候选体系结构

解决方案与风险分析

**团队名称：Anti-Gay-Fri 第26组**

**团队成员：PM:姜殊 成员：陈元杰 耿昱翔 金炜 莫其凡**

|  |  |  |
| --- | --- | --- |
| 变更时间 | 变更说明 | 变更人 |
| 2012/3/18 | **文档创建** | **耿昱翔** |
|  |  |  |
|  |  |  |

1. 系统架构要解决的问题

|  |  |
| --- | --- |
| 问题 | 问题猫鼠 |
| 软件架构 | 客户端需要界面、数据应保存在服务器上 |
| 操作响应 | 任何操作应该在四秒内响应 |
| 部署 | 软件需要部署在异地 |
|  |  |

|  |  |
| --- | --- |
| 约束与限制 | 描述 |
| 1 | 网络环境为局域网10M/S |
| 2 | 操作环境为WINDOWS |
| 3 | 提供普通电脑用作保存数据服务器 |
| 4 | 系统必须支持多人模式会议 |
| 5 | 系统供软件开发人员使用，且分布在异地 |
| 6 | 系统供讨论系统设计的细节 |

二、候选体系结构

|  |  |
| --- | --- |
| 候选体系结构解决方案1 | 描述 |
| 分层 | 采用分层结构。界面+逻辑+网路+数据结构 |
| MVC | 模块+视图+控制 |

1. 风险分析

|  |  |
| --- | --- |
| 候选体系结构 | 风险 |
| 分层  JAVA+MySQL+RIM | 组员都有过基本的分层开发经验。此项风险较低  1、每个层由一组相关的类或组件构成，共同完成特定的功能。  2、层与层之间存在自上而下的依赖关系，上层依赖下层。  3、每个层对上层公开API,但具体的实现细节对外是透明的，当某一层的实现发生变化时，只要API不变，就不会影响到其他层的实现。  二、优点  1、可伸缩性好  2、可维护性、灵活性，层与层之间耦合度低  3、可扩展性好  4、可重用性好  5、可管理性，程序分层后可以降低系统管理的难易程序，将程序分为多层后，可以将工作分解给不同的开发小组，从而便于管理，系统越复杂、规模越大，分的层数就需越多。    **风险**        软件分层越多，对软件设计人员的要求也就越高，在设计阶段，必须花更多的时间构思合理的体系结构，如考虑如何划  分层、定义每个层的职责；如果在体系结构方面存在缺陷，比如，层与层之间出现自下而上的依赖关系，一旦业务逻辑发生变化，可能表示层的代码也要修改。    容易变更需要修改，但层与层之间的相对剥离容易应对迭代开发的风险 |
| MVC  JAVA+MySQL+RIM | 风险  没有明确的定义、内部原理比较复杂，需要花费一些时间去思考。模型和视图要严格的分离，调试应用程序带来了一定的困难。每个构件在使用之前都需要经过彻底的测试。MVC并不适合小型甚至中等规模的应用程序。花费大量时间将MVC应用到规模并不是很大的应用程序通常会得不偿失。  风险！一旦发生变更整个MVC模型都要改动风险成本较高不适合迭代开发  组员都没有MVC开发经验，学习成本较高也不易掌握容易失败 |
|  |  |