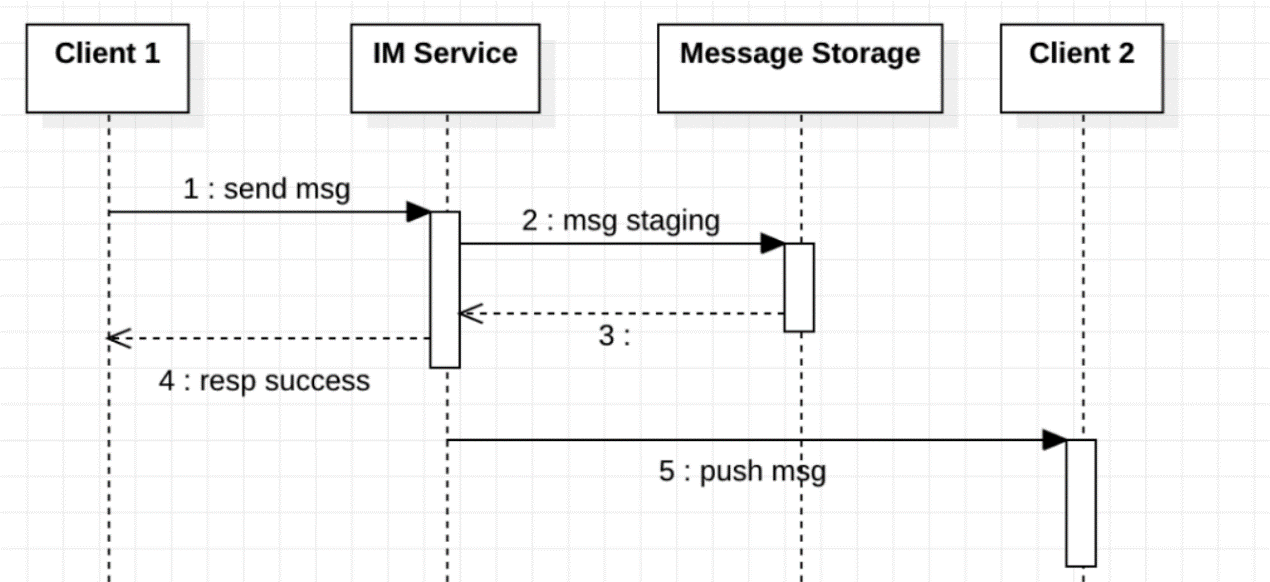
在底层协议的保障后，我们的消息就完全可靠了吗？那肯定不是，我们的服务大致是这样的，用户发送消息给服务端，服务端存储消息，再返回给用户，并把该条消息推给另一个用户，在下图流程中，每个环节都可能存在消息丢失的风险。

用户1发送到IM服务的过程中

IM服务器存储失败

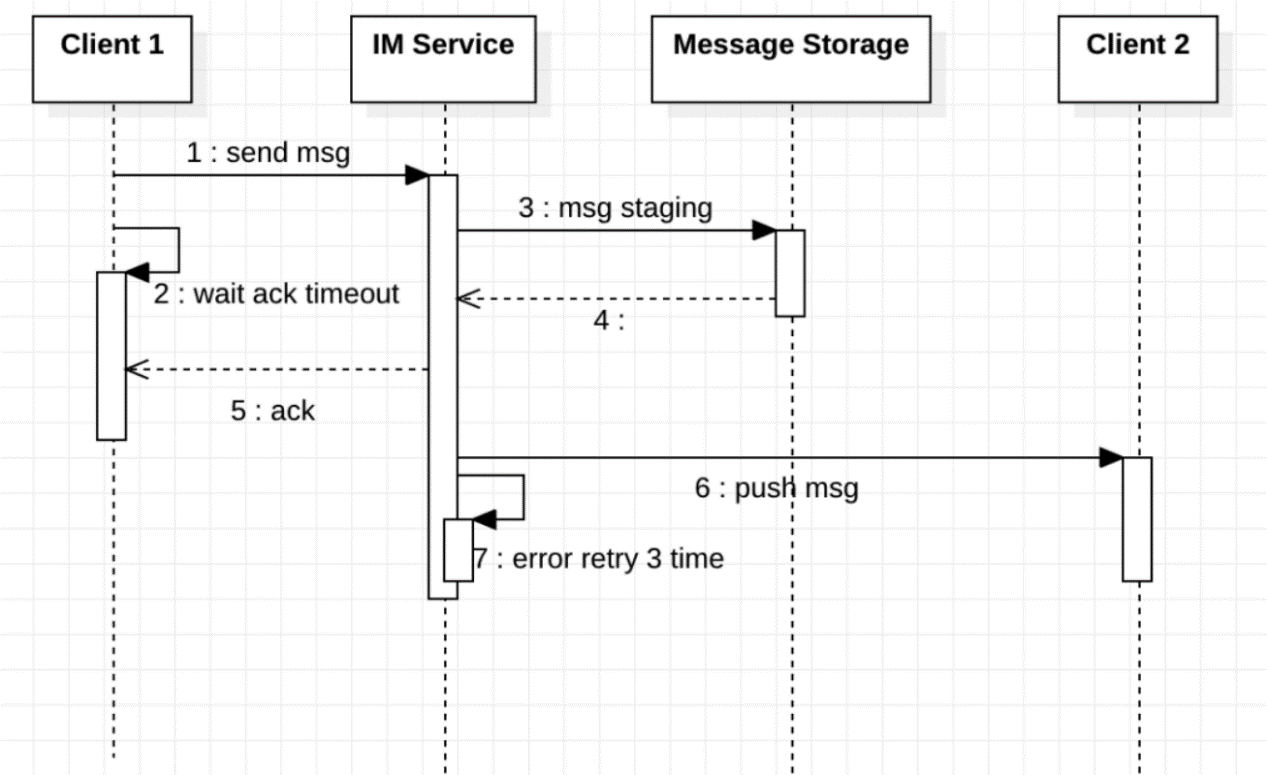
用户1等待服务器响应超时

服务器往用户2推送消息时超时、错误



解决办法：

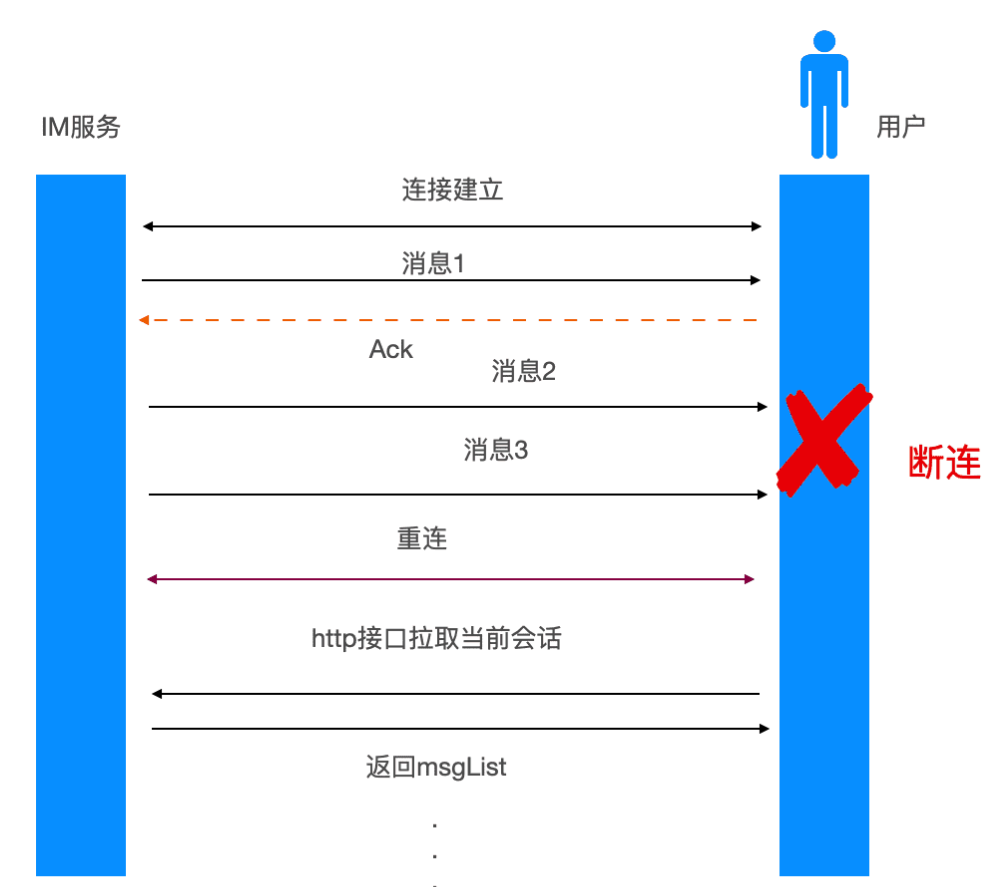
1. ack机制

为了解决用户1到服务端的可靠性问题，我们参考TCP协议的握手、重传机制，来保障应用层消息的可靠性，在发送消息后会有一个定时超时，在超时后根据需要，从ack队列中取出消息重推。一般情况显示发送失败，交由用户手动重发（比如消息左边一个红色感叹号）。用户发送消息，服务端收到消息后，生成该条消息的唯一id，以ack的形式回传给用户侧，用户侧再更新该条消息id值，后续IM功能中的撤回，去重 ，重发等逻辑都会用到该id。

1. 消息重传、去重机制

在服务端推送消息时，如出错或超时，会有相应的重发机制。比如，设置错误或超时重试三次。有时因为一些网络或其他情况。服务端会有相应的重发逻辑，在推送消息出现重发时，用户端设置对应的去重逻辑。我们会对消息列表的最新的5条消息进行排序和去重。只取最新5条主要考虑到排序与去重的效率，用户的焦点主要在最新的几条消息，如果因为一些网络原因在消息列表较远处插入消息，会造成用户的困惑与遗漏，另外5条消息的时间差基本满足大多数异常情况的消息丢失场景。如果还有消息遗漏的情况，用户在刷新消息列表时会以http的形式拉取历史消息（当前会话的消息）。

1. http拉取历史信息

ack+超时重传+消息去重，能解决大部分消息推送丢失的问题，但比如服务器宕机，电脑手机息屏，手机切换后台等等造成连接断开，通道不可用。一般在这个时候，我们会在重连或者用户窗口可视时对消息进行完整性检查，会以http的形式拉取这段时间的消息，以最后一条消息的时间戳作为参数拉取这个时间段的消息，或者拉取后端会话（session）维度的消息。

1. 心跳机制

websocket的连接是无感知的虚拟连接，中间链路出现一些异常情况断开时两边不会感知到，为了保证服务的可靠性，以及降低服务器的开销，我们会有对应的心跳机制，来检测连接是否正常，从而保持连接高可用。

心跳可以用于重传，还可用于及时释放服务器以及业务资源，取决于IM的场景与策略。比如一些客服聊天场景，客服要尽可能接待更多的用户，为及时释放客服资源，服务端在用户达到固定未收到心跳时间，及时断开客服聊天，释放相应资源。除此之外，心跳还有连接保活的功能。有时会遇到NAT(Network Address Translator)超时的情况。运营商维护NAT映射表时，为了节约资源和降低 自身网关压力，会定时清除没有数据收发的连接，具体不在这里详情阐述。但这个过程服务端和用户端都无法感知，从而会影响消息收发。下图（图9）是一些运营商的NAT超时时间

常用心跳方案：TCP keepalive、应用层心跳、智能心跳

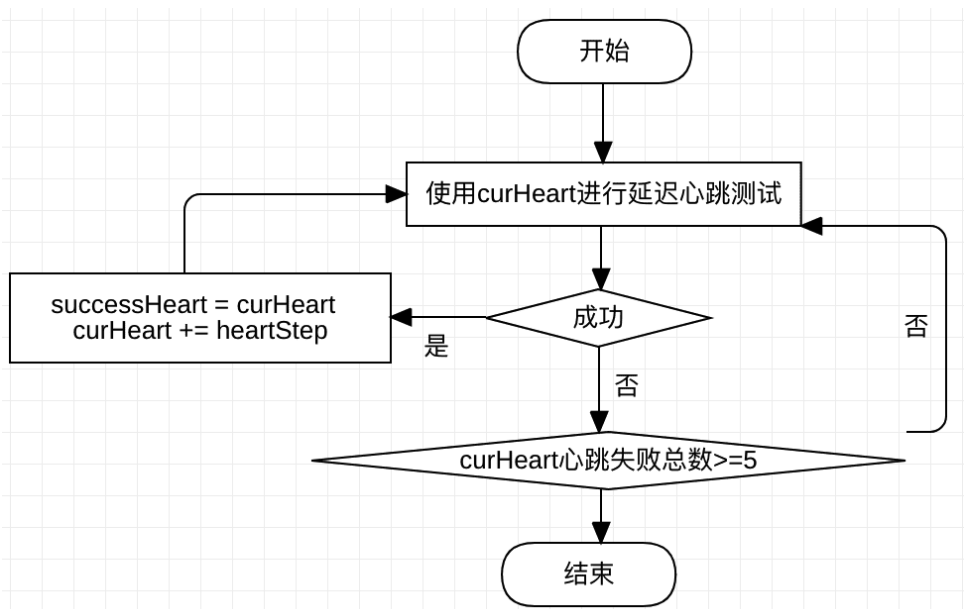
安卓版微信的智能心跳方案：

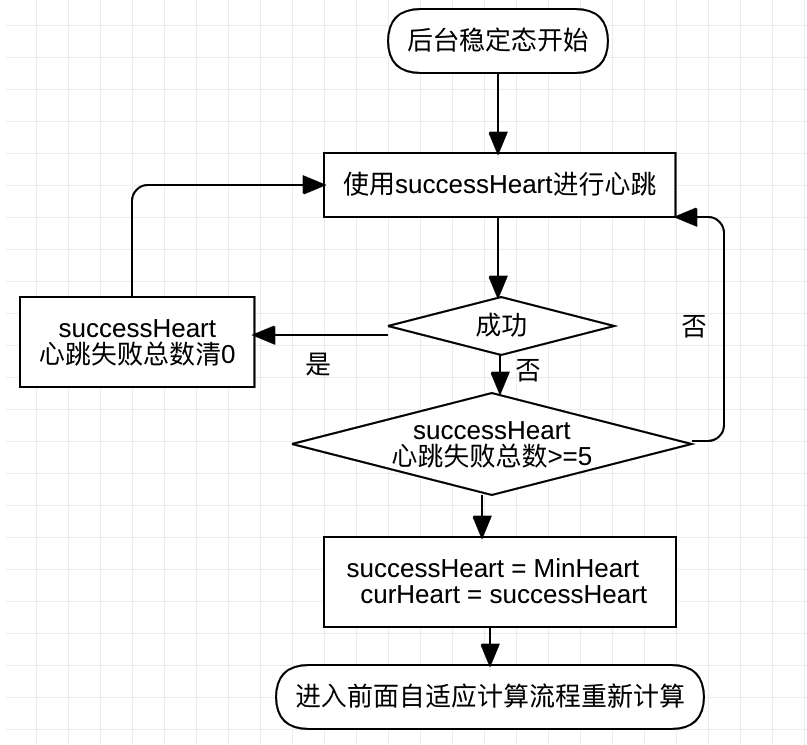
[MinHeat, MaxHeart]--心跳可选区间

successHeart--当前成功心跳

curHeart--当前心跳，初始值successHeart

heartStep—心跳增加步长

successStep—稳定期后的探测步长

1. 消息协议

业界常用的数据格式有以下：

XMPP

Protobuf（Protocol Buffer）

JSON

私有二进制

MQTT

定制化XML

一般我们会选择Protobuf，它是Google公司内部的混合语言数据标准，用pb序列化后的大小是json的10分之一，xml格式的20分之一，是二进制序列化的10分之一，并且基本上主流语言都已支持。不过考虑到上手的简单以及易调试，json也是不错的，明文的数据包在调试上也会方便许多。

**6、 V-IM开源IM介绍**

layIM（主要是聊天表情，文件处理方面）。

使用SpringBoot、oauth2.0、t-io 开发后端服务。

vue、iview 开发前端。

界面仿微信。

其他：使用 fetch 发送ajax 请求，支持跨域，electron 支持打包成为exe。