实验十 软件体系结构设计（二）

实验目的：

1. 体系结构风格和视图特点

2. 研究经典软件体系结构案例

3. 继续补充和修改自己项目的SAD

实验内容：

1. 对比书上各种软件体系结构风格和视图特点，思考自己项目属于哪种设计风格？

网上搜索最新的软件体系结构资料，如MVC、Kruchten 4+1视图等。

**Model-View-Controller (MVC)：**

**特点： 将系统分为三层，Model处理数据和业务逻辑，View负责呈现数据，Controller处理用户输入。**

**适用性： 博客系统有明显的用户界面展示（View），后台处理（Model）和用户交互（Controller），因此MVC是一种常见的选择。**

**优点： 分离关注点，便于维护和扩展。**

**缺点： 对于复杂的交互，Controller可能会变得复杂。**

**三层架构（Three-Tier Architecture）：**

**特点： 分为表示层（Presentation Layer）、业务逻辑层（Business Logic Layer）和数据层（Data Layer）。**

**适用性： 适合于需要清晰分层处理的系统，比如一个博客系统，其中表示层处理用户请求和展示内容，业务逻辑层处理业务规则，数据层负责数据存取。**

**优点： 清晰的层次分离，便于扩展和维护。**

**缺点： 可能会有一些性能上的开销。**

**面向服务的架构（Service-Oriented Architecture, SOA）：**

**特点： 系统被分解为一系列松耦合的服务，每个服务都有明确的接口和功能。**

**适用性： 如果博客系统需要与其他系统进行集成，或希望未来可以扩展为微服务架构，SOA是一个不错的选择。**

**优点： 高度的可复用性和灵活性。**

**缺点： 服务间通信可能带来额外的复杂性和性能开销。**

**事件驱动架构（Event-Driven Architecture, EDA）：**

**特点： 系统通过事件进行驱动和通信，组件间通过事件通知进行交互。**

**适用性： 适合需要实时响应和处理用户操作的博客系统。**

**优点： 灵活的事件处理机制，适用于实时和高并发场景。**

**缺点： 事件处理逻辑可能较复杂。**

**我们的项目属于三层架构。**

2. 参阅课本和网上资料，研究经典软件体系结构案例KWIC。

An Introduction to Software Architecture，4.1节

On-the-Criteria-To-Be-Used-in-Decomposing-Systems-into-Modules（Example System 1）

http://www.cs.cmu.edu/~ModProb/index.html

针对KWIC和自己项目，参考课本ch5 表5-3，小组成员每人给几种不同的体系结构风格设计打分，评最佳。

KWIC（Key Word in Context）系统经典案例展示了如何根据不同的软件体系结构风格来设计和实现系统。以下是对KWIC案例的不同设计风格的分析：

主程序/子程序架构（Main Program/Subroutine）：

特点： 使用一个主程序来控制多个子程序。

优点： 结构简单，容易实现。

缺点： 主程序与子程序耦合度高，修改其中一部分可能需要修改整个系统。

面向对象架构（Object-Oriented Architecture）：

特点： 系统由多个对象组成，每个对象包含数据和处理数据的方法。

优点： 低耦合，高内聚，便于扩展和维护。

缺点： 设计和实现较复杂。

流水线架构（Pipeline Architecture）：

特点： 将系统分解为多个独立的处理阶段，每个阶段通过流水线连接。

优点： 每个阶段独立，便于并行处理和扩展。

缺点： 流水线设计需要处理好阶段间的依赖和数据流动。

隐式调用架构（Implicit Invocation Architecture）：

特点： 组件通过事件驱动进行通信。

优点： 高度模块化，组件间低耦合。

缺点： 事件处理逻辑复杂，调试困难。



根据打分情况，MVC由于其清晰的分层和广泛的使用，可能是博客系统项目的最佳选择。

3. 补充和修改自己项目的SAD

记录项目及小组每个人工作的进度、里程碑、工作量的跟踪图或表，将其保存到每个小组选定的协作开发平台上，每周更新。

