

架构师必备底层逻辑：分层架构设计

原创 郑海波 腾讯云开发者 2024年09月11日 08:45 北京

关注并星标★腾讯云开发者

全网最快查收鹅厂技术干货 | 学习资源

点右上角的 ...，设为星标

腾讯云开发者

腾讯云计算（北京）有限责任公司

设为星标

推荐给朋友

腾讯云技术人原创集 | 涨研发技术 看腾讯经路 | 腾讯技7

📖 目录

1 分层的优点

2 分层的缺点

3 分层的原则

4 总结



有句话叫做互联网技术中的银弹，加一层解决各种问题。我们在微服务架构设计的时候会碰到分层，数据仓库设计的时候也有分层，协议设计的的时候也有分层，大部分的设计模式也是多加一层抽象。这些所有的分层都有什么共同点，分层的优缺点是什么，分层的原则是什么。我们经常拿到各种眼花缭乱的分层概念，在实践中又感觉无法完全套上去。本文尝试进行一次简单探讨。

关注腾讯云开发者，一手技术干货提前解锁🔓

腾讯云开发者

腾讯云官方社区公众号，汇聚技术开发者群体，分享技术干货，打造技术影响力交... >

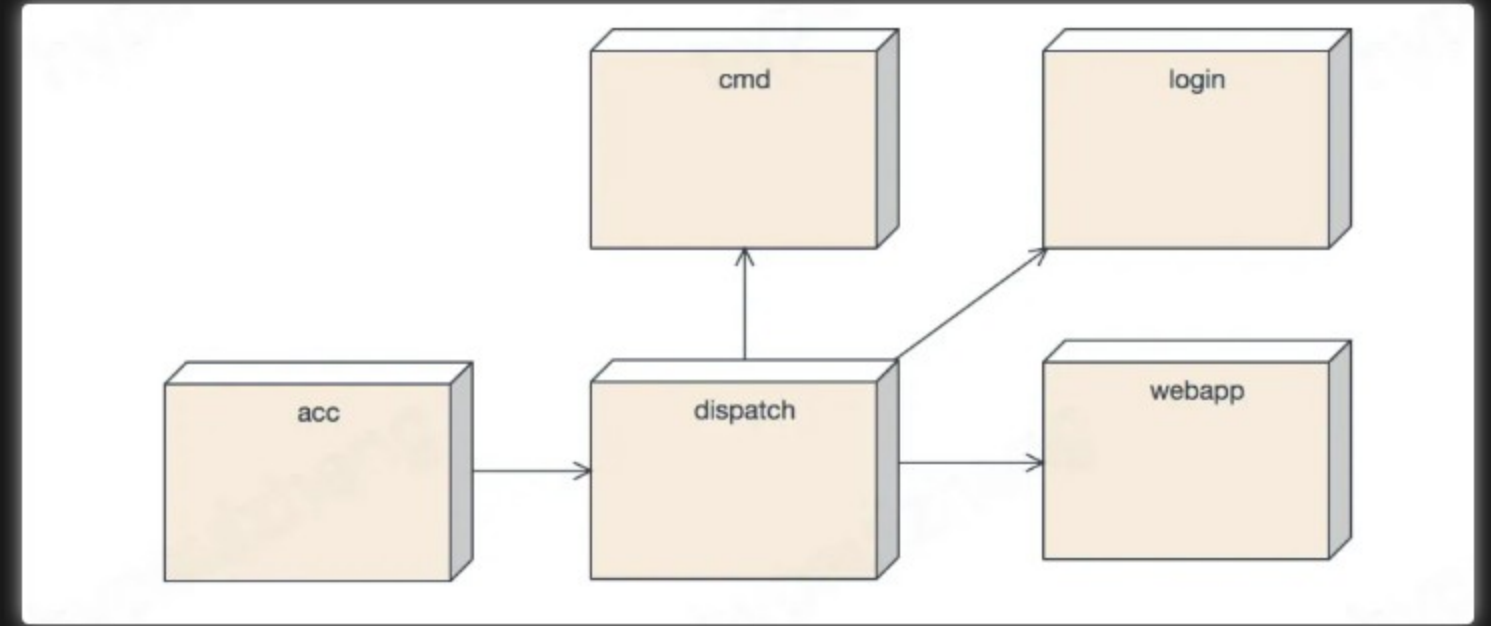
925篇原创内容

公众号

01 分层的优点

分层的优点归纳为五种：抽象稳定，功能复用，功能内聚，屏蔽复杂和变化，扩展规模。仔细一想似乎前四个都可以用抽象一词概括，但表现的侧重点不同，还是分成并列的五项。

最早接触到架构中的分层思想，来自我入职一个月后的转正答辩。下图是古早时期 QQ 空间的 wns 接入框架的架构设计，当时被评委挑战了框架设计思路。



从名字就可以看出 acc(接入，长链接管理)，dispatch(路由)，webapp(逻辑 svr，对应着 ddd 中的 bizao 的概念)。被挑战的问题是为什么需要 dispatch 这一层？不能直接 acc 转发到 webapp 吗？当时的回答是因为需要通过路由进行容灾调度，如果后台上海的 webapp 挂了，可以在 dispatch 下发配置，调度流量到深圳。（实际跨 IDC 容灾调度并不会这么用，理论上每一层都可以容灾，acc 也可以调度流量到深圳。实际上真正容灾演练的时候，都直接客户端重定向，毕竟单 IDC 挂了，大概率可能所有层都挂了）。

这里就体现了分层的优点之一，屏蔽复杂和变化。dispath 分层其他优点总结如下。

复用：dispatch 除了路由外，还承担了解压缩，鉴权，加解密，chunk 分包装等一些通用能力的封装；

内聚：这些复用的逻辑为什么封装在了 dispatch 单独一层，不放在 acc 一起，因为功能内聚（也可以叫关注点分离，变化分离，轻重分离，快慢分离都能搭上边）。acc 只做长链接的管理和请求的 upstream，downstream 的转发，采用的 multi reactor 的模型，是稳定的模块，不会经常重启导致长链接中断，dispatch 可以采用传统的 spp 框架，经常下发业务路由配置重启。

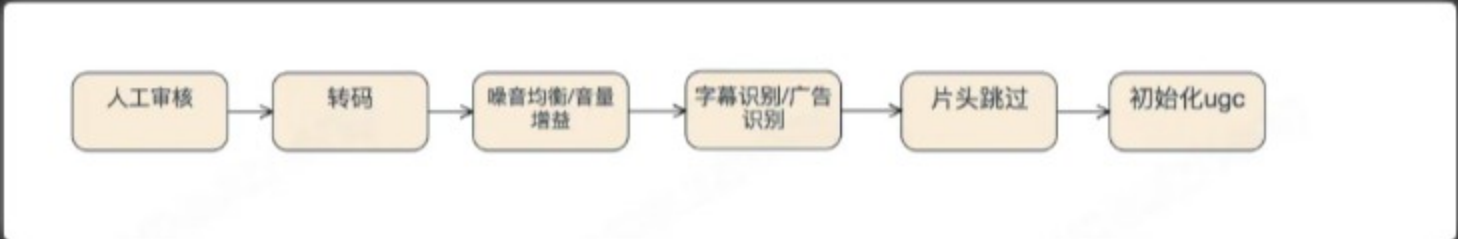
屏蔽复杂和变化和这个应该是日常架构设计最多的一种场景，在架构设计中，无论是防腐层，还是适配层都是类似优点。其他常见的场景包括：

arp 协议/dns 协议	路由 proxy，依赖转换	存储 access
屏蔽 ip 地址/mac 地址和变化，简化记忆	常见的异构双路存储备份，通过kv路由映射proxy屏蔽变化容灾。这里似乎新增了一个依赖，降低了可用性？我们会认为kv的可用性高于db，用 kv 增加一个到db的路由映射。假设 kv 可用率 5个9，db3个9。添加 kv 路由后 $(1 - (1-99.9\%) * (1-99.9\%)) * 99.999\% = 99.998$	屏蔽底层存储差异，对外只暴露中标准的存储模型协议

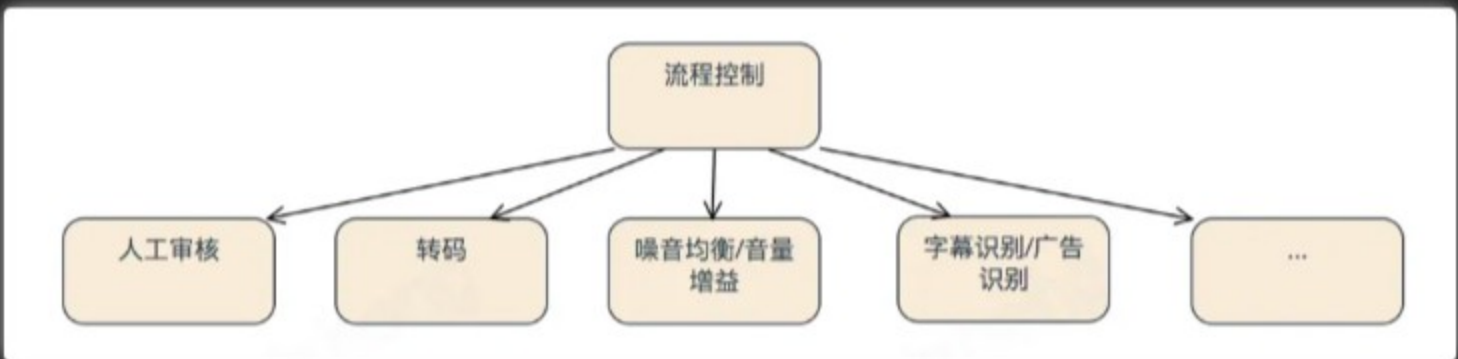
指数级扩展规模是另外一种场景。经典的内存页表，一级页表，二级页表，通过多层次划分，m+n 的存储，达到 m*n 的寻址，类似思想的还有 roaingbitmap。系统容量上的扩展，典型的如 im 消息的扩散，多层的读扩散，写扩散，m*n 的系统容量扩大。

这里简单介绍下曾经碰到的一种不同的分层读写扩散。比如有1kw的用户，1kw 的书籍，现在要对部分用户指定部分限免，平均每个人有10w 本限免书籍，每本书被10w 人授予限免。如果存储单向的关系，内存存储用户 id-【书籍 id 列表】或者书籍 id-【用户 id 列表】，都是1kw*10w 的写扩散和存储占用，后来存储设计中多设计一层号码包，变成内存用户 id-【号码包列表】+ 内存书籍 id-【号码包列表】+ 离线存储号码包 id-【书籍 id+用户 id】。读取逻辑变成同时读取2个映射关系后取交集判断是否限免。减少了在线写扩散和存储占用。

分层带来的抽象稳定则是我们代码设计中最常碰到的，也是架构重构优化中会碰到的。下面举个架构重构中的例子，这是老的音频上架流程如下：



流程冗长，不同的模块在多个不同的开发手里。上下游通过异步消息投递通信。流程中断失败，依赖各个模块内部自己重试，流程不可控，且难以监控，重构后版本如下：



通过流程控制抽象模块，串联全部的流程，把重试和上下游逻辑剥离出来，让各个模块专注于自己的业务逻辑处理。比如转码是个 CPU 消耗类模块，只专注于转码服务本身，可以部署高 CPU 配置的物理机，也方便扩容。通过逻辑抽象，抽离出公共模块，来简化系统复杂度和提升可扩展性。

02

分层的缺点

分层的缺点归纳为三个：系统复杂度增加，性能/存储消耗，依赖添加/依赖传递。

系统复杂度增加：是否会增加系统复杂度是相对的，拆分是应对复杂系统的利器，但是过早的拆分设计会导致系统复杂，如果你的上架流程就2步，添加一个流程控制层完全没有必要。

性能/存储消耗：在请求量高的场景，多一层只做转发逻辑也会耗掉大量机器，在一些高并发的场景，就不需要分层设计了，就像2015微信的企业红包，接入层做逻辑返回。

依赖添加/依赖传递：新增一个第三方组件，无论可用性是否是5个9，都是一个外部依赖，都存在风险，且随着分层的增加，系统的可用性降低， $99.99*99.99=99.98$ 可用性从四个9变3个9。这里也给应对的分层的了一些后发，就是逻辑的分层依然存在，代码目录分层可以依然存在，只不过同机部署优先本地路由，或者编成一个单体服务。

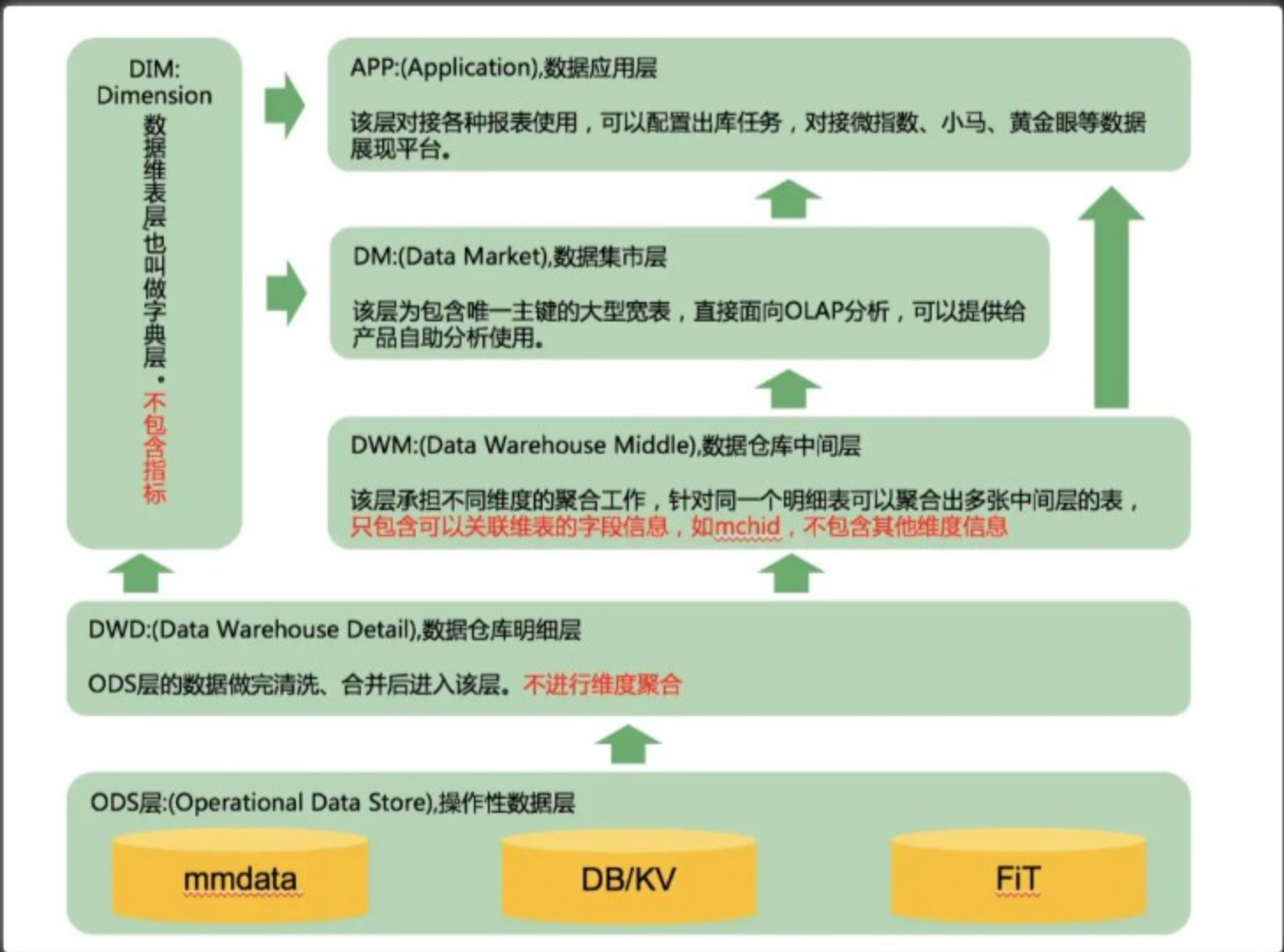
03

分层的原则

在概述完分层的优点和缺点后，我们来谈谈分层的原则。权衡优点和缺点综合考虑，这是一句废话~

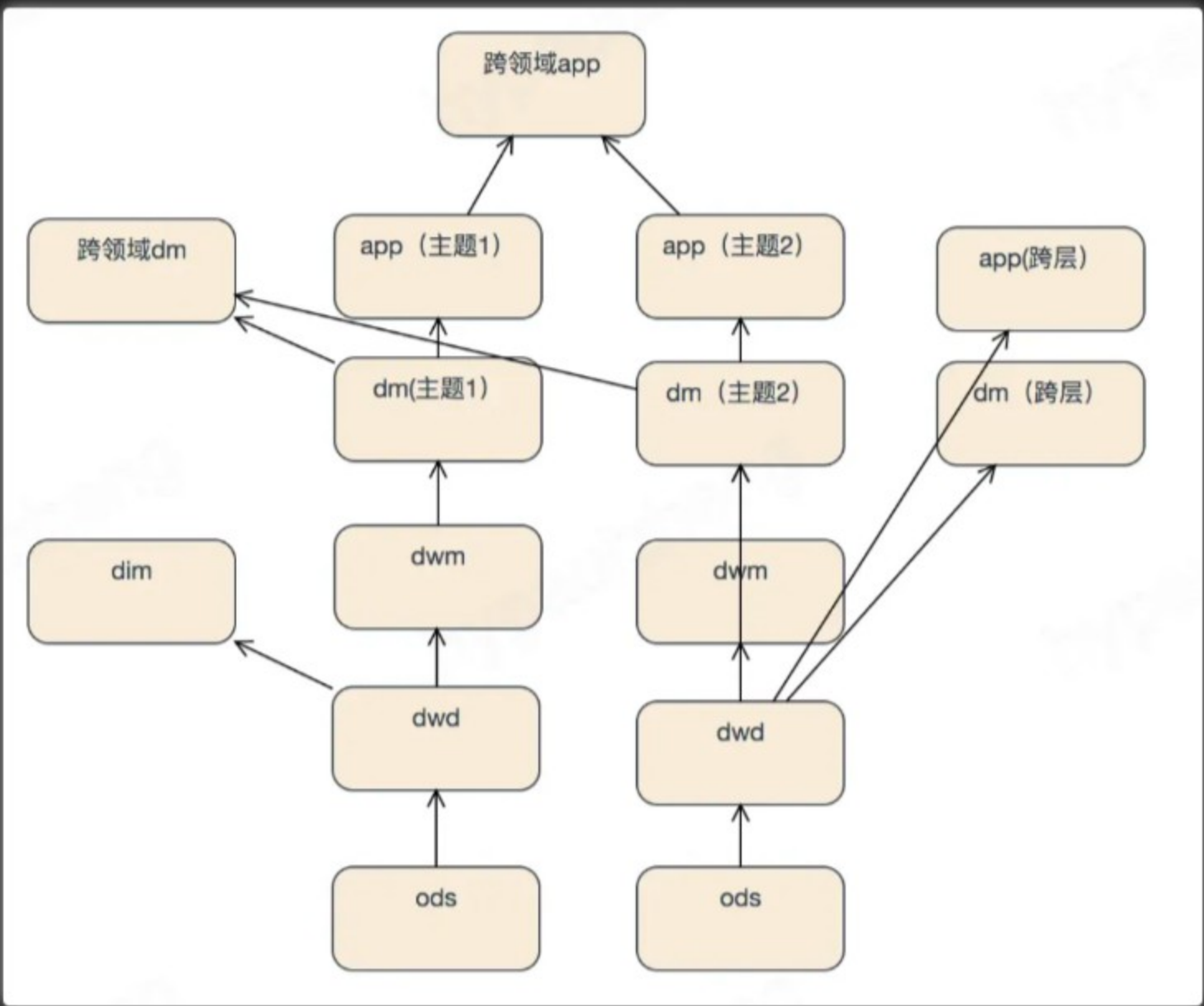
我们按照具体的例子来讨论，分层中我们经常碰到的疑惑包括，要不要加一层，按照xxx分层原则，此处应该加一层；按照可扩展性的设计，此处应该加一层。加完分层后，不同分层的调用需要遵循什么样的约束。

此处我们举个数据仓库设计的例子，标准的数仓设计分层原则如下：在数仓中，包含 dwd（从流水表中清洗的数据，主要是非法数据过滤，格式转换等），dwm（轻度聚合多维度group by），dm（面向主题层，包含指标和维度），app（高阶的统计数据，可出库到报表）。

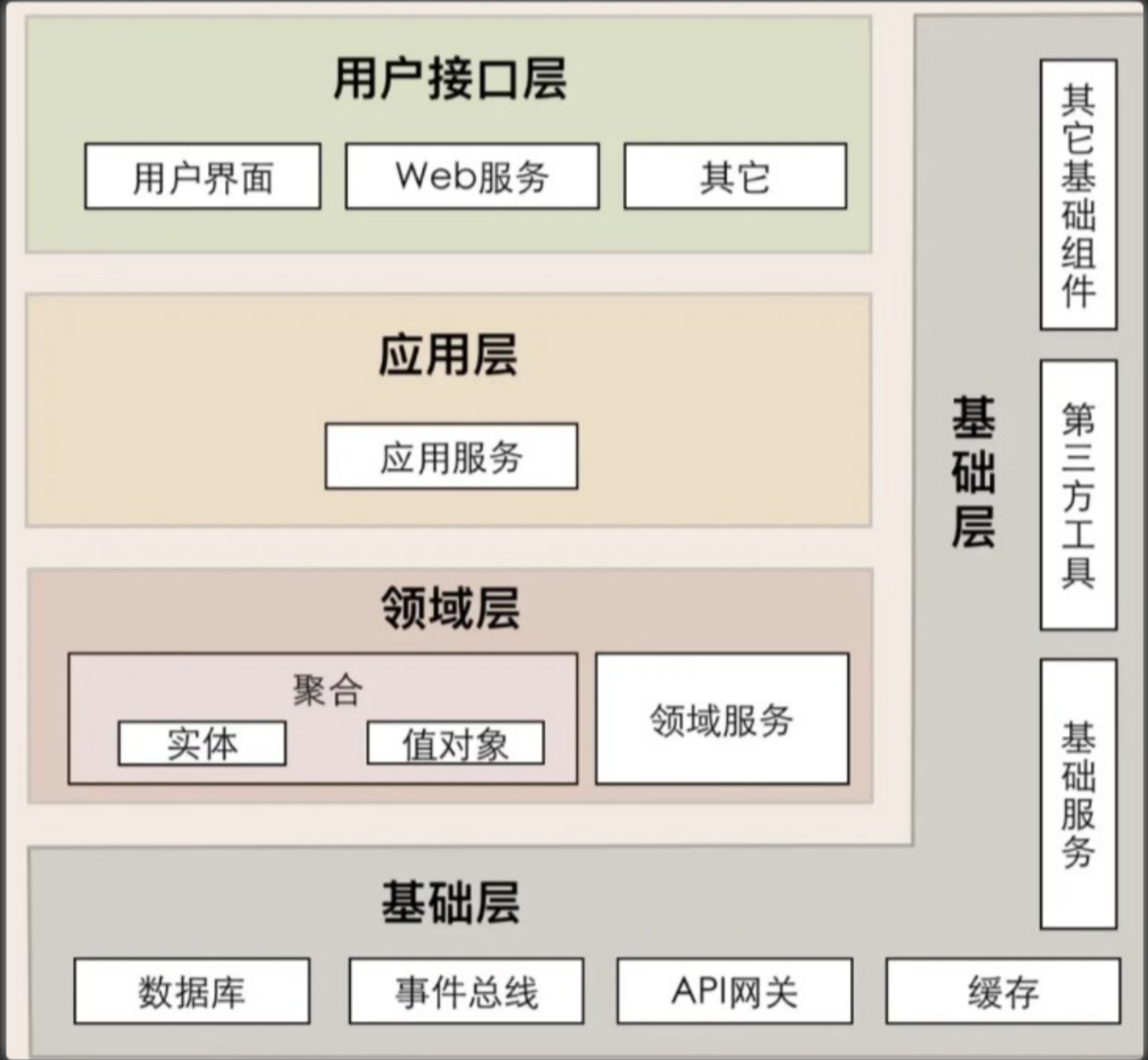


这里就会有几个问题，为什么需要 dim 这一层，没有 dim 不行吗？全部都用dm也能满足需求，因为 dm 中的数据是全的；是否一定需要 dwd，如果这个流水数据只是用来统计一个指标，app 层可以穿透到 ods 层似乎也合理？实际的app层报表需求时眼花缭乱的，往往需要横跨多个不同的主题算指标数据，那同层之间的跨主题调用怎么处理。

总结成分层的通用问题，就是是否需要这一层，看上去没有被复用，做了感觉有过度设计的感
觉；是否可以跨多层调用；是否允许同层调用。说下我们最终的方案，dwd 和 dim 都必须保
留，除了分层本身的好处外，还有一个是为了一致性的规范。允许跨多层调用到 dwd：app 到
dwd, dm 到 dwd。不要为未来还看不到的需求做过多的层次设计，否则重跑历史任务的时候
是灾难。允许同层调用，但是优先调用下层。因为越底层越稳定，同层依赖容易形成依赖循
环，或者是自依赖。最终实践的版本如下：



总体原则就是，有的层是必须保留的，仅仅是为了全局的一致性。不要为了扩展而过多设计中间层，如果中间层只有一个下游，那要三思是否需要保留中间层，所以允许跨层调用。每一层允许有多层，没有规定一层只能有一个表。按照主题域划分，主题域内保持独立，跨主题的单
独做一层，但是所有的跨主题表，都从主题表出，需要在追求中间表复用的同时保证结构的清
晰。



同样 DDD 微服务的标准分层如上图，微服务有一个核心概念叫做独立自主的领域，领域层要尽量少的依赖领域外的东西，才能够足够的稳定，满足越底层越稳定的依赖的要求，但这样的设计容易走到了贫血模型的模式，有 DDD 的概念，却没有 DDD 的实际作用。在充血模型实践中，领域层可以网关调用系统外的接口，可以通过异步消息投递，调用系统内他领域的应用层。应用层没有业务逻辑，应用层负责串联一个系统用例，只是薄薄的一层。大部分逻辑下沉到领域层，至于领域层内分多少层多少模块，这个就按照业务实际情况是否可以复用，是否需要内聚判断，不用拘泥于是不是一层一个模块。

04

总结

分层优点：抽象稳定，功能复用，功能内聚，屏蔽复杂和变化，扩展规模。

分层缺点：系统复杂度增加，性能/存储消耗，依赖添加/依赖传递。

分层原则：为了系统一致性，可以强制保留某些分层；允许跨层调用，不允许反向调用；优先依赖下层，不依赖同层；保证越底层越稳定；允许一层有多层；还有一句废话，权衡有优缺点，利大于弊就行。

-End-
原创作者 | 郑海波

感谢你读到这里，不如关注一下？👉



腾讯云开发者

腾讯云官方社区公众号，汇聚技术开发者群体，分享技术干货，打造技术影响力交流社区。

925篇原创内容

公众号



你还知道哪些架构师必备的底层逻辑？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18日中午12点开奖。

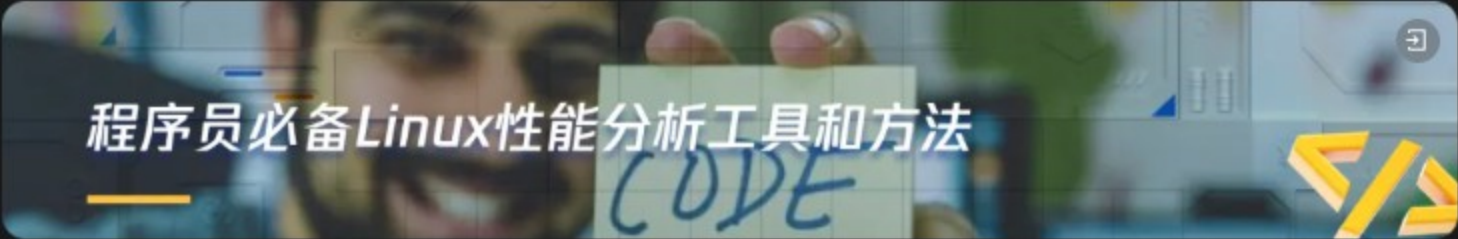


📣欢迎加入腾讯云开发者社群，享前沿资讯、大咖干货，找兴趣搭子，交同城好友，更有鹅厂招聘机会、限量周边好礼等你来~



(长按图片立即扫码)

精品 · 知识推荐



程序员必备Linux性能分析工具和方法



数据库内核工程师必读论文清单



腾讯文档前端工程架构改造实践

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

腾讯技术人原创集 · 目录

< 上一篇

为什么这段代码会阻塞？

下一篇 >

5 款程序员画图神器，全免费！