## 风险评估

软件项目风险是指在整个项目周期中所涉及的成本预算、开发进度、技术难度、经济可行性、安全管理等各方面的问题，以及由这些问题而对项目所产生的影响。项目的风险与其可行性成反比，其可行性越高，风险越低。软件项目的可行性分为经济可行性、业务可行性、技术可行性、法律可行性等四个方面。而软件项目风险则分为产品规模风险、需求风险、相关性风险、管理风险、安全风险等六个方面：

#### 产品规模风险

项目的风险是与产品的规模成正比的，一般产品规模越大，问题就越突出。尤其是估算产品规模的方法，复用软件的多少，需求变更的多少等因素与产品风险息息相关：

1. 估算产品规模的方法
2. 产品规模估算的信任度
3. 产品规模与以前产品规模平均值的偏差
4. 产品的用户数
5. 复用软件的多少
6. 产品需求变更的多少

#### 需求风险

很多项目在确定需求时都面临着一些不确定性。当在项目早期容忍了这些不确定性，并且在项目进展过程当中得不到解决，这些问题就会对项目的成功造成很大威胁。如果不控制与需求相关的风险因素，那么就很有可能产生错误的产品或者拙劣地建造预期的产品。每一种情况对产品来讲都可能致命的，这些的风险因素有:

1. 对产品缺少清晰的认识
2. 对产品需求缺少认同
3. 在做需求分析过程中客户参与不够
4. 没有优先需求
5. 由于不确定的需求导致新的市场
6. 不断变化需求
7. 缺少有效的需求变化管理过程
8. 对需求的变化缺少相关分析等

#### 相关性风险

许多风险都是因为项目的外部环境或因素的相关性产生的。控制外部的相关性风险， 能缓解策略应该包括可能性计划，以便从第二资源或协同工作资源中取得必要的组成部分，并觉察潜在的问题,与外部环境相关的因素有：

1. 客户供应条目或信息
2. 交互成员或交互团体依赖性
3. 内部或外部转包商的关系
4. 经验丰富人员的可得性
5. 项目的复用性

#### 技术风险

软件技术的飞速发展和经验丰富员工的缺乏，意味着项目团队可能会因为技巧的原因影响项目的成功。 在早期，识别风险从而采取合适的预防措施是解决风险领域问题的关键，比如：培训、聘请顾问以及为项目团队招聘合适的人才等。关于技术主要有下面这些风险因素：

1. 缺乏培训
2. 对方法、工具和技术理解的不够
3. 应用领域的经验不足
4. 对新的技术和开发方法应用不熟悉

#### 管理风险

尽管管理问题制约了很多项目的成功，但是不要因为风险管理计划中没有包括所有管理活动而感到惊奇。在大部分项目里，项目经理经常是写项目风险管理计划的人，他们有先天性的不足——不能检查到自己的错误。因而，使项目的成功变得更加困难。如果不正视这些棘手的问题，它们就很有可能在项目进行的某个阶段影响项目本身。当我们定义了项目追踪过程并且明晰项目角色和责任，就能处理这些风险因素：

1. 计划和任务定义不够充分
2. 对实际项目状态不了解
3. 项目所有者和决策者分不清
4. 不切实际的承诺
5. 不能与员工之间的进行充分地沟通

#### 安全风险

软件产品本身是属于创造性的产品，产品本身的核心技术保密非常重要。但一直以来，我们在软件这方 面的安全意识比较淡薄，对软件产品的开发主要注重技术本身，而忽略了专利的保护。软件行业的技术人员流动是很普遍的现象，随着技术人员的流失、变更，很能会导致产品和新技术的泄密，致使我们的软件产品被它公司窃取，导致项目失败。而且在软件方面关于知识产权的认定目前还没有明确的一个行业规范，这也是我们 软件项目潜在的风险。

#### 回避风险的方式

1. 以开发方诱导能保证需求的完整，使需求与客户的真实期望高度一致。再以书面方便形成《用户需求》这一重要的文档，避免疏漏造成的损失在软件系统的后续阶段被逐步地放大。
2. 设立监督制度，项目开发中任何较大的决定都必须有客户参与进行的，在该项目中项目监督由项目开发中的质量监督组来实施。
3. 需求变更需要经过统一的负责人提出,并且要用户需求的审核领导认可,需求变更应该是定期而不是随时的提出,而且开发方应该做好详细的记录,让客户了解需求变更的实际情况。
4. 控制系统的复杂程度，过于简单的系统结构，对用户来使用比例会有明显的折扣，甚至造成软件寿命过短。反之，软件结构的过于灵活和通用，必然引起软件实现的难度增加，系统的复杂度会上升，这又会在实现和测试阶段带来风险。适当控制系统的复杂程度有利于降低开发的风险。
5. 从软件工程的角度看，软件维护费用约占总费用的55%~70%，系统越大，该费用越高。对系统可维护性的轻视是大型软件系统的最大风险。在软件漫长的运营期内，业务规则肯定会不断发展，科学的解决此问题的做法是不断对软件系统进行版本升级，在确保可维护性的前提下逐步扩展系统。
6. 设定应急计划，每个开发计划都至少应该设定一个应急预案去应对出现突发情况和不可遇知的风险。

## 成本预算

#### 成本预算方式

1. 自上而下的预算方法

自上而下的预方法主要是依据上层、中层项目管理人员的管理经验进行判断，对构成项目整体成本的子项目成本进行估计，并把这些判断估计的结果传递给低一层的管理人员，在此基础上由这一层的管理人员对组成项目的子任务和子项目的成本进行估计，然后继续向下一层传递他们的成本估计，直到传递到最低一层。

使用此预算方式，在上层的管理人员根据他们的经验进行的费用估计分解到下层时，可能会出现下层人员认为上层的估计不足以完成相应任务的情况。这时，下层人员不一定会表达出自己的真实观点，不一定会和上层管理人员进行理智地讨论，从而得出更为合理的预算分配方案。在实际中，他们往往只能沉默地等待上层管理者自行发

现问题并予以纠正，这样往往会给项目带来诸多问题。

自上而下更适用于项目启动的前期，与真实费用相差在30% ~ 70%之间。

Scrum使用自上而下的成本预算方式，它不会立即精确地确定成本，而是以最大限度容纳客户对未来产品要求所

产生的变更。

1. 自下而上的预算方法

自下而上方法要求运用WBS（Work Breakdown Structure，工作分解结构）对项目的所有工作任务的时间和预算进行仔细考察。最初，预算是针对资源（团队成员的工作时间、硬件的配置）进行的，项目经理在此之上再加上适当的间接费用（如培训费用、管理费用、不可预见费等）以及项目要达到的利润目标就形成了项目的总预算。自下而上的预算方法要求全面考虑所有涉及到的工作任务，更适用于项目的初期与中期，它能准备地评估项目的成本，与真实费用相差在5% ~ 10%之间。

***注解：WBS***

*WBS是面向提交成果对项目的分解，从提交成果的列表可以确定每个提交成果需要执行的活动。Scrum会对WBS进一步细化，把每个迭代分解为更细小的工作包。*

#### 确定项目支出

总体成本预算就是结合下列多个成本预算方式，组成开发的总体成本：

1. 零基数预算

在成本预算的初期应该使用零基数的计算原则，而不可以使用类似于：以上一年总体费用加上20% 这样粗略的方式计算项目成本。

1. 软硬件成本、物品成本

物品成本是指类似于：服务器（RAM 硬盘 CPU NIC卡 RAID簇）成本、维护成本、机房租金、光纤通讯成本、软件成本等的成本。

计算成本时需要考虑组装硬盘需时的长短，技术人员需要具备的质素，产品供应商能否提供保证质量，管理时是否需要额外的管理人员这些多方因素。

1. 软件许可证成本
2. 外包成本

当使用类似：视频、短信、移动电信类服务、门户网站等子项目时可以考虑以外包形式完成，以降低开发成本。

1. 人力资源成本

计算人力资源成本时应该使用以最高和最低的工作效率估算平均效率的方式，计算出人力资源的平均成本。

1. 维修保养成本

## 客户沟通的过程

从客户沟通的方向出发来看，软件项目可分为：需求识别、方案定制、项目实施、项目结束等4个不同的阶段，各个阶段都具有不同的沟通重点。

#### 需求识别阶段

1. 文本沟通

在需求识别的前期，应该通过问卷、原型展示、界面展示、逻辑处理展示、准化文档模板等方式进行全方位多角度的分析，随时将不明确之处反馈给客户,以期待客户解答。并以文本记录的方式建立需求分析书，并要求客户审核需求分析书，以达到需求分析与客户的真实期望高度一致的结果。

1. 业务逻辑沟通

在进行业务沟通时，应该了解客户的行业语言，以促进业务分析的过程，越过应用需求和开发之间的鸿沟。沟通过程提倡以草图或者可视信息化的方式进行,针对不同层面的企业用户提供最适合的操作界面。以多角度的方式思考问题，要抓住需求重点,尤其是客户方领导所关注的创新类和实用类需求。

1. 需求变更的规范化管理

需求变更在软件开发类项目中是可以理解的,但必须对需求变更做好规范化的管理，以避免出现需求无止境变更的风险。需求变更必须由统一的负责人提出,并且由用户需求的审核领导者认可。需求变更的提出应该是定期而不是随时的，开发方应该做好详细的文本记录,让客户了解需求变更的实际情况和开发方为之所付出的成本代价。

#### 方案定制阶段

该阶段项目的主要任务是与客户共同制定一个以前期明确的需求、双方的资源、项目开始的阶段、实施的时间约定、项目费用限制等为基础的具有可操作性的项目计划,从本阶段开始争取客户全面参与项目的管理，并以双方的共同利益考虑项目实施的具体计划与风险规避。

#### 项目实施阶段

在该阶段,软件项目团队应该与客户共同领导项目的实施。同时,项目团队应实时评估客户满意度,并通过持续改进的方式提高客户满意度,还应要求客户参加必要的培训,以及在必要时检查项目产品。在出现客户的需求变更前,应主动与客户沟通交流,使客户充分了解项目的每个环节,以及变更带来的影响,减少需求变更。如果出现客户需求变更,应与客户一起共同解决由变更引起的成本、进度、质量变化。

#### 结束阶段

该阶段主要进行项目成果的移交，并把系统交付给维护人员，帮助客户实现商务目标，结清各种款项。完成这些工作后应该进行项目评估，审核此项目的成果并总结项目经验。

#### 售前人员注意事项

在产品型项目作为开发成果时，相关销售人员应该注意：对产品的推销不应该过分承诺。如果过分承诺,会给后续的项目实施带来困难;一旦承诺没有兑现,也会降低客户满意度,影响今后合作。如果有附加承诺,一定要以文本形式记录，让实施项目经理知晓并传达给项目组成员。

***注解：在软件项目中,需要明确以下四种客户角色***

1. *要明确最终使用部门和用户,要去了解他们现有的工作方式,要让他们知道项目的目标框架,知道项目要解决他们的哪些困难,但绝对不是全部困难,这样可以较好的控制项目范围。*
2. *要明确需求的提出者,他或者他们要能够代表最终客户群体。提出产品需求的这类客户要具有一定的技术、业务能力和权威,能够真正代表最终客户团队的意愿和想法,最好有IT基础,能够用IT语言描述问题和需求,以利于双方的沟通、协作,避免产生歧义。*
3. *要明确做需求确认的中层领导,他要把握方向。软件开发项目是解决实际生产或者管理问题,同时 也是领导系统建设的具体实现,做需求确认的客户领导,既要了解高层领导的系统建设要点和方向,又要谙熟具体业务和生产管理实际。如果是这样的客户领导来把 握和决策,对企业软件开发项目的顺利进展作用非凡。*
4. *要明确谁来对成品提意见,谁来验收。项目验收环节,是项目的收尾环节,如果验收的人对项目初期的需求目标不了解,会从态度和产品实际使用效果上对验收产生负面的影响,对提供产品的企业关闭项目非常不利。根据实践总结,由需求提出人和确认人来做项 目的验收工作,能够促进项目的顺利完成,避免延期。*

## 需求分析

#### 需求分析的过程

需求过程包括需求开发和需求管理2个部分：

1. 需求开发就是对开发前期的管理，与客户的沟通过程，可以分为4个阶段：需求获取、需求分析、编写需求

和需求验证。

1. 需求管理：就是软件项目开发过程中控制和维持需求约定的活动。包括：变更控制、版本控制、需求跟踪、需求状态跟踪。

#### 需求的层次

需求的层次包括：业务需求、用户需求、功能需求、非功能需求等4个方面。

#### 需求开发阶段的重点

1. 提取业务对象

业务对象是指系统使用的真实对象，例如一个供应链管理(Supply Chain Management ,简称SCM)业务对象主要包括：生产批发商、零售商、送货商、顾客多个层次。

1. 提取业务流程

在了解业务逻辑的过程中，应该列举出所开发软件模块的各自职能，并细化每个工作流程，深入分析业务逻辑。

1. 性能需求

在分析的前期应该注意客户对所开发软件的技术性能指标，如存储容量限制、运行时间限制、安全保密性等。

1. 环境需求

环境需求是指软件平台运行时所处环境的要求，如硬件方面：机型、外部设备、数据通信接口;软件方面：系统软件，包括操作系统、网络软件、数据库管理系统方面;使用方面：使用部门在制度上，操作人员上的技术水平上应具备怎样的条件。

1. 可靠性需求

对所开发软件在投入运行后发生故障的概率，应该按实际的运行环境提出要求。对于重要的软件，或是运行失效会造成严重后果的软件，应提出较高的可靠性要求。

1. 安全保密要求

在需要分析时应当在这方面恰当地做出规定，对所开发的软件给予特殊的设计，使其在运行中，其安全保密方面的性能得到必要的保证。

1. 用户界面需求

为用户界面细致地规定到达的要求。

1. 资源使用需求

开发的软件在运行时和开发时所需要的各种资源。

1. 软件成本消耗与开发进度需求

在软件项目立项后，根据合同规定，对软件开发的进度和各步骤的费用提出要求，作为开发管理的依据。

1. 开发目标需求

预先估计以后系统可能达到的目标，这样可以比较容易对系统进行必要的补充和修改。

#### 需求分析的任务

需求分析的主要任务是借助于当前系统的逻辑模型导出目标系统的逻辑模型，其流程如下：

1. 确定对系统的综合需求（功能、性能、运行、扩充需求）
2. 制作产品需求文档 (PRD)
3. 分析系统的数据需求（概念模型、数据字典、规范化）
4. 导出目标系统的详细的逻辑模型（数据流图、数据字典、主要功能描述）
5. 开发原形系统
6. 从PRD提取编制软件需求规格说明书（SRS）

***备注：SRS格式***

*1.引言 2系统概述（项目背景、系统目标、核心业务流程） 3.术语说明 4.系统结构（架构图、功能图）*

*5.主体功能与业务逻辑（重点） 6.接口需求（内部、外部接口、） 7.网络总体设计（拓扑网络、主机、组网）*

*8.运行环境（Linux、Windows、IIS、 WebLogic、Tomcat、OLAP、OLTP、JDK 8.0 、.NET Framework 4.0等）*

## 面向对象程序设计（略）

#### 设计原则

1. SRP单一职责链

每个类都应该只负责做一件事。

1. OCP开封闭合原则

软件的实体（类、模块、函数等）应该是可以扩展的，但是不可修改的。

1. LSP替换原则

子类必须能替换他们的基类型。

1. DIP依赖倒置原则

高层模块不应该依赖于低层模块，二者都应该依赖于接口与抽象类。抽象不应该依赖于细节，细节应依赖于对象。

1. ISP接口隔离原则

不应该强迫客户依赖于并未使用的接口，而应该把胖接口分离。

#### 实现UML建模

1. 业务对象的提取
2. 根据SRS、CRC等实现用况建模
3. 实现业务顺序图
4. 建立类图，根据用况图建立对象之间的关联
5. 绘制活动图、实现协作图、状态图

## 开发管理

#### 建立项目计划

1. 设计总体架构

针对系统的实施需要，采取适当的且成熟的框架结构。

1. 控制可扩展度

扩展度过大，将提高系统的复杂程度，延长开发时间；扩展度过低，会直接影响系统的二次开发与维护。控制系统的可扩展性，能提高开发效率，降低系统维护的难度。

1. 建立基础设施

合理分配软、硬件等基础设施的部署所需要的时间与成本（例如：服务器的订购安装、光纤接入、软件平台订购）。

1. 划分开发任务

利用WBS（Work Breakdown Structure，工作分解结构）对可交付结果进行分类与划分。每个项目都能划分为多个不同阶段，每个阶段又可以分为多个工作包（Work Package），工作包是WBS里最小的可交付结果，最后从工作包中分解出多个开发任务列表。

1. 部署开发进度

一个项目应该按进度划分为多个开发阶段，每个阶段的开发周期一般在30~60个工作日以内。在此阶段内应该与客户举行协商会议，制定产品路线图，在开发过程中邀请客户积极参与并提出反馈意见。然后把该时段内的开发任务按照开发难度，依赖性，重要性等多方条件划分为多个迭代周期。

在Scrum 敏捷软件开发原则中，应该把每个迭代任务进一步细分为多个开发任务列表，开发任务的开发时间应该控制在15个工作小时以内，如果开发时间超出15个工作小时，应该考虑把开发任务再度细化。开发任务建议应该由组员自主选择，而不要使用强制分配的方式。

1. 测试项目成果

每个工作包都应该同步部署测试工作，提高项目的质量。对出错BUG的工作包应该由测试人员以文本方式记录，向开发人员展示错误所在，让开发人员及时进行修改。

#### 管理开发团队

1. 组建团队

按照工作任务与项目时间的前提条件建立团队，按团队职责分配人员，一般团队人数应该控制在8~12人之间。当团队人数超过15人时，应该考虑把团队分解成2个独立团队，负责不同的开发任务。

1. 分配开发任务

在每个迭代周期内（一般是15~30个工作日），应该把每个工作包进一步细分为多个开发任务，开发任务的开发时间应该控制在15个工作小时以内，如果开发任务的开发时间超出15个工作小时，应该考虑把任务再度细化。而开发任务应该以自由选择的方式分配给每个组员。

1. 监督开发进度

在迭代的前期举行一次会议，让组员了解开发的进展及流程，并以自主选择的方式分配开发任务。期间可使用Microsoft Project等工具记录开发流程的进展，在每个工作包完成开发后应该进行性功能的测试，并以文本方式记录测试结果。

每天举行一次15分钟的站立会议，让组员交待昨天已完成的开发任务，当天将要做的任务，与开发过程中所遇到的问题。并在每周末举行一次例行会议，交待总体进程。

在迭代末期举行一次冲刺会议，总结项目的进展，交行已完成的任务，回顾该迭代周期内所遇到的问题，为下一个迭代做好准备。

1. 系统测试

对每个已完成的工作包进行适时的测试，保证系统质量与性能。对测试结果进行文本的记录，并把测试结果与绩效工资收入挂钩，并以真实数据计算组员的绩效收入。

1. 解决开发中所遇到的问题

对开发人员进行前期培训，可适当按工作能力分配任务，指导组员的开发。当遇到问题时应该在当天的站立会议时即时提出，并在15个工作小时内解决所遇到的问题以防止问题进一步扩大。

#### 监管产品质量

1. 质量需要的是计划、设计而并非审查的。在产品建立的初级，必须与“质量保证”（QA）的部门进行协商，以正式文档的方式，决定恰当的质量策略和标准。
2. 在开发过程中使用TDD（测试驱动开发）的模式，提高开发质量。测试人员应该以文本方式记录bug，并与开发人员共同工作的，把突出的缺陷演示给开发人员，以提高修改的效率。
3. 在每个迭代的结束时进行一次产品效果的演示，从客户、使用者、高层领导中收集反馈信息。在团队内部举行评审会议，分析测试结果，了解产品性能，为下次迭代所需要做的改进做好计划。

#### 修改项目计划

1. 在产品需要识别阶段，应该以文档形式记录产品功能与开发流程，在开发计划需要修改时，应该与客户共同探讨，让客户了解计划修改对项目进度所造成的影响。
2. 项目计划的修改应该由统一的负责人提出,并且由用户需求的审核领导者认可。需求变更的提出应该是定期而不是随时的。
3. 计划的变更应该做好详细的文本记录，让客户了解需求变更的实际情况和开发方为之所付出的成本代价。

## 产品交付

#### 项目的后期审核

在项目开发最终完成后，对开发人员来说可算是放下工作的重担，但对项目经理来说这往往是项目的关键时刻。前期的风险评估、成本预算、需求分析、软件设计都是为了引导项目走向这一时刻，此时所有的目光都将投向项目管理人员。你可能发现大量而琐碎的工作将要在几个小时内完成，此刻项目经理更需要保持清醒与镇定，把最后的工作视为微型项目来对待。细致地对项目进行后期的审核，分析项目成果、项目团队的效率、可交付产品的价值，以此审核结果可作为项目管理经验总结的一部分。

#### 质量评审

在项目交付前，应该把项目交给相关的“质量保证”（QA）部门进行质量评审，并邀请典型用户感受产品的质量。

#### 项目的最终交付

正常情况下在项目的前期就会订立项目交付的协议，项目交付方式分为非正式验收与正式验收两种。一般在项目完成后都会先进行非正式验收，让客户体会项目的质量并提出反馈意见，最后在客户肯定产品质量后再以书面协议的形式进行正式的产品验收。

#### 项目的最终报告

在项目的最后，应该制定项目的最终报告，此报告可以视为是对该项目一个记录，但报告不必包含项目的所有方面。一般最终报告应该包含以下方面：

1. 最初引进项目时的初期项目视图
2. 对该项目的价值评估及支持性信息
3. 项目的范围
4. 项目的开发流程及WBS
5. 项目的会议记录
6. 项目变更的报告及变更的理由
7. 与项目相关的沟通过程文件
8. 项目的审核报告与客户验收报告
9. 项目成员的表现报告
10. 项目的最终成果