**软件体系结构风格对比**

**一 数据流风格**

当输入数据经过一系列的计算和操作构件的变换形成输出数据时，可以应用这种体系结构。

管道/过滤器、批处理序列都属于数据流风格。

**管道/过滤器风格**：

把系统任务分成为几个独立的功能模块。这些功能模块采用通过系统的数据流连接。

在管道-过滤器风格下，每个功能模块都有一组输入和输出。功能模块称作过滤器（filters）；功能模块间的连接可以看作输入、输出数据流之间的通路，所以称作管道（pipes）。

管道-过滤器风格的特性之一在于过滤器的相对独立性，即过滤器独立完成自身功能，相互之间无需进行状态交互。

**管道/过滤器风格优点**：

设计者可以将整个系统的输入、输出特性简单的理解为各个过滤器功能的合成。

基于管道-过滤器风格的系统具有较强的可维护性和可扩展性

支持一些特定的分析，如吞吐量计算和死锁检测等

管道-过滤器风格具有并发性

**管道-过滤器风格缺点**：

交互式处理能力弱

具体实现比较复杂：数据流同步问题；数据加密与解析

**二 调用—返回风格**

主要包括：主程序/子程序；面向对象风格；层次结构

**层次结构风格**：

整个系统被组织成一个分层结构，每一层为上层提供服务，并作为下一层的客户。

**层次结构风格优点**：

层次结构风格支持系统设计过程中的逐级抽象

基于层次结构风格的系统具有较好的可扩展性

层次结构风格支持软件复用

**层次结构风格缺点**：

并不是每个系统都可以很容易地划分为分层的模式

很难找到一个合适的、正确的层次抽象方法。

**三 C/S风格**

客户机/服务器（C/S）体系结构是基于资源不对等，且为实现共享而提出来的。

C/S体系结构有三个主要组成部分：数据库服务器、客户应用程序和网络。

**C/S风格优点**：

界面和操作可以很丰富

安全性高

响应速度快

**C/S风格缺点**：

适用面窄

用户群固定

维护成本高

**四 B/S风格**

B/S体系结构主要利用不断成熟的WWW浏览器技术，特别是浏览器嵌入的多种脚本语言。

B/S体系结构有三个主要组成部分：浏览器、 Web服务器、数据库服务器。

**B/S风格优点**：

维护和升级方式简单

交互性较强

**B/S风格缺点**：

在速度和安全性上需要花费巨大的设计成本

通常需要刷新页面（Ajax等可以一定程度缓解该问题）

通信开销大  
**五 对等网络**

主要包括：文件共享网络

对等网络（P2P）体系结构中，每一个构件都只执行它自己的进程，并且对于其他同级构件，每个构件本身既是客户端又是服务器。

P2P的通信方式类似于客户-服务器体系结构中的请求/应答方式，但不同的是，在P2P中任意一个构件都可以向其他同级构件发送请求。

**对等网络风格优点**：

规模易于扩展

可以增加系统容量

对于构件和网络的故障具有很好的容错性

**对等网络风格缺点**：

如果文件内容经常变化，共享速度（如需要快速传输大型文件时）和文件的质量（如图像或视频）有着重要影响，或者如果一个端必须信任另一个端（如文件内容是受保护的或者包含了重要的合作信息），则P2P体系结构不是最佳选择。

**六 发布-订阅**

主要包括：隐含调用

在发布-订阅的体系结构中，构件之间通过对事件的广播和反应实现交互。如果一个构件对某个事件感兴趣则可订阅该事件，一旦该事件发生了，另一个构件则进行发布来通知订阅者。

和客户-服务器以及P2P的构件相比，发布-订阅构件对其他构件的存在一无所知，相反，发布者只是简单地宣布事件，然后等待反应；订阅者只是简单地对事件通知作出反应，而不管事件是如何发布的。

在共享环境下，这种体系结构风格是一种常用的集成工具。

**发布-订阅风格优点**：

支持系统演化和可定制性

构件便于复用

**发布-订阅风格缺点**：

可能减弱系统的可扩展性和可复用性

不易于测试

**我们所开发的博客网站项目使用Django这一Web开发框架，属于B/S设计风格。**