|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 属性 | 优先级 | 黑板模式 | 分层模式 | 管道-过滤器模式 | 浏览器-服务器模式 | 模型-视图-控制器 |
| 好的性能 | 2 | 1 | 2 | 1 | 5 | 4 |
| 易于修改 | 5 | 1 | 5 | 4 | 3 | 4 |
| 好的安全性 | 3 | 1 | 5 | 4 | 5 | 5 |
| 易于复用 | 4 | 1 | 5 | 4 | 3 | 5 |
| 有效的数据表示 | 1 | 1 | 3 | 3 | 2 | 3 |
| 总分 |  | 15 | 67 | 53 | 54 | 66 |

## 评价说明

### 优先级

一个Web应用, 往往会随着用户需求而迭代版本, 这就要求对应用的易修改性有较高要求; 其次, 在开发新功能时, 能复用之前的逻辑或代码能减少开发人员的工作量, 并且不易出错, 易于复用也很重要; 安全性排第三位, 用户的隐私和网站的内部逻辑需要仔细保护, 防止泄露和被破解; 性能和数据表示往往是固定的, Web应用的性能取决于服务器和代码逻辑, 前者和后者与其他应用相比, 往往差距不大, 数据表示就是键值对, 非常固定.

### 模式

我们对该应用有确定的方案, 黑板模式各方面都不适用.

分层模式是Web应用的基本模式, 分层模式的安全性, 易修改性和易复用性使开发变得便捷, 并且各个层的数据表示独立, 底层数据表示更灵活, 但层数过多会导致系统冗余或性能下降.

管道过滤器模式适用于数据流, 而我们的Web应用是事件驱动的, 但管道过滤器模式在安全, 易修改和易复用方面仍然有很大诱惑力, 但每个过滤器都需要对数据进行全部扫描, 性能堪忧.

Web应用要么选择C/S要么选择B/S, 我们的应用基于大众, 选择B/S, HTTPS的安全性非常可靠, 并且没有中间件, 系统轻便, 性能高.

模型-视图-控制器模式是最近几十年Java Web应用的宠儿, 该模式把表现层, 业务逻辑层, 数据层分开, 使得业务代码不用和前端代码混合在一起, 可维护性和灵活性都提高了很多. 三个模块之间接口固定, 模块独立易于修改, 安全性高, 完全使用模型访问数据库, 数据表示更灵活.