1、分层架构

1.1、展现层、业务层、数据层

- 层的职责
    - 展现层，或称为表现层，用于显示数据和接收用户输入的数据，为用户提供一种交互式操作的界面。
    - 业务层，或称为业务逻辑层，用来处理各种功能请求，实现系统的业务功能，是一个系统最为核心的部分。
    - 数据层，或称为数据访问层，主要与数据存储打交道。
- 层间关系
    - 展现层会向业务层传递参数，发出服务请求，并获取业务层返回的信息显示在界面上。
    - 业务层接收展现层的命令，解析传递过来的参数，判断各种合法性，并具体实现功能的各种“运算”要求，返回展现层所要的信息。
    - 数据访问层不能被展现层直接调用，而必须由业务层来调用。

1.2、UI层、SI层、PD层、DM层

- UI层，即用户界面层，负责封装与用户的双向交互、屏蔽具体交互方式。
- SI层，即系统交互层，负责封装硬件的具体交互方式，以及封装外部系统的交互。
- PD层，即问题领域层，负责问题领域或业务领域的抽象、领域功能的实现。
- DM层，即数据管理层，负责封装各种持久化数据的具体管理方式。

2、分层技巧

设计思想是“封装外部交互”。

2.1、从需求-上下文图

借助上下文图，把系统外部的4种事物识别到位：

1. 外部用户：终端用户、管理员。
2. 文件或数据库等持久化存储设施：关系数据库、实时数据库、Flash、一般文件、分布式文件系统等。
3. 外部系统与底层硬件。
4. 时限触发机制。

2.2、到设计-如何分层

- UI交互层是否存在?
- UI交互层的职责包含哪几部分?
- 系统交互层是否存在、包含哪几部分?
- 数据管理层是否存在、包含哪几部分?
    - 没有持久化需要（Flash存储、文件、数据库），就不包含数据管理层。
- 问题领域层的职责包含哪几部分?
    - 问题领域层一定存在。

3、优缺点

3.1、优点

- 层次结构清晰，技术关注点在一定程度上被分离。
- 基于分层架构进行分工，程序员将分属UI组、数据库组、集成组等，有利于技能的专业化提升。

3.2、缺点

- 没有划分出细粒度模块。