实验六

郑家彤 202100032018

工作量估算，风险管理，软件需求规格说明SRS（1）

实验目的：

1. 工作量估算

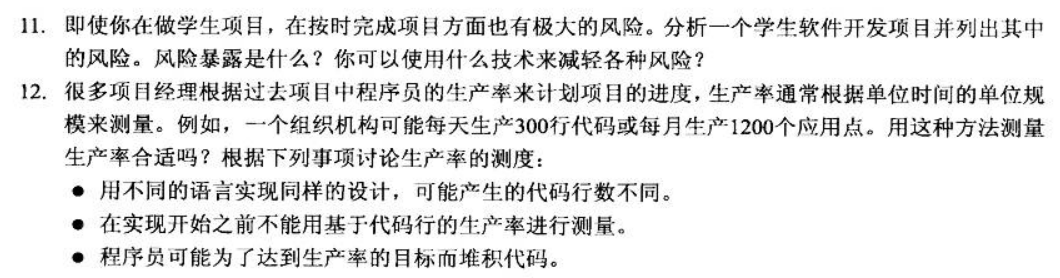
2. 风险管理

3. 学习软件需求规格说明SRS文档的要求和结构

实验内容：

1.工作量估算：

(1)ch3 习题12（小组讨论）。



* 不同编程语言的差异：不同的编程语言有着不同的特性和抽象层次，这意味着实现相同功能的设计，使用不同的语言可能会导致代码行数有显著差异。例如，使用更高级、抽象层次更高的语言（如Python或Ruby）可能需要较少的代码行数，而低级语言（如C或汇编语言）可能需要更多的代码来完成同样的任务。因此，仅以代码行数作为生产率的指标，并不能公平地反映出实际的工作量和技术选择的优势。
* 前期无法准确预测：在软件开发的初期阶段，尤其是需求分析和设计阶段，很难甚至不可能准确地基于代码行来预估生产率。这是因为在这个阶段，项目的具体实现细节、技术选型、以及设计的优化程度都还不确定。因此，过早地使用代码行作为生产率指标，可能会导致计划过于理想化，忽视了实际开发过程中的不确定性。
* 堆积代码：如果程序员的绩效考核过于侧重于代码行数的产出，这可能会激励他们采用非最优的编码实践，比如不必要的代码重复、过度注释或者编写冗长而不必要的代码，仅仅是为了提高所谓的“生产率”。这种做法不仅会降低代码质量，增加维护成本，还可能损害软件的性能和可读性。

(2)参考书3.7（P94)皮卡地里电视广告销售系统按COCOMOII的工作量模型进行工作量估算的例子（结合P79-80表），估算自己项目的初始工作量。

企业用户子系统

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 比例因子 | 非常低5 | 低4 | 一般3 | 高2 | 非常高1 | 极高0 |
| 先例 |  |  |  | √ |  |  |
| 灵活性 |  | √ |  |  |  |  |
| 去除的重大风险 |  |  |  | √ |  |  |
| 团队交互过程 |  |  |  | √ |  |  |
| 过程成熟度 |  | √ |  |  |  |  |

预计初始工作量为30人日，约1.5人月，比例因子=2+4+2+2+4=14，计算比例指数为1.01+0.01\*15=1.15 ，即预计工作量调整为1.51.15=1.594 人月。

个人用户子系统

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 比例因子 | 非常低5 | 低4 | 一般3 | 高2 | 非常高1 | 极高0 |
| 先例 |  |  |  |  | √ |  |
| 灵活性 |  | √ |  |  |  |  |
| 去除的重大风险 |  |  |  |  | √ |  |
| 团队交互过程 |  |  |  | √ |  |  |
| 过程成熟度 |  | √ |  |  |  |  |

初始预计工作量为20人日，约1人月，计算的比例因子为 1+4+1+2+4=12，因此计算比例指数为1.01+0.01\*12=1.13 ，即预计工作量调整为1^{1.13}=1 人月。

客服管理子系统

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 比例因子 | 非常低5 | 低4 | 一般3 | 高2 | 非常高1 | 极高0 |
| 先例 |  |  | √ |  |  |  |
| 灵活性 | √ |  |  |  |  |  |
| 去除的重大风险 |  |  | √ |  |  |  |
| 团队交互过程 |  |  |  | √ |  |  |
| 过程成熟度 | √ |  |  |  |  |  |

预计初始工作量为40人日，即约2人月，计算的比例因子为$ 3+5+3+2+5=18 ，因此计算比例指数为 1.01+0.01\*18=1.19 ，即预计工作量调整为 21.17=2.28人月。

2. 风险管理

ch3 习题11 （小组讨论）

**风险及缓解**

缺乏经验: 学生开发者可能缺乏实际项目管理和软件开发的经验，这可能导致低估任务难度、时间规划不合理或技术选型不当。

缓解技术: 提供培训工作坊，引入导师制度，鼓励学习现有开源项目或案例研究。

团队协作问题: 缺乏有效沟通、角色不明确或团队成员间的冲突可能严重影响项目进度。

缓解技术: 定期举行团队会议，明确分配角色和责任，使用项目管理工具（如Trello, Jira）增强协作，开展团队建设活动。

时间管理不当: 学业压力、兼职工作或其他个人事务可能干扰项目进度。

缓解技术: 制定灵活但明确的时间表，设定可达成的短期目标，鼓励提前完成任务以应对不可预见的延误。

技术挑战: 选择的技术栈过于复杂或不稳定，可能导致开发难度大增，甚至技术方案需中途变更。

缓解技术: 进行技术预研，选择成熟稳定的技术栈，预留技术缓冲时间以适应学习和调整。

需求不明确或频繁变更: 项目初期对需求理解不足或客户（如指导老师、项目发起者）在项目进程中频繁更改需求。

缓解技术: 在项目启动时进行详细的需求分析，制定需求变更管理流程，确保所有变更都经过评估和文档化。

资源限制: 硬件、软件或资金资源的不足可能制约项目进展。

缓解技术: 早期识别所需资源，寻求学校或外部赞助，利用云服务和开源工具降低成本。

测试与质量控制不足: 缺乏充分的测试可能导致软件存在大量bug，影响最终交付质量。

缓解技术: 实施持续集成/持续部署(CI/CD)，自动化测试，定期代码审查，确保质量控制贯穿整个开发周期。

分析自己项目中可能存在的风险，并进一步细化风险管理（做出风险分级及应对预案）。

3. 学习国标中的文档11《软件需求规格说明SRS》，了解文档的要求和结构及与其他相关文档 （07,08,12,17等）的关系。（文档见实验1的压缩文件）

项目跟踪，建立能反映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，每周更新。

（1）软件需求规格说明(SRS) - 此文档详细阐述了计算机软件配置项（CSCI）的具体需求，包括软件的外部接口、内部接口、数据、功能、性能、能力、环境、质量因素、设计与实现约束、安全保密性需求等。它为CSCI设计和合格性测试提供基础，并强调了每个需求的可追踪性。

（2）配置管理计划(SCMP) 定义了软件配置管理策略，包括如何在项目中实现配置项的标识、控制、状态记录、报告、检查、评审以及配置项的备份等。它涉及配置管理的机构、任务、职责、接口、控制方法、适用标准和约定、实现的里程碑。并管理供货单位、工具、记录、维护和备份策略。

（3）数据需求说明(DRD) 描述了项目中数据的逻辑需求，包括静态和动态数据的处理、输入输出、输入、输出、内部生成、内部生成数据和配置项的详细信息。它还涉及数据采集的控制、处理、存储、报告、影响、标准、命名规则和数据生命周期管理，以及软件配置。

（4）接口需求说明(IRS) 文档专注于系统、子系统、配置项之间接口需求，包括硬件、软件、手工操作的接口。它详细说明了接口的标识、图、承担者、类型、数据元素特性、协议、处理要求、通信方法和采集范围、承担者等。为设计与测试和合格性提供补充基础。

（5）系统需求规格说明(SSS) 文档概述了系统或子系统需求，包括系统性质、功能、硬件和软件需求、接口、配置、操作环境、安全保密性、维护性需求、故障处理、用户需求。文档提供了系统概述、引用文件、机构、任务、职责、标准和实现的里程碑。为设计和测试提供基础。