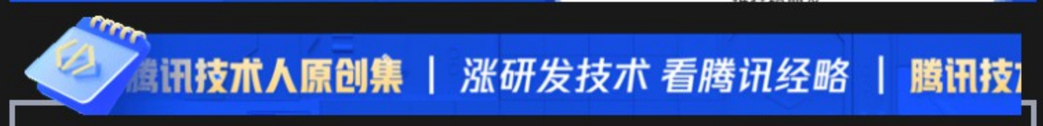
什么才是真正的架构设计？

原创 刘光利 腾讯云开发者 2024年09月18日 18:31 广东



👉目录

就算你是一个打螺丝的，你依然每天游走在这个系统的“架构”里，在里面修修补补，你得从“架构”的全局角度去审视你每天忙碌的价值和意义。经历的项目多了， 在进入新的团队，有些老项目，在了解业务背景后， 你头脑中可能已经闪现出一张“架构”了，然后你去看代码的时候大喜：“果然如此”， 这种“架构”背后的代码让你读起来神清气爽；也有些项目，你在读代码的时候发现和你脑海中闪现的“架构”不一样，这时你只能骂咧咧的合上笔记本，心想怎么会“架构”出来这种坨坨，喝杯咖啡之后，继续来啃里面的“屎山”。

01

什么是架构

前面多处提到了“架构”这个词，架构架构，到底什么是架构？，每个人都有不同的理解，实际工作中，对于同一张架构设计图，由于不同的人对于“架构”、“系统”、“模块”这些相关概念的理解不一，讨论的时候往往很难形成统一结论。

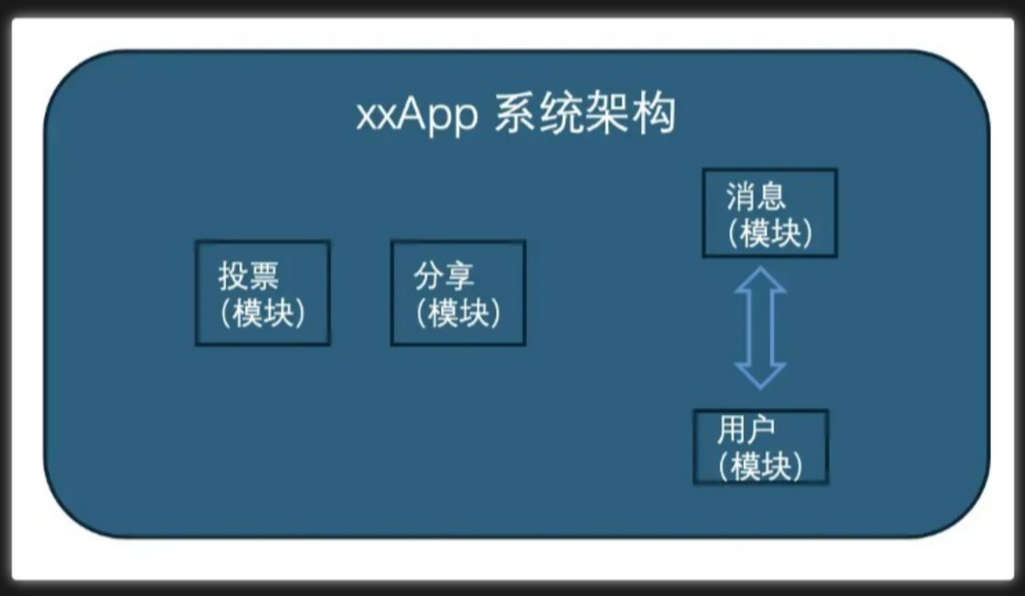
首先搞清楚什么是“架构”， 网络上有不少文章对此做解释， 其中李运华大佬的《从零开始学架构》前两个章节介绍得比较清晰。“架构”一词可以作为名词， 也可以作为动词。作为名词描述的是软件的结构组织关系；作为动词，指软件结构的设计和演变过程。先来看看做为名称， 架构的组成要素：

系统：当“架构”做名词的时候， 可以简单的把系统同等预架构， 可以说一个 App 的系统架构。

模块：系统不能描述架构的内部细节， 需要划分为各个模块， 例如 xxApp 的架构：

组件：可以认为是系统的最小组成单元，例如消息模块， 如果继续细化的话， 可以拆分成消息发送器、消息接收器等组件。

关联：组件、模块往往不是独立存在的，通常都存在着某种关联， 例如消息模块和用户模块，存在着依赖关系。



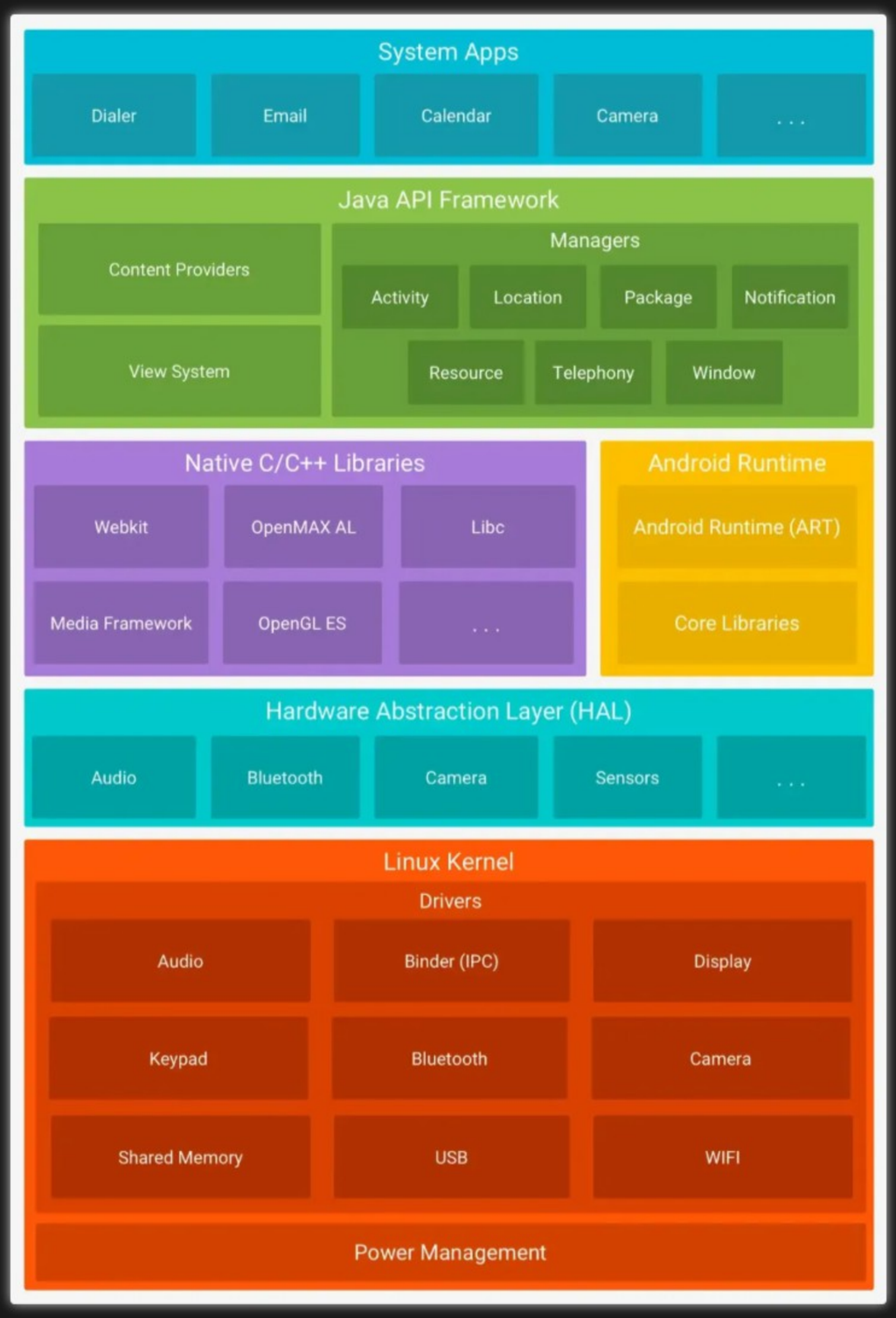
子系统：当一个架构规模和复杂度都比较大时，光用模块和组件可能已经描述不清楚了， 可以进一步把某些相关的模块群化为子系统。例如微信可能的架构图：

以上从架构的名词层面对齐了架构组成元素的概念。“架构”作为动词的时候，我们讨论的是架构的一些方法论。

02

纵向架构

纵向架构，强调的是分层，核心就是分层思想，这个在操作系统架构上已经是一个经久不衰的设计思想了。



这样分层隔离的好处不言而喻， 如果你做过 Android 模拟器开发， 你就知道这里的 HAL 层是多么重要， 通过它可以轻易替换 Linux kernel 的实现， 通过模拟，让 Android 可以轻松跑在 Windows 的系统里。

分层架构是一种将系统程序按功能和职责划分为不同层次的设计方法。每一层都有其特定的职责,通过清晰的接口与其他层进行交互。这种架构方式有助于提高代码的组织性、可维护性和可测试性， 可以轻松的替换、扩展其中一层的核心能力。关于 App 如何做分层架构设计， 我在 16 年就写过一篇文章《Android 应用架构概述 https://www.jianshu.com/p/e157cc64ea5c》，现在回头看， 分层的本质思想依然没有变。

03

三层架构

最基本的分层架构通常包含以下三层：

- **表现层(Presentation Layer)**：负责用户界面的展示和用户交互，包括 UI 组件、视图控制器等，主要关注用户体验和界面逻辑 。
- **业务逻辑层(Business Logic Layer)**：实现核心的业务逻辑和规则，处理数据的加工、转换和验证、协调表现层和数据访问层之间的交互。
- **数据访问层(Data Access Layer)**：负责与数据源进行交互，包括本地数据存储(如 SQLite、Core Data)和网络请求，提供数据的 CRUD(创建、读取、更新、删除)操作。

04

四层架构

一定规模的 App 通常还会继续分层，例如可以扩展出基础层出来。

表达的是分层的思想和方法， 具体分多少层， 取决于 App 规模和复杂度。

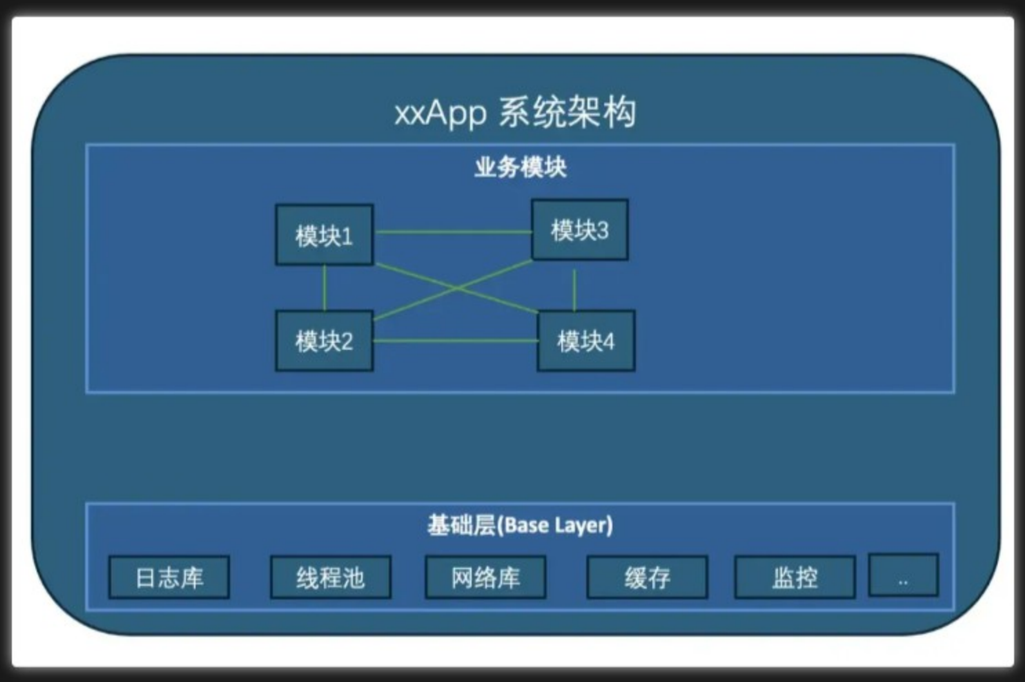
检验标准：是否成功做到了分层， 检验的标准就是是否真正给 App 带来了前面提到的分层的好处，例如：要替换其中一层里的核心能力，是否可以低成本就替换， 当某一层修改的时候，是否可以单独的对该层进行单元测试。

05

横向架构

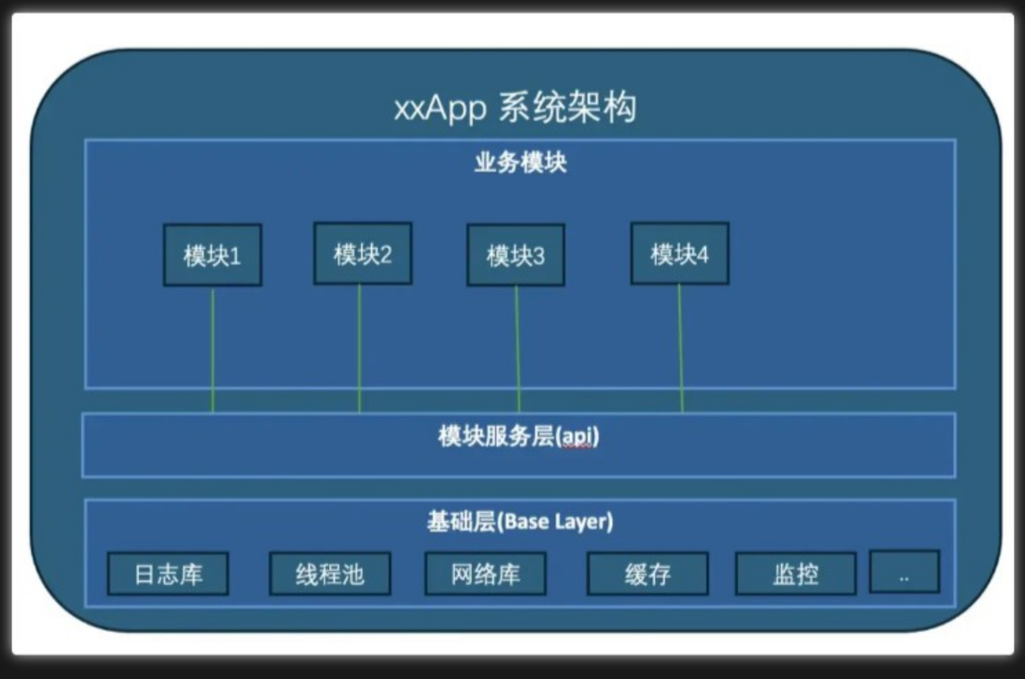
随着 App 功能规模、团队规模不断扩大， 这时候可能需要层业务模块角度关注架构设计，目的是降低功能规模增加而带来的复杂度爆炸增长，横向架构强调的是业务模块之间的横向解耦，从而降低复杂度。

5.1 复杂度的评估



图中4个模块都存在着依赖关系， 那么复杂度计算为 24：复杂度=4x3x2x1

如果我们通过某种手段进行解耦（例如通过增加一个 api 服务层隔离）：



通过这样解耦，复杂度变为了 5：四个模块加一个隔离层。

通过上面的分析，基本可以看到， 模块化解耦是一个 App 发展到一定规模后绕不开的话题，淘宝、微信等这样的巨型 App 早已在迭代过程中进行了多轮的模块化重构， 参考：《微信 Android 模块化架构重构实践

https://cloud.tencent.com/developer/article/1005631》

检验标准：横向架构是否成功， 检验的标准是，如果这个模块给独立小组开发，是否可以独立开发、编译、调试。

5.2 架构不是一套打遍天下

我经历过几十万、几百万的中小型 App 搭建， 也参与千万日活的大型 App 应用架构，后者的架构更复杂，用到的技术和手段也更丰富，是不是就可以把后者套用上去呢？答案显然是否定的，不可能说微信的架构很牛， 那我们就搬过来用。架构设计需要满足三原则：

详细可以参考：

《架构设计三原则https://time.geekbang.org/column/article/7071》

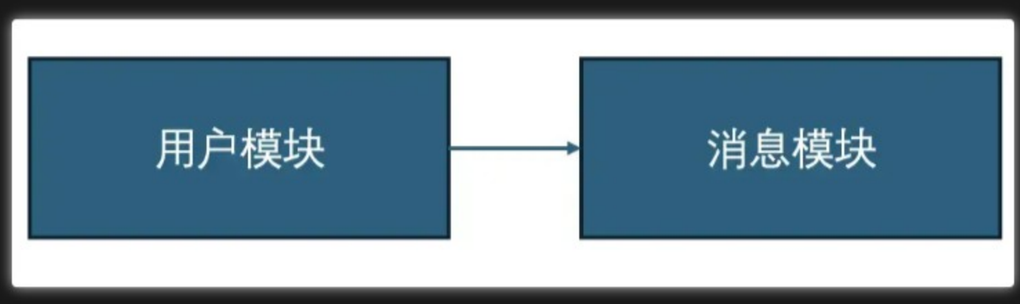
- **适合原则：**根据当前的业务类型、规模选择合适的纵向、横向架构设计方案
- **简单原则：**所有在做架构时，在有复杂方案和简单方案都可以满足当前业务是，尽量选择简单方案 。
- **演化原则：**微信、淘宝这样的巨型 App，也不是一开始就设计成这样的架构的，很多 App 开始的时候可能都没有出现模块化、插件化这样的横向架构设计，都是随着功能和用户规模不断扩展，架构不断重构演化而来的。首先，设计出一个满足现有业务的架构，架构要在实际应用中不断的优化，保留其优秀的部分，修改有缺陷的设计、改正错误的设计、去掉无用的设计，使得架构不断完善。当业务发展时，旧的架构可以要重构、甚至重写，但是有价值的经验、教训却会在新的架构中体现。

5.3 先有架构再有设计模式

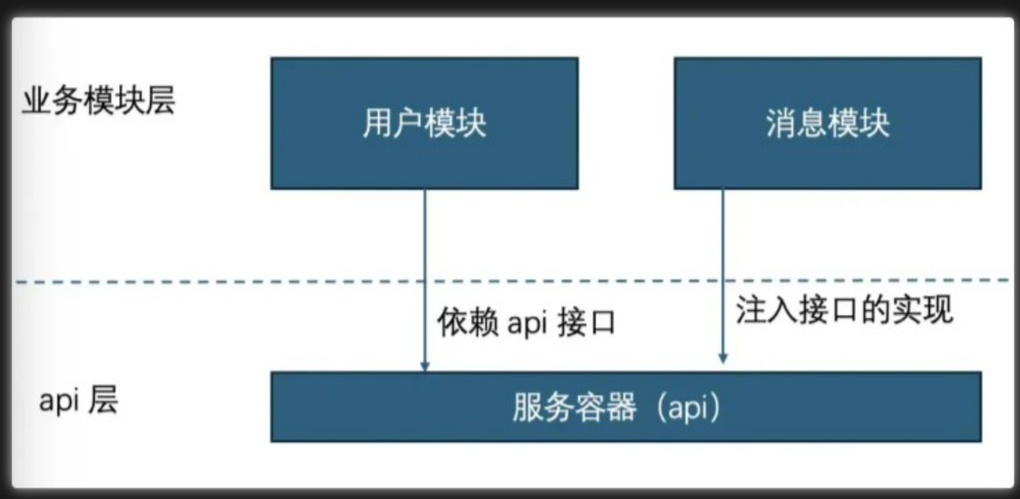
初入这个行业的时候， 很多设计模式是看得云里雾里， 我觉得是我们搞反了顺序，导致我们很难理解为什么用这个模式、 SOLID 设计原则有什么作用。

应该是先有业务场景， 然后再有什么样的架构， 为了到达这个架构的要求， 然后才是各种设计模式和设计原则的灵活选取和应用。

例如，我们为了让用户模块和消息模块解耦，假设用户模块依赖消息模块的未读消息数这个信息。



为了满足横向架构的要求，解除模块依赖，架构变成：



通过这样调整，我们不知不觉中使用了 SOLID 中的好几个原则，例如依赖倒置原则。

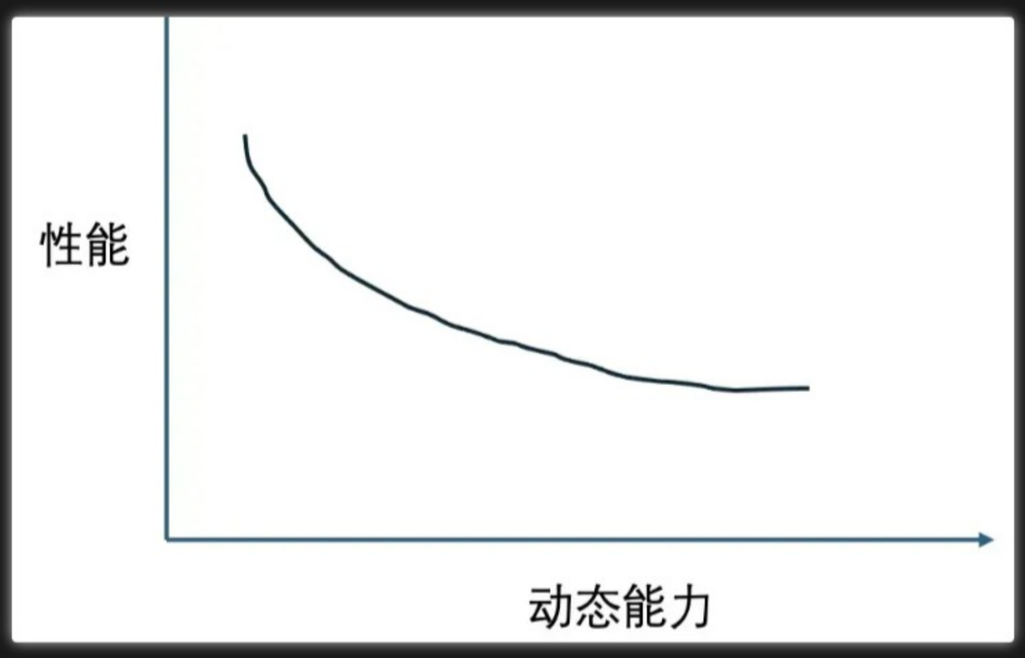
5.4 跨平台和动态性

跨平台和动态性是做 App 架构绕不开的一个话题。

两类跨平台：第一：UI 跨平台，代表方案 WebView、Weex、RN、Flutter 等类 web 方案。

第二：C++ 跨平台， 通常是在追求性能的某些特定场景， 例如音视频编解码处理，某些复杂的加解密算法等。也有一些团队由于领导班子技术栈的优势， 会选择将业务甚至 UI 也是用 C++ 跨平台，例如腾讯会议。

动态性：代表方案 WebView，Weex ，小程序等类 web 方案和原生插件化方案。



关于怎么选取，我们要完美的动态性或者跨平台性的时候，可能就是牺牲了某些性能，例如 WebView ，在跨平台性和动态性都堪称完美，但却是可选方案中性能最差的，再比如：如果我们使用 C++ ，性能上肯定没问题，但是在开发效率、团队人员技术要求上都要面临巨大挑战。

还有些选择是由业务本身决定的，例如类似应用宝这样的应用商店应用，根本就不用考虑跨平台问题， 因为它只能跑在 Android 平台， 但是它对动态性要求非常高，作为厂商的竞争对象无法在厂商应用商店上架，所以类似插件化这样的动态方案就显得非常重要。

架构中对于动态性和跨平台方案的选取，这同样是一个取舍问题， 架构干的活，大多数时候都是在做取舍和平衡。

06

总结

1. 架构作为名称描述 App 系统组织元素和组织关系。
2. 架构作为动词，强调的是架构的整体方法论， 纵向分层架构，横向模块化隔离架构，在此之下灵活使用设计模式和设计原则实现架构目标。
3. 架构要适应业务自身需求和变化， 做到三原则。

07

延展阅读

结合实践的理论才能被充分理解，腾讯云开发者社区此前联合手机 QQ 团队策划了手机 QQ 技术变迁系列好文，点击以下文章标题即可阅读，从优秀的架构实践中看懂方案设计的本质。

《不畏移山，手机QQ技术架构升级变迁史》

《QQ 客户端性能稳定性防劣化系统 Hodor 技术方案》

《QQ 9"傻快傻快"的？！带你看看背后的技术秘密》

《QQ 25年技术巡礼丨技术探索下的清新设计，打造轻盈简约的QQ9》

-End-

原创作者丨刘光利

感谢你读到这里，不如关注一下？👇

你在做架构优化的时候有哪些独门经验？欢迎评论分享。我们将选取**点赞本文并且留言评论**的一位读者，送出腾讯云开发者定制发财按键1个（见下图）。9月25日中午12点开奖。

📣📣欢迎加入腾讯云开发者社群，享前沿资讯、大咖干货，找兴趣搭子，交同城好友，更有鹅厂招聘机会、限量周边好礼等你来~

（长按图片立即扫码）

「 精品 · 知识推荐 」

赛博玄学，一键三连少一个Bug!

为好文章 点 收藏 点亮

腾讯技术人原创集 234 # 架构师 5

腾讯技术人原创集 · 目录 ≡

〈上一篇
5 款程序员画图神器，全免费！

下一篇〉
为什么软件行业仍在重蹈 50 年前的覆辙？读《人月神话》有感