实验六 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：**

**ch3 习题12。**

很多项目经理根据过去项目中程序员的生产率来计划项目的进度，生产率通常根据单位时间的单位规模来测量。例如，一个组织机构可能每天生产300行代码或每月生产1200个应用点。用这种方法测量生产率合适吗?根据下列事项讨论生产率的测度:

●用不同的语言实现同样的设计，可能产生的代码行数不同。

●在实现开始之前不能用基于代码行的生产率进行测量。

●程序员可能为了达到生产率的目标而堆积代码。

1）对于一个功能点，使用不同的语言种类进行设计产生的代码量差异巨大。以C和HTML语言为例，C语言实现一个功能点需要130行代码，HTML语言实现一个功能点则只需要15行代码，因此，不能简单的按照每天生产的代码行和每月生产的应用点来判断生产率。

2）在实现开始之前，虽然已经对工程量进行了估计，但项目的难度和具体的难点可能还未知，设计和需求可能在开发过程中发生变化。比如开发一个OS内核的速度显然要慢于开发一个简单的web项目，以程序员过去写代码的速度估计本项目的速度不一定合适。因此，在项目开始之前使用基于代码行的生产率进行测量可能不具备实际意义。

3）程序员可能会倾向于写更多的代码以达到目标，而不是专注于编写高质量的、精简的代码。这可能会导致代码质量下降，增加维护成本。

**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。**

目前需要实现的主要为三个模块，分别为“个人用户子系统”、“企业用户子系统”和“客服管理子系统”三大部分。根据小组的项目活动图图得出初始的预计工作量，再结合COCOMOII模型分别调整了这三个模块的预计工作量。考虑到按照关键路径执行时部分工作为并行推进的，因此各模块计算应用点时将有一定的差异。如下为三个模块的评估：

**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 $ 3+4+2+3+4=16，因此计算比例指数为，因此计算比例指数为1.01+0.01\times16=1.17 ，即预计工作量调整为，即预计工作量调整为1^{1.17}=1 $ 人月。

**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，计算的比例因子为$ 2+5+2+2+4=15 ，因此计算比例指数为1.01+0.01\times15=1.16 ，即预计工作量调整为1.5^{1.16}=1.6 $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$ 4+5+3+3+5=20 ，因此计算比例指数为 1.01+0.01\times20=1.21 ，即预计工作量调整为 2^{1.21}=2.31 $人月。

**2. 风险管理**

**ch3 习题11**

即使你在做学生项目，在按时完成项目方面也有极大的风险。分析一个学生软件开发项目并列出其中的风险。风险暴露是什么?你可以使用什么技术来减轻各种风险?

风险：

1. 时间管理风险：学生可能会低估项目所需的时间，导致项目延期。

2. 技术能力风险：学生可能缺乏必要的技术能力来完成项目。

3. 沟通风险：团队成员之间沟通不畅，导致信息不对称和决策延迟。

4. 范围蔓延风险：项目范围可能会不断膨胀，导致需求不明确和项目无法按时完成。

5. 资源限制风险：学生可能面临时间、人力、技术和资金等资源的限制。

减轻风险技术：

1. 制定详细的项目计划，并进行时间管理，确保按时完成关键里程碑。

2. 提前评估团队成员的技术能力，进行技术培训和知识分享，以提高整体技术水平。

3. 建立有效的沟通机制，如定期会议、沟通工具等，确保团队成员之间的信息流畅。

4. 确定明确的项目范围，制定变更管理流程，防止范围蔓延。

5. 确保有足够的资源支持项目，如合理分配任务、提供必要的技术设备和资金支持。

**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）。**

|  |  |  |
| --- | --- | --- |
| Risk | Exposure | Mitigating actions |
| 初期需求分析不够充分 | （.30）(10)=3 days | （1） 让用户参与开发，提供一个协作开发环境，通过线上方式让用户切身参与到开发中来。并采取相应的激励措施，以激发用户参与的积极性；（2）开发用户界面原型：系统分析员需要借助白板、白纸等沟通方式，帮助用户清楚表述需求。然后，开发一个用户界面原型，以便用户确认需求。用户界面原型的作用仅；（3）开需求讨论会议：集中各个开发人员的的想法和开发愿景，举办一次需求研讨会；（4）强化需求分析与评审 |
| 项目开发缺少可见性 | (.20)(30)=6 days | （1） 迭代开发：将产品的交付过程分为多个阶段，按照功能递增式交付，每次迭代，都要充分接收用户的评审意见，以便为自我纠正。渐近式的功能交付，有利于降低开发人员的压力，增加用户的满意度，有利于增强项目的可见性，是最好的进展报告；（2）技术评审：持续集成把最终的一次大规模的集成调试过程分散到项目开发时间表的每一周、每一天、甚至每个小时。让项目中的各个人员都能够随时掌握当前的整体进度，并迅速发现集成过程中出现的问题并进行解决 |
| 新技术的引入，对于云服务器的应用经验不足 | (.10)(10)=1 days | (1)T形软件开发：在项目开发早期，开发小组应该建立系统的架构，解决关键技术难题、开发系统的基础构件，并对系统所需要应用的技术做深度探索；(2)充分论证：新技术开发是探索性很强的工作，潜在着许多失败的风险。在可行性分析阶段，要广泛搜集相关信息，设计多种可行方案，进行充分论证。在制定决策时，情报的数量和质量致关重要。掌握的信息越多、越准确，才能作出正确的的决策，项目失败的风险也就相对减少；反之，承担的风险就会增大；(3)同行经验：在探索过程中要充分利用互联网，通过搜索已有的经验 |
| 技术兼容性风险 | (.01)(5) =0.5 days | (1)设计先行 |
| 性能问题，如网站在不同浏览器上的显示兼容性问题 | (.01)(5) =0.5days | (1)性能规划：在系统设计时，应做好前期做性能规划，对可能出现性能问题的环节做到充足的估计；(2)性能测试：在开发过程中，要重视性能测试和压力测试，尽可能模拟现实使用环境，搭建测试平台。另外，由于开发环境的计算机往往比生产环境的计算机配置高，在做测试时应尽量找一些配置低的机器、较小的网络带宽进行测试；(3)充足的调试时间：在项目开发计划中，为后期性能优化留有余地。在对系统进行性能优化后，要进行性能测试和压力测试，可能还要做几次回归测试。因此，应该留有充足的时间和人力 |
| 可用性问题 | (.20)(10)= 2 days | (1)了解用户：到用户工作现场，了解目标用户使用软件的真实目的，从用户的角度、从用户的立场出发，了解如何通过软件系统替代用户的业务处理流程中，最繁琐、最容易出问题、或者是大量重复劳动的环节，让软件提高用户的工作效能和效率。；(2)参与型设计：与用户协作，让用户参与用户界面的设计、评审与测试，确保用户能够全面地、及早地发现可用性等方面的问题，并及时纠正；(3)竞争性分析：通过对市场上同类竞争性产品进行分析，或者对这些产品进行实验性测试，了解这些产品的用户界面问题，从而对新系统的开发提供启发 |

**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）**

在软件工程中，软件需求规格说明（Software Requirements Specification，SRS）是非常重要的文档，用于描述软件系统的功能需求、性能需求、非功能需求等方面的详细规格。

在国标中，不同的文档之间存在一定的关系，包括但不限于以下几种：

1. 文档07《软件设计规格说明SDS》：SRS中描述的需求将为SDS提供基础，SDS会进一步详细描述系统的设计方案和架构。

2. 文档08《软件测试计划STP》：SRS中描述的需求将作为软件测试计划的依据，帮助测试团队设计测试用例和测试策略。

3. 文档12《软件配置管理计划SCMP》：SRS中的需求描述也会影响到软件的配置管理，SCMP将规定如何管理软件的版本、变更等配置信息。

4. 文档17《软件维护手册SMM》：SRS中记录的需求信息对于软件维护也是非常重要的，SMM将指导维护团队如何管理和更新软件。

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。