接口设计原则

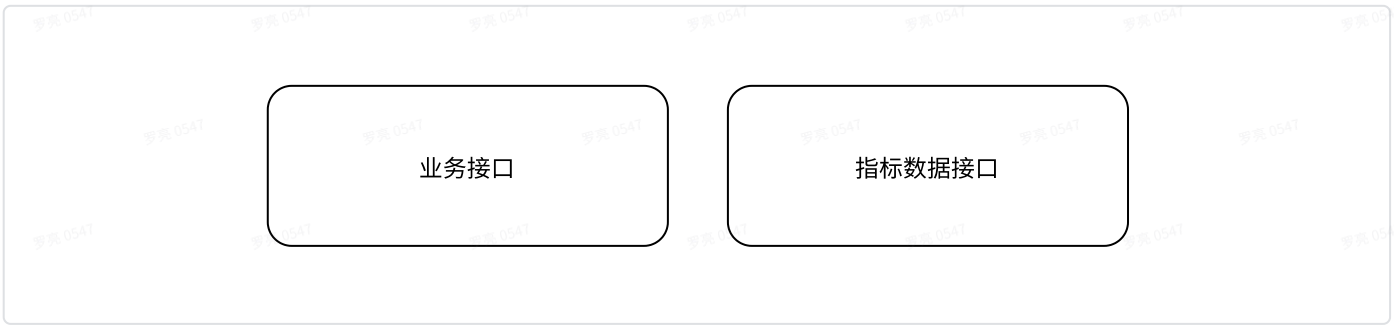
1、起因

dataSpace的数据查询通用接口是否要向前端透出?

换而言之就是：将dataSpace的数据查询通用接口定义为**外部接口**还是**内部接口**?

按照接口设计，提供了登录态字段用于鉴权，应该是外部接口

- 透出的好处
  - dataSpace变成了一个更优的低代码平台了，开发更简易
  - 接口调用次数更少，页面接口分层更清晰，将dataSpace完全抽离



- 透出的坏处
  - 需求迭代可能需要在指标数据上二次开发，导致迭代代价变大

2、接口设计原则

- 星图新接口定义必要原则
  - 对外接口需要登陆态,名称以用户类别分类，不可修改（s_demander_id、s_author_id、s_mcn_id、s_star_id)
  - req必须包含base.Base,resp必须包含base.BaseResp，用于信息透传
  - req中明确标记optional 和 required，下游调用时可以明确哪些是必传的
  - resp中的字段，除非要支持nil可以标注optional,否则default,避免下游判断nil；结构体都标准optional，避免下游使用零值
  - enum不要使用0值
  - 用注释明确接口的方法类型（http_post 、http_get)
- 星图修改接口必要原则（开闭原则）
  - 不允许修改已有字段

    - 新增字段必须标注为optional，避免已有调用出错
- 接口定义设计
    - 明确内部接口还是外部接口
    - 合理的接口命名
    - 接口功能定义要具备单一性，充分考虑接口的可扩展性，避免做大而全的接口（单一指责原则）
    - 版本控制，需求迭代导致接口无法拓展，要求维护多个版本。为了不影响老版本的使用，旧版本接口也需要维护。版本定义可以通过接口取名（V2）或添加version参数
- 接口实现设计
    - 接口实现做好分层（handler,service,business,dal,rpc,common,const)，尽量实现低耦合
    - 清晰的日志打印，方便问题定位