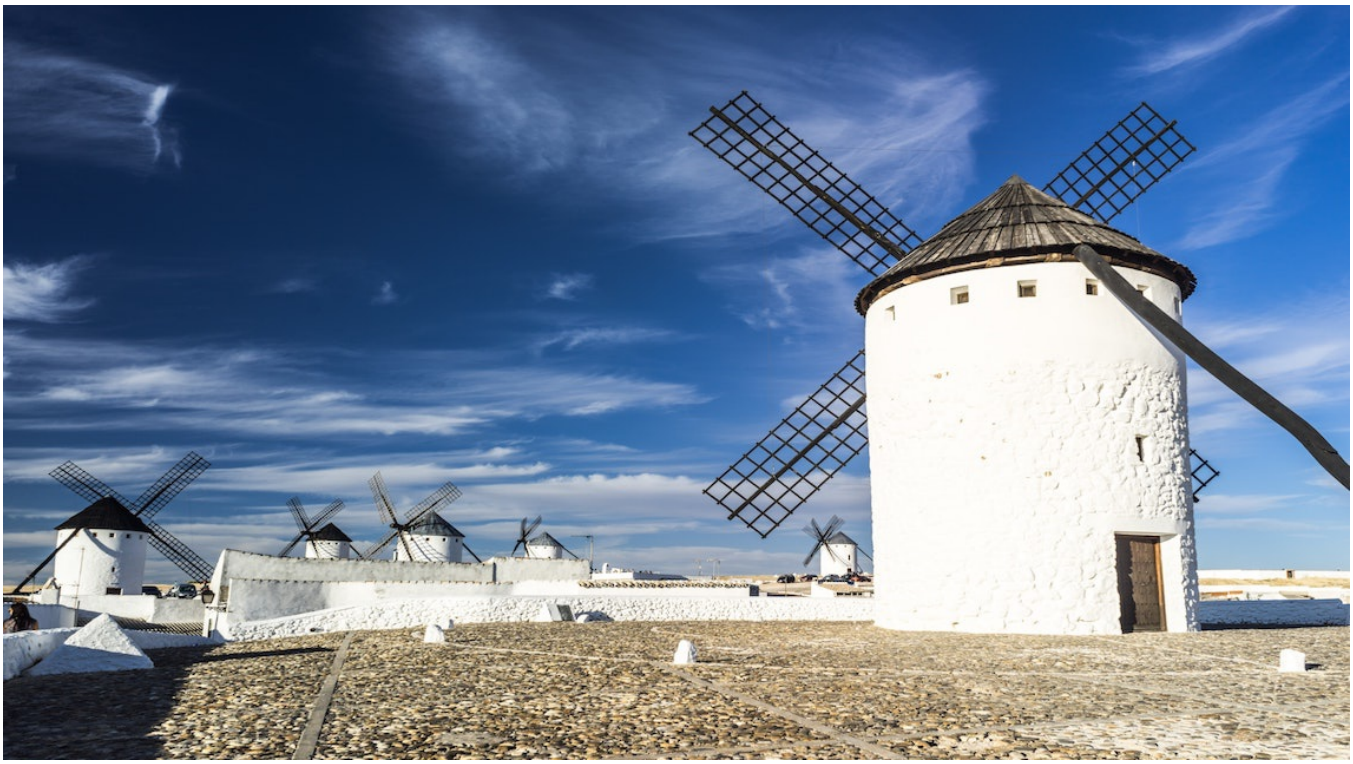
49 | 技术干货那么多，如何选？

2018-11-23 胡峰



在我刚进入行业的早些年，也是互联网的早期，其实网上的信息都不算特别多，而技术干货类信息更是少，所以就养成了一个习惯，遇到好的技术干货类文章就会收藏下来。这个习惯延续了多年，后来某天我突然发现仅仅是微信收藏夹内保存的技术干货型文章就已经累积了半年之多，都没时间去阅读和筛选。

收藏了如此多的干货，半年没读似乎也没缺了啥，那么还有必要读吗？2011 年时，我刚进入互联网行业，那已是互联网时代的成熟期，移动互联网的孕育期，也肯定是信息爆炸的时代，但依然是技术干货寥寥的时期。如今，却已是连技术干货也进入了爆炸期，那我们该如何挑选与应对？

循证与决策路径

为什么我们会去挑选和阅读技术干货文章？我想，循证大概是一个原始诉求，通过分析别人走过的路径，来拨开自己技术道路探索上的迷雾。

循证方法，也是我早年刚接触 J2EE 开发时遇到的技术决策指导思想，记得**J2EE Development without EJB**一书的译序中有一段话，很好地阐释了“循证”方法：

> 任何一个从事 J2EE 应用开发的程序员或多或少都曾有过这样的感觉：这个世界充斥着形形色色的概念和“大词”，如同一个幽深广袤的魔法森林般令人晕头转向，不知道该追随这位导师还是该信奉那个门派。

> 这时，Rod Johnson 发出振聋发聩的一呼：尔等不必向泥胎偶像顶礼膜拜，圣灵正在尔等自身——这就是他在书中一直倡导的“循证架构”。选择一种架构和种技术的依据是什么？Rod Johnson 认为，应该是基于实践的证据、来自历史项目或亲自试验的经验.....

所以，我们去阅读技术干货文章，**想从别人的分享中获得对自己技术方案的一个印证。这就是一种行业的实践证据，毕竟想通过听取分享去印证的，通常都是走过了一条与自己类似的道路**。技术道路的旅途中充满着迷雾与不确定性，我们不过是想在别人已走过的类似道路中获得指引和启发，并得到迈出坚实下一步的信心。

这就是**循证方式的技术决策路径**。

多年前，我们刚开始做咚咚这个 IM 系统时，就是沿着这条路径一路过来的。

刚启动是 2012 年，一开始其实是完全不知道怎么迈步，专门花了三个月时间来研究业界的 IM 软件系统都是怎么做的。当时行业 IM 第一的当属 QQ，但那时腾讯公司的技术保持神秘而低调，在互联网上几乎找不到任何公开的技术分享资料。

退而求其次，我们只好研究起开源的 IM 软件，也就是基于 XMPP 开放协议实现的一类开源 IM 服务端和客户端，并以此为基础去设计我们自己的 IM 架构，然后实现了一个最初的原型。

再后来，腾讯终于有一位即时通讯的 T4 专家出来分享了一篇关于 QQ 的后台技术架构演进之路，记得是叫《1.4亿在线背后的故事——QQ IM后台架构的演化与启示》。我仔细听了一遍，又把分享材料翻过好多遍，思考并体会其中的架构演化道路。

数年后，微信在移动互联网时代崛起，并且在 IM 领域甚至还超越了 QQ，微信团队也分享了其后端架构演进之路。此时，我们自身的架构也基本成型并运行一些年了。而我也注意到，关于 IM 类消息应用最核心的一个技术决策是：**消息模型**。微信的方式和我们并不一样。

微信的方式是基于消息版本的同步加存储转发机制，而我们则是基于用户终端状态的推送加缓存机制。微信的机制从交互结构上更简洁和优雅一些，在端层面的实现复杂度要求更低，符合其重后端、轻前端的设计思路和原则。

然而，循证的方式就是：即便你看到了一个更好的技术与架构方式，但也要结合自身的实际情况去思考实践的路径。消息模型，作为一个核心的底层架构模型，也许刚起步未上线时，变更优化它只需要一两个程序员一两周的时间；但经过了数年的演进，再去完全改变时，就需要各端好几个团队配合，并忙上一两个季度了。

循证，不一定能立刻给你的当下带来改变，但可以给你的演进路径方向带来调整，未来将发生改变。

切磋与思考方式

技术干货多了以后，在类同的领域都能找到不同公司或行业的实践分享，这时不仅可以循证，还能够达到切磋和多元化思考的目的。

处在 IM 这个领域，我就会经常去看关于 IM 相关技术领域的干货文章，所以我知道了微信的消息模型采用了推拉结合，还有基于版本的同步机制。但我不会再纠结于为什么我们不同，而是去看到它的好处与代价。

一个更具体的切磋案例：大家都熟悉且特别常用的功能 —— 群消息。关于群消息模型，微信采用的是写扩散模型，也就是说发到群里的一条消息会给群里的每个人都存一份消息索引。这个模型的最大缺点就是要把消息索引重复很多份，通过牺牲空间来换取了每个人拉取群消息的效率。

好多年前我们刚开始做群时，也是采用了的写扩散模型，但后来因为存储压力太大，一度又改成了读扩散模型。在读扩散模型下，群消息只存一份，只需记录每个群成员读取群消息的偏移量，偏移量的记录相比消息索引量要小几个量级，所以减轻了存储压力。

而之所以存储压力大在于当时公司还没有一个统一的存储服务组件，我们直接采用Redis的内存存储，当时原生的 Redis 在横向和纵向上的扩展性都比较受限。这在当时属于两害相权取其轻，选择了一个对我们研发团队来说成本更低的方案。

再后来公司有了扩展性和性能都比较好的统一存储组件，实际再换回写扩散模型则更好。毕竟读扩散模型逻辑比较复杂，考虑自己不知道加了多少个群了，每次打开应用都要检查每个群是否有消息，性能开销是呈线性递增的。

同一个技术方案在不同的时期，面临不同的环境，就会带来不同的成本，并做出不同的选择与取舍。虽然看起来是在走类似的路，但不同的人，不同的时代，不同的技术背景，这些都导致了终究是在走不同的路。路虽不同，但可能会殊途同归吧。

切磋带来的思考是：**你不能看见别人的功夫套路好，破解难题手到擒来，就轻易决定改练别人的功夫**。表面的招式相同，内功可能完全不同，就像金庸小说里的鸠摩智非要用小无相功催动少林七十二绝技，最后弄得自废武功的结局。

切磋，主要是带给你不同的思维方式，用自己的功夫寻求破解之道。

连结与知识体系

干货多了，时间有限，自然就存在一个优先级的选择阅读问题。

就我个人来说，我的出发点很简单，有两点：基于功利性和兴趣。说起功利性也别觉得不好，毕竟整个商业社会都是基于功利性为基础的，所以基于此的选择其实是相当稳定的。考虑下所在组织和团队的功利性需求来做出技术的选择，有时甚至是必须的，而不能完全由着兴趣来驱动。

我在前文《领域：知识与体系》中已经有过说明，我会把过去自己所掌握的所有技术总结编织成

一张“网”，若一个技术干货分享的东西离我的“网”还太远，我就会放弃去了解。因为如果不能连结到这张“网”中，形成一个节点，我可以肯定它就很难发挥任何作用，很可能是我看过之后没多久就遗忘了。

如今技术发展百花齐放、遍地开花，但人生有限，所以你必须得有一种方式去做出选择，最差的可能就是所谓的随性选择。我觉得很多情况下是需要一个选择指导框架的，而对于如何选择阅读技术干货的问题，前面比喻的那张“网”就是一个我自己的指导框架。

即便是针对同一个问题或场景，我们也可以将已知的部分连结上新的知识和实践，形成更密、更牢固的技术体系之网。

刚做 IM 时，曾经有个疑惑，就是 IM 的长连接接入系统，到底单机接入多少长连接算合适的？很早时运维对于长连接有个报警指标是单机 1 万，但当时我用 Java NIO 开 2G 最大堆内存，在可接受的 GC 停顿下，一台 4 核物理机上测试极限支撑 10 万长连接是可用的。那么平时保守点，使用测试容量的一半 5 万应该是可以的。

之后一次机会去拜访了当时阿里旺旺的后端负责人，我们也讨论到了这个长连接的数量问题。当时淘宝有 600 万卖家同时在线，另外大概还有 600 万买家实时在线，所以同时大概有 1200 万用户在线，而当时他们后端的接入服务器有 400 台，也就是每台保持 3 万连接。他说，这不是一个技术限制，而是业务限制。因为单机故障率高，一旦机器挂了，上面的所有用户会短暂掉线并重连。若一次性掉线用户数太多，恢复时间会加长，这会对淘宝的订单交易成交产生明显的影响。

他还说了一次事故，整个机房故障，导致单机房 600 万用户同时掉线。整个故障和自动切换恢复时间持续了数十分钟，在此期间淘宝交易额也同比下降了约 40% 左右。因为这种旺旺在线和交易的高度相关性，所以才限制了单机长连接的数量，而当时已经有百万级的单机长连接实验证明是可行的。

在一篇关于微信红包的的技术干货文章《100亿次的挑战：如何实现一个“有把握”的春晚摇一摇系统》里提到：

> 在上海跟深圳两地建立了十八个接入集群，每个城市有三网的接入，总共部署了 638 台接入服务器，可以支持同时 14.6 亿的在线。

简单算一下，大概就是 228.8 万单机长连接的接入能力，14.6 亿怕是以当时全国人口作为预估上限了。实际当然没有那么多，但估计单机百万长连接左右应该是有的。这是一个相当不错的数量了，而采用 Java 技术栈要实现这个单机数量，恐怕也需要多进程，不然大内存堆的 GC 停顿就是一个不可接受和需要单独调优的工作了。

以上就是从干货中提取知识和经验总结的案例，形成对已有知识的连结。这就是不断加固并扩大自己的技术知识体系之网。

总结来说：面对众多的技术干货，从循证出发，找到参考，做出技术决策，决定后续演进路线；在演进路上，不断切磋，升级思考方式，调整路径，走出合适的道路；在路上，把遇到的独立的知识点，不断吸收连结进入自己的技术知识体系之网。

回答了标题的问题，这篇文章也该结束了。面对技术这片大海，我们都是一个渔民，三天打鱼，两天结网。愿你的“网”越结越大，捞的“鱼”也越来越多，也欢迎留言分享下你的“打鱼”和“结网”经历。