**软件架构为谁而设计**

为用户设计：使用软件的人，需考虑功能的使用性

为客户设计：给我们钱做这个软件的人，考虑客户的约束条件

为开发人员设计：考虑开放质量

为管理人员设计：项目经理等，需考虑项目的管理，跟进等



**软件架构视图**

**什么是软件架构视图**

从某个角度，描述系统的组成的图

**多组涉众，多个视图**

对于不同的角色（如客户与程序员），他们掌握的技能不同，因此需要提供不同的视图

**逻辑架构与物理架构**

逻辑架构视图与物理架构视图是常用的架构视图

逻辑架构：

逻辑架构设计任务：

识别功能块

规划功能块的接口

明确功能块之间的关系

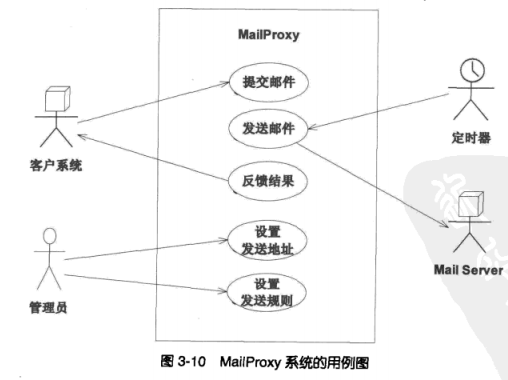
物理架构：

物理架构描述了组成软件系统的物理元素在硬件上的部署策略

**MailProxy案例**

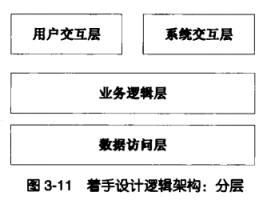
**MailProxy简介**

MailProxy具有代发客户邮件，接受管理员管理的功能，用例图如下：

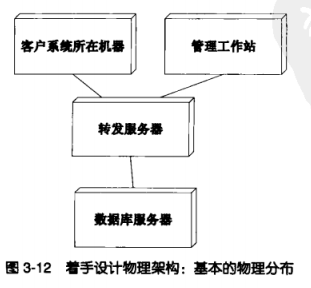


**逻辑架构设计迭代1**

MailProxy系统需要与人交互，还需要与外部系统交互，于是有了用户交互层（与管理员交互）与系统交互层（与客户系统和MailServer交互），初步设计如下：



**物理架构设计迭代1**



**逻辑架构设计迭代2**

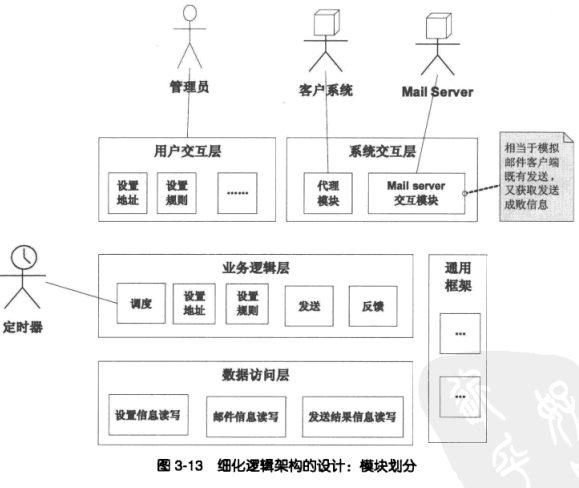
进一步细化逻辑架构

由于我们的系统需要与MailServer交互，所有我们新增一个“MailServer交互模块”

我们需要接收客户系统发来的邮件，但我们的产品一直迭代，我们不能因为我们产品的改变也要客户系统跟着改变，所有我们提供“代理模块”供客户系统使用

我们需要为管理员提供UI管理界面，所有在用户交互层提供“设置地址”，“设置规则”的UI模块

最终我们的逻辑架构如下：



根据物理架构提供的信息，我们将这些模块分为3个工程：

代理模块：包含代理模块，MailProxy系统专门发布给合作开发单位（客户系统）的API.

管理员web应用：包含“设置地址”，“设置规则”等UI模块，供管理员使用

MailProxy服务器程序：包含剩下的模块

注：同属于一个层并不一定部署在一台机器上

**物理架构设计迭代2**

有了详细的逻辑架构后，我们可以设计详细的物理架构

