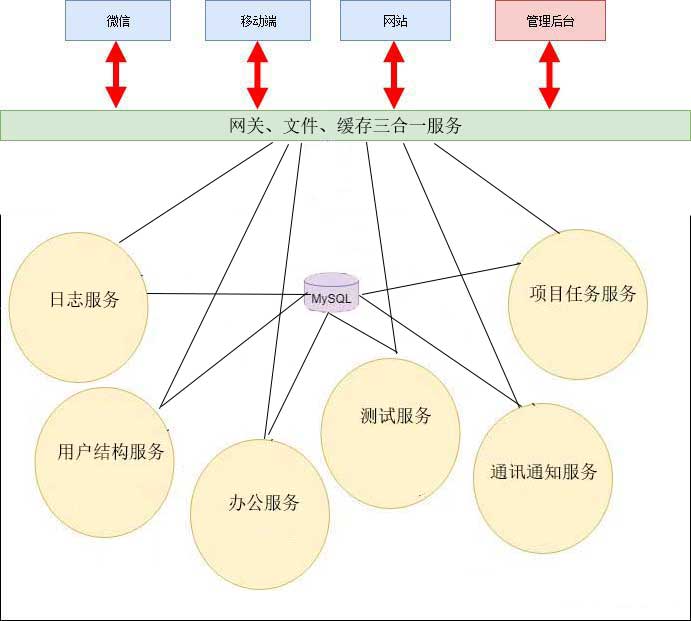
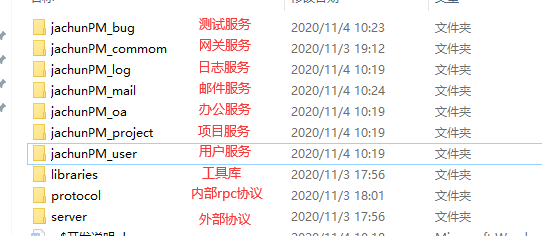
1. 微服务区分
   1. 服务序号1，网关与文件服务层，这层主要是对外的web或者api服务，将收到的请求转发给对应的服务，并且将结果返回给客户端。主要为http（s）1与2版本协议和ws（s）协议，并集成file文件服务，把上传的文件保存。为基础服务，重点开发，后期尽量少改动，需部署在有外网权限的高性能，大内存大硬盘的机器上。不带边车的Service Mesh模式
   2. 服务序号2，日志服务，保存来自于其他服务发过来的日志。
   3. 服务序号3，用户结构服务，处理与保存公司架构，人员权限信息。
   4. 服务序号4，项目任务服务，处理与保存与生产（外包）部门相关的：产品、项目、任务、版本模块。
   5. 服务序号5，测试服务，针对项目任务服务进行扩展，主要与研发相关的：测试、bug、发行等功能。
   6. 服务序号6，办公服务，处理与保存考勤相关的模块。
   7. 服务序号7，通讯通知服务，需要对外联网，实现如转发rtx，发email，发短信，im通讯等功能。
2. 单机或者局域网部署示意图
   1. 只部署1个mysql服务，但是从b到g都各自建立一个库，如后期需要扩展，每个服务对于的库均运行于独立的mysql服务之中
   2. 缓存保存在网关中，由各自服务，维护自己的缓存



1. 开发目录



1. 开发规则
   1. 服务为单层api请求设计，以msg为一个基础处理单位，具体看protocol\msg.go的接口。
   2. 服务与服务的通讯经过网关转发。
   3. 除了配置文件和只读缓存，不在服务上保存其他缓存数据，减少上下文依赖，以确保服务崩溃重启后，能直接去处理数据，以回调来开发
   4. 在protocol层或者host转发层，已充分进行多协程调用和defer回收panic，因此在写服务Handler时候，不允许直接使用gorutine 来处理msg，也无需defer回收panic。
2. 快速开发debug建议，查看protocol协议，查看网址与对应api的cmd号，全局搜索cmd号找到对应的服务模块与func进行修改。
3. 新功能开发流程
   1. 根据需求是否要新建一个服务
   2. 在protocol\协议生成器\ 文件夹下，对应的微服务协议，新增请求与响应协议，并运行protocol下协议生成器的main.go,生成新的协议
   3. 转到对应服务文件夹下handle文件夹，新增cmd与func关系
   4. 编写func详细代码
   5. 回到网关，如果是http页面，则与第6条相似，到http\handle.go新增网址与api关系，并在http下处理view页面。
4. 分布式事务注意事项
   1. 由msg进行开始事务，此msg后续请求的api，都运行在事务中
   2. 请求中的commit会被拦截下来，所以请勿直接在commit后直接执行某些成功后执行的代码，请使用CommitCallback()来设定，真的提交commit后，会调用回调。
   3. RollbackCallback()不但在Rollback后执行，在EndTransaction中也会执行，但请放心，只要执行了任意callback，都会设置为nil避免二次执行。
   4. 最终由第一个申请事务的服务器，来执行commit，这时候有个二次确认，二次确认后通知所有服务器进行commit，在二次确认和最终commit会有几ms的延迟，如果这几ms内服务器崩溃，则会出现commit但是没有提交到数据库中，是一个安全隐患，所以尽量少使用分布式事务。本地事务，请使用Hostconn.DB发起。