第一讲项目管理概述

1.1 人类有组织的活动逐步分化为两种类型:

作业：连续不断、周而复始的活动

项目：暂时性的、一次性的活动

项目是未完成某一独特产品或服务所做的一次性努力。

项目的特性：临时性独特性逐步完善性。

项目与作业的区别：

项目：独一无二、有限时间、革命性改变、状态的不平衡、目标之间不均衡、多变的资源需求、柔性的组织、效果性、风险和不确定性、以达到目标为宗旨；

作业：重复的、无限时间、渐进性的改变、平衡、均衡、稳定的资源要求、稳定的组织、效率性、经验性、已完成任务为宗旨。

项目的重要性—项目的价值，项目是实现价值、成就事业的载体。

项目与项目管理的价值

国家、企业和个人来说：项目是发展基本元素，项目是进步和成长的主要载体。

项目管理：将相关的知识、技术、工具、技能、等应用于项目任务，以满足项目干系人对项目的需求和期望的过程。

什么是项目管理?

PMI 对项目管理的定义为：把各种知识、技能、手段和技术应用于项目活动之中，以达到项目的要求。通过应用和综合诸如启动、规划、实施、监控和收尾等项目管理过程来进行。

PMRC 对项目管理的定义为：以项目为对象的系统管理方法。通过一个临时性的专门的柔性组织，对项目进行高效率的计划、组织、指导和控制，以实现项目全过程的动态管理和项目目标的综合协调与优化。

项目管理的 7 特点：管理对象是项目,或被当作项目来处理的运作；全过程贯穿着 系统工程的思想；组织具有特殊性；体制是一种基于团队管理的个人负责制；项目管理的方式是目标管理；要点是创造和保持一种使项目能顺利进行的环境；项目管理的方法和手段具有先进性、开放性。

项目的三重制约：时间、成本、范围

项目管理的价值：企业的成功在于有效地推行项目管理；项目管理是一种有效地知识积累方式，是进行知识管理的有效途径；越来越多的企业引入项目管理，把它作为主要的运作模式和提高企业运作效率的方案。

项目管理的生命周期（阶段）：启动、计划、实施、监控、收尾。

目前主要存在三大项目管理研究体系：

（1）美国项目管理协会—PMI 的 PMBOK

2004 的 PMBOK 把项目管理划分为 9 个知识领域和 44 个管理过程;

9 个知识领域包括:4 个核心知识领域 (范围 时间 成本 质量管理) 4 个辅助知识领域 (人力资源 沟通 风险 采购管理) 1 个项目综合管理

(2) 国际项目管理协会 (IPMA)的国际项目管理知识体系 (ICB):

(3) 中国的 iPMBOK: 将信息技术与信息化综合起来视为一体，用 IT 信息化来综合描述信息技术与信息化，与此对应的项目称为 IT 信息化项目。

人们还习惯于将以计算机为主体的各种项目称之为 IT 项目。

利用有限资源、在一定的时间内，完成满足一系列特定的 IT 信息化目标的多项相关工作叫做 **IT 项目**。

iPMBOK 的 IT 项目管理定义:

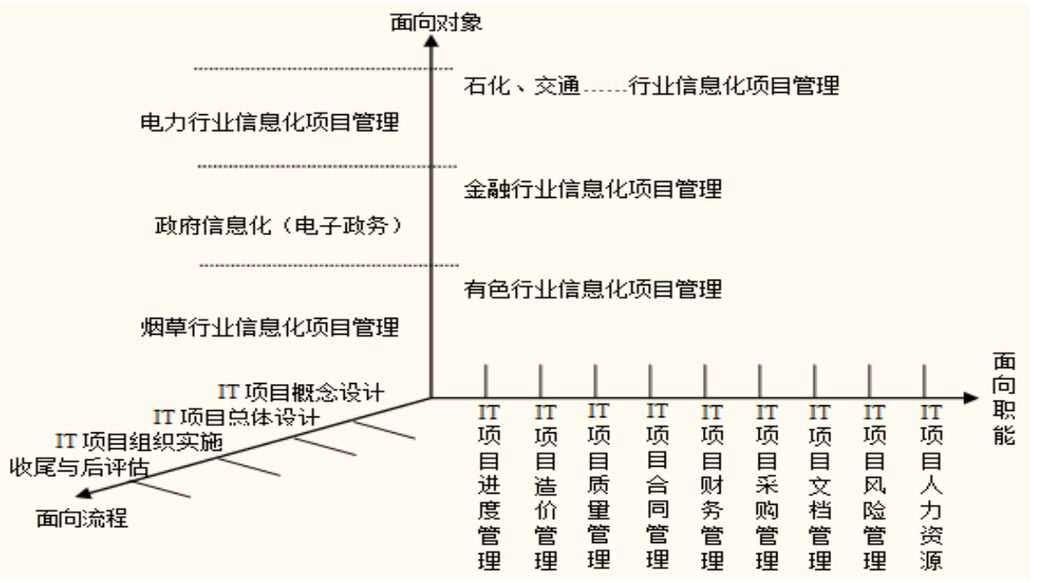
IT 项目管理就是把各种知识、技能、手段和技术应用于 IT 项目活动之中，以达到 IT 项目的要求；通过应用和综合诸如启动、规划、实施、监控和收尾等 IT 项目管理过程来进行的。

项目经理是负责实现 IT 项目目标的个人。

IT 项目特征:IT 项目除了具有独特性、一次性、整体性、临时性、不确定性、资源多变性、有一个主要发起人等特征外，还具有明显的 7 **特殊性**：目标的不确定性；需求的不稳定性；费用的不可控性；项目的时限性；对智力的依赖性；项目评价的主观性；项目的创新性

iPMBOK2004 采用三维结构模型，包括:面向项目管理职能的职能型 iPMBOK；面向项目管理过程的流程型 iPMBOK；面向项目管理对象的离散型 iPMBOK。

iPMBOK2004 采用三维结构模型



九大知识领域与项目管理阶段: 综合管理、范围管理、时间管理、费用管理、质量管理、人力资源管理、沟通管理、风险管理、采购管理。

软件以计算机为载体，软件的发展以计算机的发展为基础，同时，软件是计算机系统的核心和灵魂，软件的发展又有力的推进了计算机硬件的发展。

软件项目属于 IT 项目范畴，软件项目是以软件为产品的项目，其核心是软件。是指利用有限资源、在一定的时间内，完成满足一系列以软件为核心的多项相关工作。

软件的开发和应用通过软件项目来实现。

软件系统通常包括：在计算机运行中能够提供所希望的功能和性能的程序；使程序能够正确运行的数据结构和数据；描述系统结构的系统文档和如何使用与维护该系统的用户文档。

软件是计算机系统中与硬件相互依存的另一部分，它是程序、数据和文档的完整集合。

软件的 8 特点:软件产品的抽象性；软件生产过程的特殊性；软件缺陷检测的困难性；软件维护的复杂性；软件对环境的依赖性；软件开发方式与软件发展的不对称性；系统开销的主导性；与社会因素的关联性。

软件项目的成果是软件产品，与其他任产品的最大区别是无形和没有物理属性，其**特点**是：高度复杂性；智力密集、可见性差；单件生产、过程不规范；劳动密集、自动化程度低；开发工作渗透了人的因素；开发方法多样性。

软件项目管理的重点：人员的组织与管理；软件度量；软件项目计划；风险管理；软件质量保证；软件过程能力评估；软件配置管理。

启动阶段的任务:立项、组建项目组、方案设计与项目可行性报告、项目开工会。

输出:项目可行性报告：项目策划(需求问题、可选的技术解决方案、方案评价与选择、里程碑)、风险分析（项目风险水平定性分析表、子项目风险情况一览表)、项目质量保证说明书。项目章程(含项目成员表)

立项是项目前期工作的一个重要环节，可行性研究是它的重要工作内容。

可行性研究：对拟实行项目作详细的技术、经济分析，提出若干方案并进行比较，从而对项目的合理性作出结论。通过该过程储备一些可供选择的项目，使项目投资的基础工作超前；或取消一些不合理、不可行项目，避免或减少投资决策失误。

可行性研究的主要内容：旧系统描述（旧系统流程、费用开支、使用人员、主要问题和缺陷)；新系统概述（实现的功能、对旧系统的主要改进之处:、开发需要的条件)；经济可行性（投入：初始投资概要、运营费用概要。收益预测：直接经济收益、间接收益（效率提高、客户满意度、企业/组织形象提升等等)、经济可行性结论： 行 / 否)；技术可行性（新系统的技术要求、现有技术能否达到要求、技术可行性结论：行 / 否)；社会环境可行性（法律、政策、管理环境、文化环境（技术普及程度、用户的工作与生活习惯)、社会环境可行性结论：行 / 否)；可行性结论（立即开发、等条件成熟后再开发、不开发)

组建项目组：为项目建一个团队，配置合理、团结一心的团队是项目成功的保障。要求：合建一个合理项目组，有明确的分工，清晰的组织分解结构 OBS。寻找合适的人选，了解他们在技术、管理方面的优劣。争取领导和职能部分的支持。结果：一个项目成员表。

项目组的一般结构：赞助人、项目经理、核心团队、外围团队。

赞助人的职责：项目赞助人通常对项目提供资金和支持职责，包括，挑选和任命项目经理，批准项目核

心成员的组成；提供资金和审批重大财务事项；监控项目组执行情况；项目经理的求助对象。

项目经理的职责：项目经理直接对项目赞助人负责，保证项目成功的实施。包括，与赞助人协商，就项目目标和所需的资源达成共识；挑选核心成员，并取得他们的支持；在项目的进程中不断了解客户的需求；在项目计划过程中领导及指导小组成员；保证与项目干系人的沟通并汇报项目的进程；监控项目的进程，保证项目按时间计划执行。

项目成员的责任：项目核心成员对项目经理负责，保证项目的完成，包括，参与项目的计划制定；服从项目经理的指挥，执行计划分配的任务；配合其他小组成员工作；保持与项目经理沟通。

软件项目组的其它角色：产品经理，作为业主的代表，全程参与项目开发，负责确定软件功能需求，回答开发团队对需求的疑问；系统分析师；架构师；用户界面设计师；程序员；测试工程师；培训与实施工程师...

项目策划/任务书的基本要素：项目内容描述；项目里程碑项目评价标准；约束条件；项目利益干系人。结果：一个项目成员表

定义里程碑：划分项目阶段((一期，二期...)，一期，第一阶段...)；项目进展阶段的标志性成果。

项目评价标准

对项目组(承接单位)执行能力进行评价（是否通过 ISO9000 认证；是否通过信息产业部系统集成企业资质认证；无规范质量体系；有无规范管理方法；有无成功类似项目；获得信息产业部系统集成项目经理证书人员数)；对项目阶段性成果进行评价。

假设与约束条件：假设：说明项目的假设条件。约束条件：说明项目启动和实施过程中的限制性条件；项目的风险分析；成本因素。

项目的风险分析：整体各阶段的风险分析(分析、设计、实现、测试、运行)。风险因素：费用、工期、质量、组织、技术；自然、行为、经济、政治、组织；对因素进行加权计算。动态地进行项目风险评估和排解。

启动阶段关键点 1.与客户、SPONSOR、高层的沟通，明确需求及获得相关支持 2.明确项目目标和定位 3.开工会、统一思想、明确团队运作制度。

启动阶段常见问题：1.需求不明确及需求沟通不够 2.项目组成员选择不合理 3.为促成项目，过于乐观地分析项目可行性。

第三讲 Scrum 重点!

Scrum 特指一种敏捷开发的模型。简言之就是是一种以人为核心、迭代、循序渐进的开发方法。强调以人为本，专注于交付对客户有价值的软件。在高度协作的开环境中，使用迭代式的方式进行增量开发，经常使用反馈进行思考、反省和总结，不停的进行自我调整和完善。Scrum 的项目过程有一系列的 Sprint 组成。

非敏捷开发方式是瀑布式开发，**瀑布模型的主要缺陷**是：程序的维护成本会越来越高（需要很多人)；团

队氛围压抑（感受不到激情）；不方便做需求变更（引起客户不满）。

Scrum 框架：三个角色（产品负责人，scrum master，团队）四个仪式（sprint 计划会议，每日站会，sprint 评审会议，sprint 回顾会议）三个物件（产品 backlog，sprint backlog，燃尽图）。

Sprint （冲刺）计划会议：计划会议要有足够的时间，最好至少 8 个小时；取出部分产品需求做成 sprint 需求，并写成索引卡；确定并细分每一个索引卡的故事（Story）；进行工作认领（不是分配）；确定每日站立会议的时间和地点；确定好演示会议和回顾会议的日期

站立会议：10-15 分钟；迟到将接受惩罚；自问自答三个问题；昨天做了什么；今天要做什么；遇到了什么问题；更新燃尽图；Sprint 开发周期：；使用好任务看板；需求，设计，开发，测试，维护；注意燃尽图；不要使用软；取代看板；可以选择性的和 XP 的某些方式结合

演示会议：演示是跨团队的，会产生不同团队之间的交流；不要关注太多的细节，以主要的功能为主；让老板和客户看到；非常的重要，绝对不可以被忽略

回顾会议：时间在 1-3 个小时；找最舒适的地方（要有回顾看板）；开始的时候轮流发言，而不是主动发言；记录问题，总结，并讨论改进的方法，放在回顾看板上；每人三个磁铁，将最重要的 2-3 个改进点，成为下一轮的产品需求

Scrum 的主要缺陷：压力大；不方便跨时区，跨语言；程序维护成本偏高；无法被中断。

改善 Scrum：和客户坐在一起；结对编程；测试驱动开发（TDD）；使用编码规范；32 小时工作制。

许多企业面临的问题挑战：产品投放市场的时间太慢；项目失败的比例高的离谱；投资回报低，经常失败；对变化与变更的响应，难度大且成本高；客户体验及客户为导向很差；软件质量不过关；生产力需要大幅提高；员工士气，动力及责任感很低；需要普遍的微观管理；人员流失率特别高。

哪些类型的项目已经在使用Scrum：大型企业级软件项目；商业软件产品；消费者软件项目/大型网站；高可靠性系统；财务支付系统；智能家居项目；战斗机项目；大型数据库应用；手机项目；多地点同步开发；非软件项目。

敏捷原则：通过尽早和持续地交付有价值的软件来满足客户。欢迎对需求提出变更。要不断交付可用的软件，周期越短越好。业务人员与开发人员必须在一起工作。要善于激励项目人员。最有效的沟通方法是面对面的交谈。可用的软件是衡量进度的主要指标。敏捷过程提倡可持续的开发。对技术的精益求精。要做到简洁。团队要定期反省如何能够做到更有效。

产品负责人的职责如下：确定产品的功能。决定发布的日期和发布内容。为产品的ROI负责。根据市场价值确定功能优先级。每个Sprint，根据需要调整功能和优先级。接受或拒绝接受开发团队的工作成果。

ScrumMaster（项目经理）职责：保证团队资源完全可被利用并且全部是高产出的。保证各个角色及职责的良好协作。解决团队开发中的障碍。做为团队和外部的接口，屏蔽外界对团队成员的干扰。保证开发过程按计划进行。

Scrum团队：一般情况人数在5-9个左右；团队要跨职能。团队成员需要全职。在项目向导范围内有权利做任何事情已确保达到Sprint的目标。高度的自我组织能力。向Product Owner演示产品功能。团队成员构成在sprint内不允许变化。

Scrum物件之产品Backlog：一个需求的列表。一般情况使用用户故事来表示backlog条目。理想情况每个需求项都对产品的客户或用户有价值。Backlog条目按照商业价值排列优先级。优先级由产品负责人来排列。在每个Sprint结束的时候要更新优先级的排列。

燃尽图直观的反映了Sprint过程中，剩余的工作量情况，Y轴表示剩余的工作，X轴表示Sprint的时间。随着时间的消耗工作量逐渐减少，在开始的时候，由于估算上的误差或者遗漏工作量有可能呈上升态势。

扩展Scrum：一般情况一个团队的人数控制在5-9人。大型项目可以采用多团队，通过team of teams来扩展Scrum。影响扩展的因素：团队规模，项目类型，项目周期，团队分布。

第4讲　IT 项目计划阶段

计划阶段的任务：分解项目工作；设定活动排序；估算资源、工期、成本；作出风险计划、沟通计划；项目计划。

输出：WBS；网络图/甘特图；进度计划；风险计划；沟通计划。

计划阶段的**工具、方法、模板**：活动排序：网络图；；工期估算：三点估算法、专家判断法；成本估算法：自下而上法、专家判断法、类别估算法、参数成本法；进度计划：甘特图、里程碑图、关键路径法；模板：WBS，进度计划表，风险管理表，沟通计划表。

工作分解结构：一种面向可交付成果的项目元素分组，它自顶向下，定义了全部的项目工作范围。每下降一级，都表示一个更详细的项目工作定义。分解到可预测、可管理的单个活动，对工作项和工作结构进行验证和评审。

工作分解的原则：①百分百分解原则：完全穷尽，彼此独立，子层集合等于父层包括的所有工作。②分解到最底层：是最小的控制单元，包含所有具体工作(活动)，一个清晰的任务，一个清晰的责任人，能够估算工作量和工期，通常活动的长度应小于两周(80 小时)③清楚地描述项目的逻辑范围，并得到各方的认可。

工作分解的方法：①纵向分解：以交付产品物理/逻辑结构进行划分：物理产品，以产品结构为基础进行划分；软件产品，以外部功能项、结构组件等 为基础进行划分；中间输出或可交付的成果②横向分解：横跨产品所有内容的分解。如设计，测试、安装、培训、文档、风险管理等。它们通常是技术性，或支持性的③识别项目中的其它所有工作领域。

分解方法：自上而下法、头脑风暴法。**表达方式：**图形式、目录式

图形式分解举例：按项目的主要交付结果分、按职能分、按产品本身结构分、按项目实施顺序分。

需求验证四步骤：审查需求文档；依据需求文档编写测试用例；编写用户手册；确定产品验收合格的标准。

需求验证九内容：有效性检查；完备性检查；一致性检查；现实可行性；可检验性可跟踪性；可调节性检查；可读性。

排序方法：按工作的客观规律排序，将项目目标要求排序，按轻重缓急排序，根据项目本身的内在依赖关系排序。

技巧：只用工作分解结构(WBS)的最低层项，首先把最相关的项排好(建一个子网)，再合并所有子网，先不要考虑资源、日期或工期。

工具：前导图法(PDM)：用一个方块表示一个工作项，用箭头表示先后关系，连接各个方块，所有工作形成一个有向图。

资源、工期和成本估算：资源类型：人员，物资，技术；资源估算要素：各个项目成员需要什么资源，什么时候需要，需要多少？谁决定资源的分配？；估算方法：专家判断法。

工期估算：根据项目范围和资源的相关信息，确定(估算)完成所有活动所需要的工期。

估算方法：三点估算法：采用 a 乐观、c 悲观、b 最可能的三点工期估算法，计算平均值、标准差值确定工期的方法，工期=(a+4b+c)/6。专家判断法：由项目经理组织 1~3 名团队成员对任务消耗的工期进行估算，并确定项目日程。

任务的工期估算要以“谁来做”和“如何做”为基础。

成本估算信息来源：历史项目、任务执行者、专业评估人员、专业权威

估算方法：自下而上估算方法：估算最详细的计划活动费用，然后汇总到更高层级；专家判断法；类比估算法：利用历史信息和专家判断，只有当满足以下条件时比较可靠：(先前的项目与当前项目类似、做估算的个人或小组具有必要的经验)；参数成本估算法：如确定项目成本与计算机程序代码行数的关系。

进度计划：根据 WBS、活动排序、工期估算和所需要资源的结果进行分析、制定出项目进度计划。

进度制定的工具：（1）关键路径法：关键路径：工期总和最长的一条路径。是完成该项目的需要的最短时间，其中的每一项任务都是关键任务，关键任务的延迟会导致项目或阶段延迟（2）甘特图

风险计划：①识别风险。风险级别：（1）项目级风险：环境风险、金融风险、法律风险、项目方案总体规划风险（2）模块级风险：项目管理风险、工程分包风险、配套采购风险②评估风险：考虑风险发生的可能性，风险对项目的影响。

对风险的基本处理方法：规避:改变项目计划，以排除风险或改变条件，使项目不受其影响；转移：设法将风险的后果连同应对责任转移到第三方；减轻：设法把不利的风险事件的发生概率或后果降低到可以接受的临界值；接受：主动接受、被动接受。

沟通管理的要点四个适当：适当的时间，将适当的信息，通过适当的渠道，发送给适当的利益干系人，并确保利益干系人正确理解。

分析利益干系人与项目的兴趣及影响程度，针对每个利益干系人制定**沟通计划。**

沟通的三大原则：及时、准确、信息量恰到好处。

计划阶段的关键：明确项目范围、全面的风险识别、各关键干系人的识别与沟通计划

计划阶段常见问题：对工作任务分解不充分、风险防范意识不强，无沟通计划、计划通常由个人制定，没有在项目组达成共识。

第 5 讲　IT 项目实施与监控

实施阶段：建立与完善项目联络渠道、建立项目信息控制系统、实施项目激励机制、按计划执行 WBS 的各项工作、解决实施中的问题。

指导/监督/预测/控制阶段：质量、进度、成本、风险、需求的变更等。

实施控制阶段任务：沟通，项目监控，变更管理。**输出：**项目会议纪要，项目状态报告，项目变更管理表。

软件企业与传统工业企业不同，与现代企业的其他行业相比，最主要**特征**是：（1）企业最主要的“资产”是一批掌握技术、熟悉业务、懂得管理的“人”。（2）人力成本是软件企业的主要成本（3）知识和经验的积累是软件企业主要的财富积累。因此：（1）软件企业的人力资源管理，是企业最主要的管理内容。（2）软件项目组的管理过程，几乎全部是围绕“人”来进行的管理。

“**人件**”：“人件”是迪马可和李斯特 1987 年合写的《人件：富有成果的项目和团队》一书中的术语，原指的是与计算机互动的人的条件。以后大家逐渐理解为与计算机硬件、软件相对应的**人**。

“**湿件**”“湿件”一词源自美国鲁迪·卢克 1988 年出版题为“湿件”的科幻小说。　“湿件“是新经济增长理论中的一种知识类型。新经济增长理论认为，经济的投入要素分为两大类：”硬件”和”知识”。**硬件**：非知识，非人类要素，如物品、自然资源、能源和物质基础设施等。**知识**又分为“软件”和“湿件”两种类型，“软件”也称“思想”，是编码化的、储存在人脑之外（如书籍、磁盘、录音录象带等）的知识；“**湿件”**也称“技能”或“只可意会的知识”，是储存于人脑之中、无法与拥有它的人分离的知识，包括能力、才干、信念等等。

软件与湿件之间的差异，在于编码化程度的不同。软件表示的思想可以用言语、符号或其他表达手段来表述，湿件则无法形式化，总是处于只可意会不可言传的状态。

新经济增长理论揭示了知识的两种类型：思想和技能的生成、传播和使用的不同特征。

新思想的产生是很不容易的，但新思想一旦产生，却能很方便、廉价地传播，可以为任何数量的人使用。而**技能**则是由从人的内在素质到个人经验再到训练等种种因素组成，只有拥有它的人才能使用，它的传输复杂、代价高昂和缓慢。

知识积累和创新，是思想（软件）、技能（湿件）与机器设备、基础设施等物质生产要素（硬件）之间互动的结果，体现为复杂的持续性学习。它对经济组织的核心竞争力至关重要。

未来是湿的：　“我们可以把湿件理解为：是处于生命状态的东西，它和软件可以保存于无生命的代码状

态不同，和包括机器、设备在内的硬件更是不同。所以说，微软在软件的维度中存在，而开源运动在湿件的维度中存在。”更广泛的意义上说，前现代的组织，是按硬件的方式组织的；现代的组织，是按软件的方式组织的；后现代的组织是按湿件的方式组织的。所以，未来是湿的。

软件项目中人的特征：（1）高知识更新性：软件项目所需要的人的知识，是不断更新的知识。三年前熟悉的知识，可能三年后就基本没有什么价值。（2）高主观经验性：虽然软件的知识在不断更新，但是，开发经验、行业经验却是长期积累的。一个在行业中长期从事应用系统开发的熟练的系统分析师，是各软件企业抢手的热门人才。（3）高自主性：正是由于上述特点，高层次的软件人才，还是处于卖方市场。这使软件人才在人力资源市场的双向选择中，处于主动地位。（4）主观能动性：软件开发的特点，决定了软件人才个人行为，在开发过程中的影响和作用。工作绩效的好坏，工作效率的高低，在很大程度上，由项目中的个人所决定。（5）效率波动性：作为项目组中的个人，其效率的发挥，也是不稳定的。常常受各种因素的影响，呈现波动性。（6） 资源消耗性：项目中的个人，是项目资源的消耗者。进度、成本、质量控制和变化，首先是因为项目中人的因素的变化。（7）不可存储性：项目的人力资源，包括：人的时间、精力、知识、积极性等。

项目经理的职权（1）高层经理给项目经理的授权：任务分配权、预算制定与控制权、提升建议/批准权、资金使用权、处罚权。（2）项目经理的权力不是职务所固有的（3）项目经理主要通过个人的人格魅力、经验、影响力、协调能力来从事管理（4）影响力不可能只靠权力

项目经理的基本影响因素 1 权利——发布命令的正当的等级权利 2 任务——可感知到的对工作任务分配的影响力 3 预算——可感知到支配资金的能力 4 提升——提拔员工的权力 5 资金——涨工资和增加福利的权利 6 处罚——有实施处罚的权利 7 工作挑战——为特定员工分配所喜好的工作的权力 8 专门技术——拥有其他人很需要的知识、技术和技能 9 友谊——良好的人际关系。

两大沟通类型（1）项目组内沟通**。四种沟通需求：**职责分配，授权，协调，状态报告。会议，是最常用的沟通方式：项目开工会；成员进度汇报项目进展会沟通的要点：及时、公开、恰到好处。

（2）与高层、客户的沟通。需要回答以下问题：谁，为什么需要信息？他们需要什么类型的信息？何种详尽程度？频度如何？与高层沟通时，目标是什么？采用什么样的方法完成沟通？

沟通的要点：项目组全体成员对目标达成共识，项目沟通计划、规划，互相尊重，主动倾听，双赢。

有效沟通的关键要素：会前：事先了解为什么开会，以经预期要取得什么结果；考虑是否可能取消会议；确定需要参加会议的最少人数；选择会议地点，会议的布置与会议目标相协调**。会中：**做好准备，按时开始，并首先点明会议的目的和议程；每位与会者都有发言的机会；对会议内容进行口头总结。**会后：**会后发布会议纪要给每位与会者；必须产生明确的决定；所有决定必须立即付诸行动。

项目控制的基本内容：衡量当时情况；预测未来的情况；对比预测情况和当时情况；及时制定实现目标、进度或预算的修正方案。

软件项目监控的难点：软件项目的不确定性；项目内容的隐性与分散性；计划与变化的关系；培养按计

划工作的习惯；项目经理的权力有限。

软件项目监控的要点：明确项目控制的目的；加强来自各方面的综合、协调和督促；要建立项目管理信息制度（项目主管及时向领导汇报工作执行情况，定期向客户报告，并随时向各职能部门介绍整个项目的进程）

项目进度控制是一种循环的例行性活动。其分为**四个阶段**：编制计划，实施计划，检查与调整计划，分析与总结。

控制手段：制定并遵守计划、不断监督、必要时进行调整、沟通、团队工作

可视化图表工具：重要的关系、里程碑图、甘特图、费用成本曲线、资源负荷图、项目成本记录、工作绩效图、项目报告表。

作业控制的目的：采取一定措施，保证每一项作业本身按计划完成。

作业控制的内容：以工作分解结构 WBS 的具体目标为基础，针对具体的每个工作环节。检查每项作业的质量，

监控每项作业的进展情况。如果作业存在缺陷，则由项目管理部门下达指令，调整或重新安排存在缺陷的作业，以保证其不致影响整个项目工作的进行。

进度控制：采取措施保证项目按计划的时间表来完成工作，防止拖期。

影响工期的主要因素：责任心不强、信息失实或遗漏、协作部门的失误等。

按照不同管理层次对进度控制的要求分为三类：①高层次管理部门(项目总进度控制：对项目中各里程碑事件的进度控制)②多级项目的项目部门（项目主进度控制：对项目中每一主要事件的进度控制）③各作业部门（项目详细进度控制：对各具体作业进度计划的控制，这是进度控制的基础）

项目进展报告—主要内容 1. 项目进展简介：列出有关重要事项。对每一个事项，叙述近期的成绩、完成的里程碑以及其它一些对项目有重大影响的事件（如采购、人事、业主等）。2. 项目近期趋势：叙述从现在到下次报告期间将要发生的事件。对每个将要发生的事件进行简单说明。并提供一份下一期的里程碑图表。3.预算情况：一般以清晰、直观的图表反映项目近期的预算情况，并对重大的偏差做出解释。4.困难与危机:困难是指你力所不能及的事情，危机是指对项目造成重大险情的事,同时可提出高层管理人员支持的要求。5.人、事表扬

项目进展报告的形式 1.日常报告：日常报告是为报告有规律的信息，按里程碑时间安排报告时间，有时根据资源利用期限发出日常报告，也有时每周甚至每日提供报告；2.例外报告：此种报告的方式用在为项目管理决策提供信息报告；3.特别分析报告：常用于宣传项目特别研究成果或是对项目实施中发生一些问题进行特别评述。

在整个报告期内，需要**收集两种数据或信息**：1.实际执行的数据，包括活动开始或结束的实际时间；使用或投入的实际资源和成本等 2.有关项目范围、进度计划和预算变更的信息。

五种收集方法：1.发生概率统计法：即对某一事件发生的次数进行记录的信息收集方法，主要用于：延误报告次数、无事故天数、运行故障次数等。2.原始数据记录法：这是对项目中实际资源投入量和项目产出技术指标进行统计。3.经验法：这类指标的定量或定级来源于人的主观意志。4.指标法：对一些较难或者甚至无法直接获得的对象的有关信息，寻找一种间接的度量或指标。5.口头测定方式：这种方式常用于测定队员的合作质量、队员士气高低、项目主和业主间合作程度等。

第六讲 软件配置管理

为什么要进行软件配置管理？开发的系统越来越大，功能越来越复杂，众多软件开发人员，多种文件对象和类型，多种版本，多种平台，多个开发地点。

以往软件开发过程中的问题：版本难以控制；资源变化频繁导致失控；配置审核问题；项目开发中的组织管理问题

项目组成员之间沟通不够；文档与程序严重脱节；无法有效地管理和跟踪变更；测试工作不规范；对软件版本的发布缺乏有效的管理；施工周期过长,且开发人员必须亲临现场；无法构建企业内部的软件标准构件仓库。

造成的后果：重要数据丢失，开发周期漫长，开发成本增加，产品可靠性差，软件重用率低下，无法开展规范化测试工作，缺乏软件开发历史数据积累，无法为日后借鉴，维护和升级困难，施工成本增加，用户抱怨使用不便，满意度低，项目风险增加

实施软件配置管理的益处：加强了开发过程控制，加强了产品质量的可控性；节约费用：缩短开发周期、减少施工费用；有利于知识库的建立：代码对象库、业务及经验库；规范管理：量化工作量考核、规范测试、加强协调与沟通。

什么是软件配置管理？配置管理能够系统地处理变更，从而使得软件系统可以随时保持其完整性。配置管理又可称为 “变更控制”，可以用来评估提出的变更请求，跟踪变更，并保存系统在不同时间的状态。

另外一个定义：对软件开发组所建立的软件的修改进行标识、组织和控制的艺术，其目标是减少错误，提高生产力。

另外一个定义：标识和确定系统中配置项的过程，在系统整个生存周期内控制这些项的发布和变更，记录并报告配置的状态和变更要求，验证配置项的完整性和正确性。

软件配置管理作用：在质量体系的诸多支持活动中,配置管理处在支持活动的中心位置,它有机地把其它支持活动结合起来,形成一个整体,相互促进,相互影响,有力地保证了质量体系的实施。

目的：软件配置管理的目的是建立和维护在项目的整个软件生存周期中软件项目产品的完整性。

主要内容包括：及时地确定软件的配置，系统地控制软件配置的变更,保证整个软件生命周期软件配置的完整性和可追溯性。

存储库存储库是一个中央数据库，存放各种版本文件、目录、交付对象、源数据及其关联对象；位于网

络中的 SCM 服务器上，并能够通过所有客户端进行访问；用于存储全部的控制版本的组件、元素、分支和元数据。

软件开发库：指在软件生命周期的某一个阶段期间，存放与该阶段软件开发工作有关的计算机可读信息和人工可读信息的库

软件受控库：指在软件生命周期的某一个阶段结束时，存放作为阶段产品而释放的、与软件开发工作有关的计算机可读信息和人工可读信息的库。

软件配置管理就是对软件受控库中的各软件项进行管理，因此软件受控库也叫配置管理库。软件受控库通常用于保存基线

团队开发 ：版本控制：记录软件在生命周期内的所有变更情况，记录任何软件变更的历史，保存任何生效变更的版本，在版本控制下能任意回到历史过程中的任何一个版本

配置项：软件配置管理的对象是软件配置项。软件配置是指一个软件产品在软件生命周期各个阶段所产生的各种形式（机器可读或人工可读）和各种版本的文档、程序及其数据的集合。该集合中的每一个元素称为该软件配置中的一个配置项。与合同、过程、计划和产品有关的文档及数据，源代码、目标代码和可执行代码，相关产品，包括软件工具、库内可复用软件、外购软件及顾客提供的软件等

版本亦称配置标识，是指某一特定对象的具体实例的潜在存在。这里的某一特定对象是指由版本维护工具管理的软件组成单元，如：源代码；具体实例则是指软件开发人员从软件库中恢复出来的某软件组成单元具有一定内容和属性的一个真实拷贝。版本记录了配置项的演化过程。

版本标识：版本标识由版本的命名规则决定（1）数字顺序形版本标识。如：V1.0、 V1.1（2）符号命名版本标识如：X86/WinNT/（3）属性版本标识（4）正交版本标识。

基线：一个配置项在其生命周期的某一特定时间被正式标明、固定并经正式批准的一个版本，无论媒体是什么。基线形成一般是在软件生命周期各阶段末尾的特定点，亦称里程碑时间点。

作用：基线的作用是把各阶段的工作划分得更加明确，使得本来连续的工作在这些点上断开，使之便于检验和确认阶段开发成果，使后续工作在确认后的基准上进行。

特性：具有明确标识、具有明确内容、经正式审批、严格控制变更

如何进行配置？ （1）管理所有目录和全部文件的版本只是其中的一部分（2）因为软件产品与源代码是一对多的关系，SCM 需要好的配置报告或工作空间来进行管理。（3）记录并维护历史是必要的，但还远远不够！（4）一个 SCM 系统必须能够再生和重现一个软件产品的全部的完整的配置情况，而不仅仅是单个文件的版本（5）把配置管理想象成一个软件的时光机器。

工业化时期：并行开发管理

（1）大型软件项目：开发人员不断地创建新的版本、文件、目录；不断地修改已经存在的文件版本；需要从不同地方同时访问相同的代码；经常需要与其他开发人员的工作成果进行集成（2）在工业化时期我们称

之为“并行开发” （3）没有并行开发管理将会导致开发过程混乱（4）并行开发管理能力决定了一个团队的配置管理水平

软件开发是一个团队运动（1）复杂的开发需要彼此隔绝（2）由于不同原因需要同时修改相同的工作产品（3）同一个代码或产品需要支持多种方式：支持多种平台、支持多种版本、支持多项目、支持多地点

并行开发管理（1）允许多个开发人员同时工作在相同的代码，而无需等待他人完成后再开始（2）允许在相同代码上建立多个开发分支：支持多项目、支持多版本、支持多平台（3）能够使多个团队并行工作，甚至在远程（4）当变更完成并被他人认可时，可以很容易地将他们集成到一起

分支管理：并行开发的关键方法：隔离变更；对变更、BUILD、测试以及基线等进行隔离，分别进行管理；将工作分解、变更任务、工作分类等有机地组织起来；对变更任务的集成进行控制；有助于问题的交流、可见性、项目计划和跟踪以及最终的风险管理。

在集成和测试期间的 Build：隔离：将集成分支上的所有用户锁定；稳定：由专人进行 BUILD，不成功需要修改错误；对 BUILD 结果进行测试，有问题进行修复；共享：生成基线、提升老基线。

第七讲 版本控制软件介绍

使用版本控制软件的理由：及时了解团队中其他成员的进度；轻松比较不同版本间的细微差别；记录每个文件成长的每步细节，利于成果的复用(reuse);资料共享，避免以往靠拷贝文件造成的版本混乱；人人为我，我为人人。所有成员维护的实际是同一个版本库，无需专人维护所有文件的最新版本；协同工作，大大提高团队工作效率。

常用的开源版本软件软件：CVS，较早很流行的开源版本系统；SVN (Subversion)，开源，与 CVS 原理和功能相似，取代 CVS；Git，开源，分布式版本管理系统，目前发展势头很强。

Subversion 相关软件：是一个 **C/S** 系统软件，包含服务器软件部分 Subversion，客户端软件 TortoiseSVN 的版本控制系统。Subversion：是一个开源的版本控制系统，拥有 CVS 的大部分特征，并在 CVS 的基础上有更强的扩展，用来代替 CVS 系统。**TortoiseSVN**：SVN 的客户端工具，和资源管理器完美集成，基于 TortoiseCVS 的代码开发，使用上与 TortioseCVS 极其相似。

配置库：就是位于服务器端，统一管理和储存数据的地方。SVN 的核心是配置库，配置库按照文件树形式储存数据 - 包括文件和目录，任意数量的客户端可以连接到配置库，读写这些文件。通过写数据，别人可以看到这些信息；通过读数据，可以看到别人的修改。 最特别的是 Subversion 会记录配置库中的每一次更改，不仅针对文件也包括目录本身，包括增加、删除和重新组织文件和目录。

工作副本（WorkSpace,工作空间）与位于中央配置库相对应，每个人的工作空间，它是每个程序员工作的地方

程序员从配置库拿到源代码，放在本地作为工作副本。在工作副本上进行查看、修改、编译、运行、测试等操作，并把新版本的代码从这里提交回配置库库中。

SVN 的工作模式①复制-修改-合并方案(Subversion 默认的模式)。在这种模型里，每一个客户读取项目配置库建立一个私有*工作副本*——版本库中文件和目录的本地映射。用户并行工作，修改各自的工作副本，最终，各个私有的复制合并在一起，成为最终的版本，这种系统通常可以辅助合并操作，但是最终要靠人工去确定正误。②锁定-修改-解锁方案。在这样的模型里，在一个时间段里配置库的一个文件只允许被一个人修改。 此模式不适合软件开发这种工作。

SVN 服务器,支持 linux 和 windows，更多是安装在 Linux 下。

SVN 服务器有 2 种**运行方式**：独立服务器、借助 apache。2 种方式各有利弊。

SVN **存储版本数据**也有 2 种方式：BDB，在 Berkeley DB 数据库中存放数据；和 FSFS。是使用普通文件，采用自定义的格式来储存。因为 BDB 方式在服务器中断时，有可能锁住数据，所以还是 FSFS 方式更安全一点。

SVN 服务器软件：Windows 环境下常用的两种 Subversion 和 VisualSVN Server。VisualSVN Server 集成了 Subversion 和 Apache；封装 SVN Server 为 windws service，并随机器自动启动；图像化配置 Apache 服务器，指定认证方式、访问端口等简单操作；图像界面配置用户权限等。

Trunk、Tranches、Tags 之间的区别：Trunk、Tranches、Tags 只是便于用户区分不同类型版本的三个存放目录，为可选项。Trunk 为主流版本容器，用于存放该项目的主版本，通常一个项目对应一个主版本。Branches 为分支版本容器，用于存放该项目的分支版本。在项目开发过程中，经常要开发许多新功能，在开发这些新功能时，常会带来编译错误与 bug，影响原有程序的正常执行。因此，SVN 为开发这些新功能专门设有 Branches 存放目录，在新功能足够稳定的情况下，把这些版本融合到主流版本中。Tags 为特殊版本容器，每阶段保存一个版本，作为该软件的里程碑，比如发行版，方便你日后对各发行版 bug 的修改。

Subversion 的客户端有三种：第一种是 websvn 等基于 web 的，需要 web 服务器的支持，适宜于跨越防火墙、基于 Internet 的协作项目。开源项目常用它。第二种客户端软件，需要用户在本地安装客户端。以 TortoiseSVN 为代表，适宜集团内部的软件开发项目。第三种是插件，与 Netbean，Eclipse 等开发工具紧密集成

第八讲　软件项目风险管理

软件项目中的风险：不断变换的需求；低劣的计划和估算；不可信赖的承包人；欠缺的管理经验；人员问题；技术失败；政策的变化。

风险的三要素：一个未来的事件、事件发生的概率、事件的影响

风险分类：从风险的**范围**角度上看，可将软件项目的风险分为三种类型：项目风险：潜在的项目预算、进度、人员、资源、用户和需求等方面的问题。技术风险：实现和交付产品过程中所应用的各种技术所包含的风险。技术的正确性、不确定性、复杂性、技术陈旧等因素都可带来技术风险。商业风险：与市场、企业产品策略等因素有关的风险。从风险**可预测的程度**来看，可将风险分为以下三种类型：已知风险：通过评估

项目计划、项目的商业和技术环境以及其它可靠的信息来源之后可以发现的那些风险。可预测风险：能够从过去的项目经验中推测出的风险。不可预测风险：事先很难识别出来的风险。

风险管理是指在项目进行过程中不断对风险进行识别、评估和监控的过程，其目的是减小风险对项目的不利影响。

风险管理策略可以分为 4 个层次：危机管理：风险已经造成麻烦后才着手处理。风险缓解：事先制定好风险发生后的补救措施，但不制定任何防范措施。着力预防：将风险防范作为项目的一部分加以规划和执行。消灭根源：识别和消灭可能产生风险的根源。

风险管理计划的主要内容：风险应对计划（top 10 清单）；岗位职责；时间；预算；追踪等。

采取**主动的风险管理策略：**着力预防和消灭根源的管理策略。这一策略在项目计划阶段就启动了：识别出潜在的风险；评估它们出现的概率和潜在的影响；按照重要性进行排序；然后制定一个计划来管理风险。

软件项目**风险管理包括四个过程：**风险识别；风险评估；风险规划；风险监控

风险识别方法：①检查表法。风险检查表中列出了项目中常见的风险。项目相关人员通过核对风险检查表，判断哪些风险会出现在项目中。可根据项目经验对风险检查表进行修订和补充。该方法可以使管理者集中识别常见类型的风险。**风险条目罗列依据：**项目规模；商业影响；项目范围；客户特性；过程定义；技术要求；开发环境；人员数目及其经验。优缺点：快速而简单，可以用来对照项目的实际情况，逐项排查，从而帮助识别风险。由于每个项目都有其特殊性，检查表法很难做到全面周到。②**德尔菲方法。**德尔菲方法又称专家调查法，本质上是一种匿名反馈的函询法。把需要做风险识别的软件项目的情况：分别匿名征求若干专家的意见；然后把这些意见进行综合、归纳和统计；再反馈给各位专家，再次征求意见；这样反复经过四至五轮，逐步使专家意见趋向一致，作为最后预测和识别风险的依据。③**头脑风暴法。**头脑风暴法简单来说就是团队的全体成员自由地提出自己的主张和想法，它是解决问题时常用的一种方法。将项目主要参与人员代表召集到一起，各自根据自己对项目不同部分的认识，识别项目可能出现的问题。一个有益的做法是询问不同人员所担心的内容。头脑风暴法的优点是可对项目风险进行全面的识别。④**情景分析法。**情景分析法是根据项目发展趋势的多样性，对系统内外相关问题的系统分析，设计出多种可能的未来前景，然后用类似于撰写电影剧本的手法，对系统发展态势做出自始至终的情景和画面的描述。适应场合：适用于对可变因素较多的项目进行风险预测和识别的技术。特色：它在假定关键影响因素有可能发生的基础上，构造多重情景，提出多种未来的可能结果，以便采取适当措施防患于未然。

风险评估：对风险发生概率的估计和评价，项目风险后果严重程度的估计和评价，项目风险影响范围的分析和评价，以及对于项目风险发生时间的估计和评价。

风险值（风险的严重程度）R=F(P,I)P 是风险发生的概率。I 是风险发生后对项目目标的影响。

确定风险的优先次序：根据风险的严重程度进行排序，确定最需关注的前几个风险。风险评估的**方法**包括定性风险评估和定量风险评估。

定性风险评估：定性评估风险的发生概率及后果。风险概率度量：高、中、低；极高、高、中、低、极低；不可能，不一定，可能和极可能。

风险**后果度量：**高、中、低；极高、高、中、低、极低；灾难，严重，一般，轻微，可忽略等。

定量风险评估：量化分析每一个风险的概率及其对项目造成的后果，也分析项目总体风险的程度。分析方法：盈亏平衡分析；模拟；专家访谈；决策树分析；量化风险检查表……

专家访谈：邀请具有类似项目经验或相关领域经验的专家，这些专家运用他们丰富的经验和知识对软件项目的风险进行度量。其结果会相当准确和可靠，甚至有时比通过数学计算和模拟仿真得出的结果还要准确和可靠。

如果风险的影响后果大小**不易直接估算出来**，可以把后果分解为更小的部分，再对其进行评估，然后把各个部分的结果累加，得到总的评估值。例如，如果使用 3 种新编程工具，可以单独评估每种工具未达到预期效果的损失，然后把损失加到一起，这要比总体评估容易多了。

决策树分析：决策树是一种形象化的图表分析方法，把项目所有可供选择的方案、方案之间的关系、方案的后果及发生的概率用树状的图形表示出来，为决策者提供选择最佳方案的依据。决策树中的每一个分支代表一个决策或者一个偶然的事件，从出发点开始不断产生分支以表示所分析问题的各种发展的可能性。每一个分支都采用预期损益值（Expected Monetary Value, EMV）作为其度量指标。决策者可根据各分支的预期损益值中最大者（如求最小，则为最小者）作为选择的依据。预期损益值等于损益值与事件发生的概率的乘积，即：EMV=损益值×发生概率

风险规划：针对风险分析的结果，制定风险应对策略和措施的过程，其目标是应对、减少、以至于消灭风险事件。

风险规划的**主要策略：**回避风险：是对可能发生的风险尽可能地规避，采取主动放弃或者拒绝使用导致风险的方案。消除了风险的起因，将风险发生概率降为零。具有简单和彻底的优点。注意事项：对风险要有足够的认识；当其他风险策略不理想的时候，可以考虑；可能产生另外的风险；不是所有的情况都适用，有些风险无法回避，如用户需求变更；转移风险。转移风险是为了避免承担风险损失，有意识地将损失或与损失有关的财务后果转嫁出去的方法。缓解风险。在风险发生之前采取一些措施降低风险发生的可能性或减少风险可能造成的损失。接受风险。项目团队有意识地选择由自己来承担风险后果。当风险很难避免，或采取其它风险应对方案的成本超过风险发生后所造成的损失时，可采取接受风险的策略。主动接受：在风险识别、分析阶段已对风险有了充分准备。当风险发生时马上执行应急计划。被动接受：风险发生时再去应对。在风险事件造成的损失数额不大，不对软件项目的整体目标造成较大影响时，项目团队将风险的损失当做软件项目的一种成本来对待。

风险监控：实施和跟踪风险管理计划；确保风险策略正在合理使用；监视剩余的风险和识别新的风险；收集可用于将来的风险分析信息。

项目风险监控体系的建立，包括制定项目风险的方针、程序、责任制度、报告制度、预警制度、沟通程序等方式，以此来控制项目的风险。

项目风险审核：项目风险审核风险计划是否有效地实施并达到预定目标。确定监控活动是否符合项目风险计划，确定监控结果是否符合项目风险计划，系统地进行项目风险审核是开展项目风险监控的有效手段，可以作为改进项目风险监控活动的一种有效的机制。

挣值分析：通过挣值分析可以显示项目在成本和进度上的偏差。如果偏差较大，则需要进一步对项目风险进行识别、分析。挣值分析的三个基本参数包括：计划值（PV）、实际成本（AC）和挣值（EV）： 1、计划值，又叫计划工作量的预算费用。是指项目实施过程中某阶段计划要求完成的工作量所需的预算工时（或费用）。计算公式是： PV=BCWS=计划工作量*预算定额 。PV 主要反映进度计划应当完成的工作量，而不是反映应消耗的工时或费用。2、实际成本，又叫已完成工作量的实际费用。指项目实施过程中某阶段实际完成的工作量所消耗的工时（或费用）。主要反映项目执行的实际消耗指标。 3、挣值，又叫已完成工作量的预算成本。指项目实施过程中某阶段实际完成工作量及按预算定额计算出来的工时（或费用）。计算公式是：EV=BCWP=已完成工作量*预算定额。

风险评价：软件项目管理会面临很多已知和未知的问题，尤其是没有管理经验的项目经理更应该及早评价和预防项目风险。

风险评价分类：按照阶段不同可以分为：事前评价；事中评价；事后评价；跟踪评价等。按照评价方法不同可以分为定性评价；定量评价；综合评价等。

推荐的风险管理措施：软件项目计划应包括风险管理计划；在必要时任命风险管理负责人；使用 TOP TEN 风险清单作为主要的风险管理工具；建立匿名风险汇报渠道。

第 10 讲 软件质量管理--全面软件质量管理

重要的理念：**商业目标决定质量目标。提高软件质量的最终目的是为了赢利，而不是创造完美无缺的产品。因此对于普通商业软件而言，并不是“质量越高越好”，而是恰好让广大用户满意，并且将提高质量所付出的代价控制在预算之内。**

CMM 对**质量的定义是**：① 一个系统、组件或过程符合特定需求的程度；② 一个系统、组件或过程符合客户或用户的要求或期望的程度。 我们这样理解软件质量： 软件质量是许多质量属性的综合体现，各种质量属性反映了软件质量的方方面面。人们通过改善软件的各种质量属性，从而提高软件的整体质量。

软件的质量属性很多，如正确性、精确性，健壮性、可靠性、容错性、性能、易用性、安全性、可扩展性、可复用性、兼容性、可移植性、可测试性、可维护性、灵活性等。

十大软件质量因素：功能性质量因素：正确性，健壮性，可靠性。非功能性质量因素：性能，易用性，清晰性，安全性，可扩展性，兼容性，可移植性。

软件质量要素（1）从技术角度讲，对软件整体质量影响最大的那些质量属性才是质量要素（2）从商业

角度讲，客户最关心的、能成为卖点的质量属性才是质量要素。

正确性。正确性是指软件按照需求正确执行任务的能力。 正确性无疑是第一重要的软件质量属性。技术评审和测试的第一关都是检查工作成果的正确性。

健壮性。健壮性是指在异常情况下，软件能够正常运行的能力。 正确性描述软件在需求范围之内的行为，而健壮性描述软件在需求范围之外的行为。健壮性有两层含义：一是容错能力，二是恢复能力。

可靠性。可靠性是指在一定的环境下，在给定的时间内，系统不发生故障的概率。可靠性本来是硬件领域的术语。

软件可靠性问题主要是在编程时候埋下的祸害（很难测试出来），应当提倡规范化程序设计，预防可靠性祸害。

性能。性能通常是指软件的“时间-空间”效率，而不仅是指软件的运行速度。人们总希望软件的运行速度高些，并且占用资源少些。 既要马儿跑得快，又要马儿吃的少。 性能优化的关键工作是找出限制性能的“瓶颈”，不要在无关痛痒的地方瞎忙乎。程序员可以通过优化数据结构、算法和代码来提高软件的性能。

易用性。易用性是指用户使用软件的容易程度。导致软件易用性差的根本原因 ：理工科大学教育存在缺陷：没有开设人机工程学、美学、心理学这些必修课，大部分开发人员不知道如何设计易用的软件产品。开发人员犯了“错位”的毛病：他以为只要自己用起来方便，用户也就会满意。 软件的易用性要让用户来评价。当用户真的感到软件很好用时，一股温暖的感觉油然而生，于是就用“界面友好”、“方便易用”等词来评价软件产品。

清晰性。清晰意味者所有的工作成果易读、易理解，可以提高团队开发效率，降低维护代价。 开发人员只有在自己思路清晰的时候才可能写出让别人易读、易理解的程序和文档。

安全性。这里安全性是指信息安全。安全性是指防止系统被非法入侵的能力，既属于技术问题又属于管理问题。

可扩展性。可扩展性反映软件适应“变化”的能力。

兼容性。兼容性是指不同产品（或者新老产品）相互交换信息的能力。

可移植性。软件的可移植性指的是软件不经修改或稍加修改就可以运行于不同软硬件环境（CPU、OS和编译器）的能力，主要体现为代码的可移植性。

为了提高用户对产品的满意度，企业必须提高产品的质量。但是企业不可能为了追求完美的质量而不惜一切代价，当企业为提高质量所付出的代价超过销售收益时，这个产品已经没有商业价值了，还不如不开发。企业必须权衡质量、效率和成本，产品质量太低了或者太高了，都不利于企业获取利润。企业理想的质量目标不是“零缺陷”，而是恰好让广大用户满意，并且将提高质量所付出的代价控制在预算之内。

消除软件缺陷的方式：

提高软件质量最好的办法是：在开发过程中有效地防止工作成果产生缺陷，将高质量内建于开发过程之

中。主要措施是“不断地提高技术水平，不断地提高规范化水平”，其实就是练内功，通称为“软件过程改进”。在开发软件的时候，即使人们的技术水平很高，并且严格遵守规范，但是人非机器，总是会犯错误的，因此无法完全避免软件中的缺陷。当工作成果刚刚产生时马上进行质量检查，及时找出并消除工作成果中的缺陷。最常用的方法是技术评审、软件测试和过程检查，已经被企业广泛采用并取得了成效。

质量人员的主要职责：（1）负责制定质量计划；（2）负责过程检查，约占个人工作量的 20%；（3）参与技术评审，约占个人工作量的 30%；（4）参与软件测试，约占个人工作量的 30%；（5）参与软件过程改进，约占个人工作量的 20%。

质量管理计划的主要内容：1. 质量要素分析 2. 质量目标 3. 人员与职责 4. 过程检查计划 5. 技术评审计划 6. 软件测试计划 7. 缺陷跟踪工具 8. 审批意见。

技术评审的目的是尽早地发现工作成果中的缺陷，并帮助开发人员及时消除缺陷，从而有效地提高产品的质量。

技术评审的主要好处：通过消除工作成果的缺陷而提高产品的质量；技术评审可以在任何开发阶段执行，不必等到软件可以运行之际，越早消除缺陷就越能降低开发成本；开发人员能够及时地得到同行专家的帮助和指导，无疑会加深对工作成果的理解，更好地预防缺陷，一定程度上提高了开发生产率。

技术评审两种基本类型：正式技术评审（FTR）。FTR 比较严格，需要举行评审会议，参加评审会议的人员比较多。

非正式技术评审（ITR）。ITR 的形式比较灵活，通常在同伴之间开展，不必举行评审会议，评审人员比较少。

技术评审和软件测试的目的都是为了消除软件的缺陷，两者的**主要区别**是：前者无需运行软件，评审人员和作者把工作成果摆放在桌面上讨论；后者一定要运行软件来查找缺陷。技术评审在软件测试之前执行，尤其是在需求开发和系统设计阶段。相比而言，软件测试的工作量通常比技术评审的大，发现的缺陷也更多。在制定质量计划的时候，已经确定了本项目的主要测试活动、时间和负责人，之后再考虑软件测试的详细计划和测试用例。如果机构没有专职的软件测试人员的话，那么开发人员可以兼职做测试工作。当项目开发到后期，过程检查和技术评审都已经没有多少意义了，开发小组急需有人帮助他们测试软件，如果质量人员参与软件测试，对开发小组而言简直就是“雪中送炭”。

过程检查的要点是：找出明显不符合规范的工作过程和工作成果，及时指导开发人员纠正问题，切勿吹毛求疵或者在无关痛痒的地方查来查去。

流程：质量人员在执行过程检查的时候，如果发现问题，应该立即记录下来。过程问题也是缺陷，因此最好使用缺陷跟踪工具，有助于提高过程检查的效率。 质量人员首先设法在项目内部解决已经发现的质量问题，与项目成员们协商，给出解决措施。在项目内难以解决的质量问题，由上级领导给出解决措施。

缺陷的主要属性：缺陷 ID，缺陷类型,缺陷状态,缺陷描述,相关文件,严重性,优先级,报告者,报告日期,接受

者,解决方案（建议）,解决日期 。

缺陷跟踪工具的常见功能：查询缺陷,添加缺陷,修改缺陷,删除,缺陷分类图,缺陷趋势图, Email。

第 13 讲 软件项目的质量管理

现代软件工程的特点是： 从开发过程（需求、设计、编码、测试、维护）到产品过程、项目过程、再过程； 从传统意义的软件开发及管理，到软件合同、运作、管理，包括:三个方面的过程：基本过程、支持过程、组织过程。7 大活动的软件过程工程：采购、开发、维护、运作、获取、管理、支持。

质量的定义：反映实体满足明确和隐含需要能力的特性综合。明确需要：指合同中用户明确提出的要求与需要。隐含需要：指由生产企业通过市场调研进行识别与探明的要求或需要。

质量与等级的关系：等级的含义是：对功能用途相同、但技术特性不同的存在事务的一种分类或排序。确定质量和等级标准水平，是项目经理的责任。

软件质量的因素：概括为三个方面：产品修改、产品转移、产品运行。具体包括：正确性、健壮性、效率、完整（安全）性、可用性、风险、可理解性、可维修性、灵活（适应）性、可测试性、可移植性、可再用性、互运行性。

质量的要素：讨论软件的质量定义，从 4 个角度来看，即：用户的角度；开发商的角度；产品的角度；价值的角度。

1985 年，国际标准化组织（ISO）建议，软件质量**度量模型由三层组成**。高层称软件质量需求评价准则（SQRC），中层称软件质量设计评价准则（SQDC），低层称软件质量度量评价准则（ SQMC ）。

软件质量评价准则：可审查性，准确性，通信通用性，完全性，简明性，一致性，数据通用性，容错性，执行效率，可扩充性，通用性，硬件独立性，检测性，模块化，可操作性，安全性，自文档化，简单性，软件系统独立性，可追踪性，易培训性

软件质量的度量——四层模型： 质量需求层+质量特性+子质量特性+度量。

在四层模型的第一层，软件产品质量层，是产品必须满足的质量需求。它是用用户术语描述的，主要有四点：（1）产品将在用户所在组织当前使用的平台和操作系统上运行。（2） 产品将是可靠的并能防止数据丢失的机制。（3） 产品将提供完成某些任务所必需的功能。（4） 产品将易于使用。第二层：质量特性。在模型的第二层，表示与整个质量需求有关的特殊质量特性，它代表了用户的质量需求。它采用从用户角度考虑的立场，把软件质量分解成四类质量特性，这四个质量特性是软件的基本特征。

IEEE 的**四个质量特性**是：可移植性、可靠性、功能性、可使用性。（1）功能性：软件所实现的功能满足用户需求的程度．功能性反映了所开发的软件满足用户指明的或隐含的需求的程度，即用户要求的功能是否全部实现了。（2）可靠性：在规定的时间和条件下，软件所能维持其性能水平的程度。可靠性对某些软件是重要的质量要求，它除了反映软件满足用户需求正常运行的程度，且反映了在故障发生时能继续运行的程度。（3）易使用性：对于一个软件，用户学习、操作、准备输入和理解输出时，所做努力的程度。易使用性

反映了与用户的友善性，即用户在使用本软件时是否方便。（4）系统效率：在指定的条件下，用软件实现某种功能所需的计算机资源（包括时间）的有效程度。效率反映了在完成功能要求时，有没有浪费资源，此外“资源”这个术语有比较广泛的含义，它包括了内存、外存的使用，通道能力及处理时间的占用等等。（5）可维护性：在一个可运行软件中，为了满足用户需求、环境改变或软件错误发生时，进行相应修改所做的努力程度。可维护性反映了在用户需求改变或软件环境发生变更时，对软件系统进行相应修改的容易程度。一个易于维护的软件系统也是一个易理解、易测试和易修改的软件，以便纠正或增加新的功能，或允许在不同软件环境之间的易于操作。（6）可移植性：从一个计算机系统或环境转移到另一个计算机系统或环境的容易程度。

软件质量度量的实施：（1）确定软件质量需求； 在用户需求中，除功能需求外，还有非功能需求，包括：质量需求、环境需求、设计约束、开发策略等。质量需求是用户比较关心的内容。过程：**需求获取：**首先，你要理解用户的需求，区分哪些是质量需求，把这些需求记录下来，获得用户的确认。**需求分析：**拿到用户确认的需求后，你可以开始把用户的质量需求与我们设定的质量特性联系起来，一直区分到子特性。这种联系，就是把用户语言描述的需求，转变为计算机工程师语言的需求。建立了这种关联后，可以根据分类，分级，确定直接度量。**（2）确定直接度量。**直接度量就是实际的软件质量测量活动，它的输入是软件或软件过程，输出是一个测量值。它通过执行一系列的任务，获得一个质量值。在进行直接度量前，你应该有以下准备：**1）工具：**有助于计算度量值的硬件/软件工具，如：缺陷跟踪工具；**2）应用：**描述度量结果的希望值、度量值的意义、作用和对度量结果数据的使用方法；**3）数据：**获得度量结果所需的数据、程序、过程等度量对象；**4）计算：**度量程序、步骤和方法。**5）费用：**测试是要花钱（人力、物力、时间等）的。**（3）分析度量结果。**对度量过程进行跟踪和分析，需要时，可能会对度量程序、度量工具、度量方法，甚至原始数据，做出补充和调整。**（4）确认质量度量。**在度量过程中，进行度量结果的确认非常重要。**首先**，要确认度量过程是否与事实相符，脱离现实真实的度量，与目标再相符的结果也是没有意义的。**其次，**是确认方法的有效性。**其他度量：分析模型的度量，设计模型的度量，源代码的度量，对测试的度量， 对维护的度量。**

软件确认与验证的概念：确认（Validation）是这样一个过程：它评价“在软件开发过程期间（针对单元）或结束（针对系统）时，单元或系统是否满足用户特定的需求”。因此，确认的是产品质量。确认活动围绕三个基本过程来开展：测试、度量、软件可靠性增长。验证（Verification）是这样一个过程：它评价“在一个给定的开发阶段中，单元或系统是否满足在此阶段开始时确定的条件”。验证的是产品开发过程质量——工作质量。验证活动也是围绕三个基本过程来进行：审查、度量、配置管理。

测试的 V 模式：V 模型中的过程从左到右，描述了基本的开发过程和测试行为。V 模型的价值在于它非常明确地标明了测试过程中存在的不同级别，并且清楚地描述了这些测试阶段和开发过程期间各阶段的对应关系。

单元测试：单元测试的内容主要是：算法**逻辑**、数据定义的理解和使用、接口、各种 CASE 路径、边界

条件、错误处理等。单元测试的目的通常是:在开发环境中，程序设计工程师为了检查单元程序模块内部的逻辑、算法和数据处理结果的正确性等。单元测试通常由负责编码的工程师自己在代码完成后测试，也有在项目组内，由工程师相互交叉测试。

调试与测试的最大的不同点是二者的目的和视角的区别：调试包括查找 BUG、定位 BUG、修改并最终确认 BUG 已经被修复的软件故障排除过程。测试是在一个相对独立的环境下(测试应尽可能地模拟运行环境，调试是在开发环境)，运行系统单元，观察和记录运行结果，对结果进行独立评价的过程。

单元测试（模块测试）（1）单元测试没有任何记录和文档。（2）由于调试的目标是获得没有故障的程序，与功能无关的程序属性往往被忽略，或者要到集成测试、确认测试时才被发现。由于单元测试在项目组中，常常由编码工程师完成，项目经理的管理一般并不深入到单元测试层。

集成测试（子系统测试）集成测试又称组装测试，它是在单元测试完成后，组装为一个子系统后，对下列只有组装后才能发生和测试到的问题，进行检查：（1）组装后一个模块对一个模块的影响；（2）合并功能是否是预期的；（3）独立的误差在合并后的变化，是扩大还是减小，是否在可接受的范围内；（4）实际的接口测试；包括：模块之间对实际衔接的标准、时序（实时性)、应答响应、容错与错误处理等；（5）模块间的资源竞争等。

确认测试（系统测试）确认测试的目的是按照与用户确认的软件需求规格说明书的要求，检查系统的需求实现。**确认需求**的测试依据是需求阶段产生的测试脚本。 国内项目组的现实情况有以下几种：（1）没有确认测试；（2）没有独立的确认测试，测试与设计、编码不分离；（3）有独立的确认测试，但测试用例是设计和编码人员写的，因此，独立测试人员相当于按设计和编码人员的设计思路再测一遍。确认测试还包括软件经修改后的再测试（回归测试)。**回归测试**是对已测试并发现故障的部分，修改后进行再测试。回归测试不应修改测试程序、测试内容或测试标准。它与正常测试不同的仅是：它可能并不需要再完整地走一遍所有的确认测试，而是小心地选择部分确认测试程序，选择的标准是不减低原标准的整体要求。

验收测试：验收测试与确认测试非常相似，所不同的是，确认测试是项目组或组织内部的测试，验收测试是用户主导、现场参与、现场环境下的测试。验收测试通常由项目组先提出测试大纲，定义测试目的、范围、方法、测试用例、预期结果、验收标准等。经用户同意批准，可能包括用户的修改、增加后，确定测试时间，开始进入验收测试。用户在完成按测试用例的测试后，在测试记录上逐条确认、签字，最后，在测试报告上签字，完成验收测试。一般地、验收测试报告是项目初验、终验的依据和主要验收形式。

单元测试和验收测试没有什么区别？单元测试可以类比为一个建筑的质检人员对建筑进行的检测， 他关注的重点是建筑的内部结构、地基、框架以及墙壁是否垂直等。他的检测是要保证建筑的各个部分是正常的、安全的，换句话说，就是要保证施工满足建筑上面的质量标准。验收测试可以类比为建筑的使用者来对建筑进行的检测。他关心建筑的外观是否美观、各个房间的大小是否合适，窗户的位置是否合适，是否能够满足家庭的需要等。这里，建筑的使用者执行的就是验收测试，他是从用户的角度出发的。正是这种角度的不同

决定了单元测试和验收测试之间的区别。它们是对系统的不同的方面进行的测试，二者是互相补充的。不管我们在系统的构建中使用了多么聪明的方法，不管我们的系统是多么的灵活，但是首先我们的产品必须是可用的，否则我们所做的就是浪费时间，从这一点上来说验收测试要比单元测试显得更加重要。

测试方法：白盒测试在单元测试阶段，由于测试者对被测对象的内部结构、逻辑思路、接口关系等比较熟悉，一般采取白盒测试的方法，它是根据模块的内部逻辑，进行测试设计的方法。有些集成测试也采用白盒方法，关键看集成阶段的划分。 **黑盒测试**在集成测试以至此后的各阶段，测试设计和测试人员，对被测对象的内部结构不了解也不需要了解，他的目的是按需求功能进行确认。因此，黑盒测试是严格按软件需求进行测试设计的方法。**代码走查**

测试类型（1）功能测试：软件实现的功能是否符合需求规格说明书中定义的功能；（2）性能测试：软件在规定配置下的性能是否符合需求规定；（3）算法测试：确认实现的算法的正确性；（4）正向测试：按照用户正常的理解、操作方式、思维和使用习惯使用软件，得到的结果是否与需求一致。（5）逆向测试：如果不按用户正常的理解、操作发生、思维和使用习惯使用软件，软件是否能正确地进行处理。（6）边界测试：按软件的限制、假设条件的边界输入，进行测试。（7）配置测试：对软件环境进行配置变化，软件需求实现，特别是性能实现是否能符合需求规定要求。（8）负载测试：在业务处理量、数据负载量、通讯负载量达到何种情况，系统的性能变化和承载能力情况。

测试计划在拟定测试计划时，首先需要对以下情况，做出估计：（1） 完成测试设计所需要的工作量（2）完成测试设计所需要的工作时间（3） 完成测试所需要的时间：

测试分配：测试计划确定了测试的范围、内容和估计时间，根据 WBS 方法，测试计划还应说明具体测试任务的分解和测试工作的分配。测试组的成员根据分工，各自完成一部分测试任务。测试组与项目开发组还需要保持一定的同步，使测试与开发、修改在协调的步骤下进行，以节约宝贵的项目总时间。

测试报告：收集齐上述的所有测试用例，构成了测试报告的基本要件。测试报告是对所有测试用例测试过程的总结。

在测试报告中，应反映：（1）测试中出现问题的统计汇总和分析；（2）未解决问题的汇总和解决方案建议；（3）回归测试的统计和分析（度量） ；（4）对测试计划的总结或修改。

测试过程一般如下：（1）测试准备：制定人员、环境、工具、培训和外部支持计划。（2）测试计划：确定测试策略、建立测试计划。（3）测试用例：建立测试顺序树、确定测试的优先级、详细列出测试程序和测试数据，设计测试用例。（4）测试环境：了解需求、搭建环境、安装备份和恢复程序，记录初始环境、测试环境、恢复环境等。（5）测试执行：从测试计划复审测试计划进度表、恢复测试执行环境。（6）结果分析：执行结果分析、度量。（7）测试报告：错误趋势图、测试变动指示、产品检查点建议。

评审的目的是：（1）在软件开发过程中，尽早可能地发现问题，特别是过程性的问题；（2）确保对需求保持一致的意见；（3）验证任何修改和变更满足预先定义的准则；（4）为组织提供产品在质量和过程方面是

否有效的实际数据；（5）使团队成员之间在技术上建立相互的了解；（6）增加软件确认测试的有效性；（7）提高优秀软件工程师的水准。

软件审查的准备：评审人：审查一般由一个审查小组或审查委员会负责进行，审查小组内，应有以下角色构成：（1）主持审查活动的主审员；（2）被审查产品的负责人，包括产品经理、技术经理、质量经理等；（3）负责对被审查产品进行讲解和解释的主讲人；（4）来自各有关部门的审查员；（5）记录员；（6）项目经理

审查员的职责：作为被审查对象的项目组，按照审查组的要求，提交被审查材料，接受审查。

审查员，应该做什么准备？首先，明确作为审查员的定角色位、职责。审查员是那些具有相关知识和对被审查产品具有一定熟悉程度的，但不一定就是直接从事相同岗位（有时，还特别需要交叉换位）的人员。在参加审查前，他必须花一定的时间和精力，来了解产品，并能通过阅读提交的资料，了解产品与文档、标准和规范之间的差异。因此，他在审查中的责任是：（1）必须完全熟悉要审查的产品和产品所依据的文档和标准；（2）对照产品和文档，鉴别其中的差异；（3）客观地评价差异，识别是属于实现的程度差别、缺陷，还是错误；（4）判断差异是实现的个体现象，还是过程问题；（5）以对产品而不是对人的态度，对差异进行评估和分析；（6）向主审员报告审查结果和分析意见。

需求审查过程：需求审查遵循这样的过程：组织审查组；收集项目组提交的被审查资料；确定审查日期；审查员在获得审查任务分配和开始工作，包括：对资料的阅读和评审、做实地的检查、调查和询问、记录并报告；参加评审会议并报告自己的发现和分析。审查小组首先检查审查活动是否充分和没有偏差、疏漏。审查员对问题的认识有没有片面和主观。主审员根据自己的经验，可能会对年轻的审查员要求做出补充调查。通过讨论，审查小组争取对问题取得一致的意见，并形成审查报告。

追踪与改正：审查的目的是监督项目组对软件的品质，保持良好的状态和不断地改进。因此，审查小组有责任跟踪项目组对审查结果的利用情况。关注项目组的改进，是项目经理比关注审查结果更重要的事情。

设计审查：概要设计审查表（问题清单）详细设计审查表（问题清单）

概要设计重点审查以下几个方面（1）概要设计对需求的完整实现；（2）概要设计与需求的一致性；（3）概要设计向需求的反向可追踪；（4）概要设计中，对系统结构设计的逻辑性、合理性和可扩展性。

代码审查：代码的审查与具体实现工具有关，而且与具体实现工具的版本有关。代码的标准化和设计技巧，可以作为审查的范本（如果必要的话）。代码审查的一个办法是走查。就是由审查人员“读”工程师写的代码，然后对照“标准”进行检查，是对软件文档的一种书面检查。它通过人工模拟执行源程序的过程，检查软件设计的正确性。

测试审查：（1）对测试用例的审查：测试用例的哪些要素（用例名、测试日期、预期测试结果等）是否齐备？（宽度）（2）在概要设计和详细设计中确定的关键点或特殊需求是否都测试到了？（深度）（3）测试过程（步骤、环境、用户模拟等）的设计是否正确、恰当？（4）预期值与结果值的差异统计；（5）测试目

的是否达到?

现代质量管理在以下方面，做出更多的强调：（1） 以客户满意为质量目标；（2） 比注重结果更多地注重过程；（3） 管理层对质量负有责任。

PMBOK 的质量管理即项目管理知识体系，是美国项目管理协会（PMI）对项目管理所需的知识、技能和工具进行的概括性描述。

CMMI 主要关注点就是成本效益、明确重点、过程集中和灵活性四个方面。

CMMI 五个级别：完成级，管理级，定义级，量化管理级，优先级

第 14 讲　软件能力成熟度模型 CMM

软件过程:研究的是如何将人员、技术和工具等组织起来，通过有效的管理手段，提高软件生产的效率，保证软件产品的质量。

软件项目成功的三要素 People(人),Process(过程),Technology(技术)

软件工程难题:产品质量低下、进度延误、费用超支…（软件工程学科发展 30 年尚未彻底解决）.经典软件工程：研究需求分析、系统设计、编程、测试、维护等领域的方法、技术和工具.问题之源：人们逐渐意识到，由于企业管理软件过程的能力比较弱，常常导致项目处于混乱状态。过程混乱使得新技术、新工具的优势难以体现。经典的软件工程不是不好，而是不够用。用于提高软件**过程能力**的实践通称为软件过程改进。

软件过程改进:提高软件过程能力的实践通称为软件过程改进,软件过程改进的根本**目的**是：提高质量、提高生产率并且降低开发成本。

软件过程改进必须走规范化之路

软件过程的三个流派:CMU-SEI 的 CMM/PSP/TSP;ISO 9000 质量标准体系;ISO/IEC 15504（SPICE）.

CMM 是用于衡量软件过程能力的事实上的标准，同时也是目前软件过程改进最好的参考标准。

CMM 五个级别：初始级（有纪律的过程）可重复级（标准一致的过程）已定义级（可预测的过程）已定量管理级（持续改进的过程）持续优化级。

2 级 **CMM** 共有 **6** 个 **KPA**，主要涉及建立基本项目管理和控制方面的内容：需求管理；软件项目计划；软件项目跟踪与监督；软件子合同管理；软件质量保证；软件配置管理。

需求管理 RM：在顾客和顾客要求的软件项目之间建立一种一致的、共同的理解，控制系统对软件的需求，为软件工程和管理建立基线，保持软件计划、产品和**活动与分配需求的一致性。共同理解是计划和管理软件项目的基础，所以需求管理要求需求文档化，并对需求进行评审。在整个软件生命周期中，当用户需求改变时，记录全部改变，并进行评审。**

软件项目计划 SPP(SDP) ：为实施软件工程和管理软件项目制订合理计划。**SPP** 包括**:** 估计软件产品的规模和所需资源，制定开发进度计划，确定并评估风险，协商有关约定。在 **SPP** 执行过程中，记录并维护 **SPP**，严格按照 **SPP** 实施软件项目。

软件项目跟踪与监控 SPTO ：对软件项目的执行进行跟踪、监督与控制，以便随时掌握软件项目的实际开发过程。当项目的执行活动与软件项目计划 **SPP** 偏离时，项目经理能够采取有效的措施进行处理。管理活动包括：按计划对软件完成情况和结果进行跟踪和评审，必要时进行纠正。纠正活动包括：修订软件开发计划以反映项目的实际完成情况，重新计划剩余的工作，并采取适当的措施改进。

软件子合同管理 SSM：选择高质量的软件子项目承包者，并对子合同的执行进行有效的管理。选择合适的子项目承包者，并对其进行过程监督。

软件质量保证 SQA ：对软件项目和产品质量进行监督和控制，为管理者监控软件项目的过程和产品质量提供适度的可视性。通过评审和审核软件产品和活动，验证是其否与应用的标准和规程一致。出现的问题尽可能在软件项目组内部解决。内部不能解决的问题，由质量保证组进行适当的解决。

软件配置管理 SCM：保证软件项目生产的产品在软件生命周期中的完整性。

3 级 CMM 在 **2** 级的基础上增加了七个 **KPA**，既包括项目管理问题，又包括组织问题和工程问题。

软件组织建立了一个基础设施，对项目中所有有效的软件工程和管理过程的实施制度化。

KPA3.1 组织过程焦点 **OPF：**软件组织建立负责软件过程活动的责任和机制，为改进软件组织的整体软件过程能力提供**组织上的保证**。

KPA3.2 组织过程定义 OPD：由负责软件过程活动的小组开发和维护一组能提高项目软件过程整体效能的软件过程资源的集合，并为在定量过程管理中确定有意义的数据提供基础。过程定义产生了企业标准软件过程及相关过程资源。

KPA3.3 培训大纲 TP：为提高个人的技能和知识进行培训，使其更有效地完成任

KPA3.4 集成软件管理 **ISM：**协调软件工程活动和软件管理活动，将二者集成为密切相关、定义完整的软件过程。ISM 是在 2 级 CMM 的软件项目计划 SPP 和软件项目跟踪与监控 SPTO 的基础上发展起来的，主要包括软件组织的标准软件过程及与之相关的操作。

KPA3.5 软件产品工程 SPE：协调一致地执行经过定义并综合了所有软件工程活动的工程过程，以便高效地生产出稳定的软件产品。**SPE** 的任务：分析分配给软件的系统需求、制定软件需求、开发软件体系结构、设计软件、编码、测试、集成等，检验其是否满足分配需求。

KPA3.6 组间协调 IC：开发软件项目的各小组积极合作，以便使项目更好、更有效地满足用户需求。**一个软件项目**一般要设置若干个小组**(**例如，**软件工程组**、软件**估计组**、系统**测试**组、**软件质量保证组**、软件**配置管理组**、**合同管理组**、**文档支持组**等)。 这些小组只有通力协作、互相支持，才能使项目在各方面都能更好地满足客户的需要，因此，各工作组之间必须很好地协作。

KPA3.7 同行评审 PR：由同一级别的其他人员对该软件项目产品系统地检测，以便能较早地和有效地发现软件产品中存在的错误，并改正它们。**PR** 的目的是尽早和高效地除去软件工作产品中的缺陷，增强对软件工作产品和可预防的缺陷的了解。**PR** 是一种重要而又有效的工程方法，例如，可通过排查、审查或者其它评

审方法来实施。

CMM 4 级的 KPA：CMM 4 级在 **3** 级的基础上增加了两个 **KPA**，关注焦点是建立起对**软件过程**和产生的**软件产品**的**定量**理解。

CMM 5 级的 KPA：CMM 5 级在 4 级的基础上增加了 3 个 KPA，关注的焦点是为了实施连续不断的和可测量：软件过程改进，组织和项目所必须解决的问题。

CMM 的公共特性:每个成熟度级别表明一组过程能力;过程能力：遵循某个软件过程后，可能实现的预期结果的程度。关键过程域(KPA)：一组相关活动，达到一组目标。目标：某个 KPA 中的 KP 所表示的范围、边界和意图。每个 KPA 有若干个目标。公共特性：指出了一个 KPA 的实现范围、结构要求和实施内容。关键实践：描述对 KPA 的有效实施和制度化起最重要作用的基础设施和活动。

KPA 的公共特性(Common Features):每个 **KPA** 都具有的五种公共特性：执行约定、执行能力、执行活动、测量和分析、验证实施。

执行约定(Commitment to Perform)：描述软件组织为确保建立并持续执行软件过程，所必须采取的措施，主要包括制定组织策略和构建领导体制等。**1.** 策略陈述。项目为实施 **KPA** 中的实践必须遵循书面的组织管理策略，以便使组织约定**(**制度**)**在项目实施中能得到落实。**2.** 领导体制。有些 **KPA** 中的执行约定包括指定领导的职责及组织保证。

执行能力(Ability to Perform)：描述了项目或组织能成功地执行软件过程所必须满足的前提条件，通常包括组织机构、资源及资金、培训和前提。1. 组织机构。某些 KPA 中包括专门为支持该 KPA 所建立的组织机构。例如，软件质量保证组(SQA)、软件配置管理组(SCM)、软件工程过程组(SEPG)等。2. 资源和资金。资源和资金包括三部分：技术：KPA 可能涉及到的专业技术；资金：实际可供使用的资金（不是预算资金)；工具：KPA 可以使用的工具，通过实例给出活动可能使用的工具。3. 培训。包括正式培训和非正式培训，其目的是向组织的相关人员传授知识和技能。培训方式主要有：课堂教学培训、电视教学、CAI 教学、师付带徒弟、定向培训（传授一般技能或知识）

• 分级培训别：2 级 CMM 强调“接受培训”，指一般培训；3 级 CMM 强调“接受所需的培训”，指高级培训。4. 前提。前提是指前一个 KPA 的输出结果，实施本 KPA 的条件是“前提”应完成，并以此前提的结果作为本 KPA 的输入。

执行活动(Activities Performed)：软件组织执行一个 KPA 时所必需执行的活动、任务、职责分配和规程，包括制定计划和规程、实施工作、跟踪，并在必要时采取正确的措施。1. 计划制定和修订。正式计划: (1) 依据书面规程制定和修订;(2) 必须依照计划执行； (3) 经过高级经理的批准。非正式计划：作为正规计划的一部分记入文档，或正规计划的补充。**2.** 计划实施。大多数 KPA 都有一个或若干个关键实践 KP 描述按照书面规程实施的活动。全面实施文档化。负责一项任务或活动的人员必须按照书面规程来实施活动，以便使工作能重复，使对该领域有一般知识的人也能以同样的方式学习或完成这些任务和活动，并取得满意的结果。

3. 跟踪并采取正确的措施与管理。 对执行的活动进行跟踪，并进行管理和控制。CMM 2 级使用术语“跟踪……并适当采取正确的措施”，没有严格定义，其管理活动仍是“反应式”的。CMM 3 级使用术语“管理……”，有严格的定义，其管理不仅能预计问题的发生，也能阻止问题的发生。

测量和分析(Measurement and Analysis)：描述测量的基本规则，以便确定、控制和改进软件过程的状态。包括：培训大纲的质量、管理的有效性、软件产品的功能和质量。

验证执行(Verifying Implementation)：为保证所实施的活动是否遵循了已制订的软件过程步骤，对实施活动通过管理和软件质量保证进行评审和监督活动。**1.**高级经理定期审查。定期评审的目的是适时地掌握软件过程活动，以便采取适当的措施。**2.** 项目经理定期或不定期评审和监督。根据项目的具体情况，在不同阶段进行评审，以便了解项目的工作情况，清楚软件项目中发生的重要事件，比高级经理审查监督更详细更具体。**3.** 软件质量保证活动。 由软件质量保证小组 **SQA** 进行评审或审核活动。若 **SQA** 不起作用，就无法实现 **SQA** 验证活动。

CMM 的关键实践（Key Practices）：每个 KPA 有一组目标和一组公共特性；每一个公共特性用一组关键实践 KP 来描述。；KP 描述对关键过程区域的有效实施和规范化的基础设施和活动。

CMM 通过内部结构特性：CMM 通过内部结构的规范，使软件组织能够制定方针、政策、标准，并参照自身的特点来建立软件过程，以提高软件过程成熟度。这些建设性的活动对软件组织开展业务、进行实践工作、改进过程的基本环境和企业文化起到了极大的支持作用。在这样的组织中，一旦组织内的人员发生变化，组织内部开展的各项活动仍然可延续下去。

第 16 讲 软件过程与能力成熟度模型—CMMI

软件工程 CMM 的成功启发其它学科领域也开发类似的改进模型，从而**衍生出多种 CMM**，如：（1）SW-CMM 软件 CMM（2） SE-CMM 系统工程 CMM（3） SA-CMM 软件采购 CMM（4） IPT-CMM 集成产品群组 CMM（5） IPPD-CMM 集成产品群组 CMM（6） P-CMM 人力资源能力成熟度模型。

同一个组织可能**采用多种过程改进模型，因而带来一些问题**：不同改进模型(衍生 CMM)有一些对相同事物说法不一致，或活动不协调，甚至相抵触；要进行一些重复的培训、评估和改进活动，因而增加了许多成本；不能集中其不同过程改进的能力以取得更大成绩。

CMMI 全称为：Capability Maturity Model Integration，即能力成熟度模型集成。

CMMI 能帮助建立过程改进目标和次序,为质量过程提供了指南,为评估当前实践提供一个准绳。

CMMI 由**三个基本成熟度模型为基础**综合形成： （1）SW-CMM 软件 CMM（2）SE-CMM 系统工程 CMM（3）IPPD-CMM 集成产品群组 CMM，它主要面向并行工程。（4）经常还会引入外购协作 CMM 作为 CMMI 第 4 个模型源，即 SS-CMM。

具体实践：不同组织可根据需要选择 CMMI 模块，也可同时选择多个模块。例如：纯软件企业可以选择 CMMI 中的软件工程模块，设备制造企业可以选择系统工程和外购模块，集成企业可以选择软件工程、系统

工程和集成产品和过程开发。

CMMI 与 CMM 不同，它不但提出了软件能力成熟度模型，还提出了软件过程能力等级（Capability Level,CL)模型。它显示一个组织在实施控制其过程，以及改善其过程性能等方面所具备、或设计的能力。

过程能力等级包含：某个过程若干相关的特定实践，共性实践。好处是：执行这些实践，该组织的过程执行能力就能得到提高；进而增强该组织的总体过程能力。着眼点是：使软件组织走向成熟，以增强实施和控制软件过程的能力，改善过程本身的性能，这些能力等级有助于软件组织在改进各个相关跟踪、评价和验证中项改进过程。

两种过程改进的方法:组织成熟度，过程域能力

CMMI 对不同过程改进方法采用不同**表示法：**组织成熟度-分级(阶梯式)表示法，过程域能力-连续表示法

分级表示法：规定了一系列已经证明的改进措施,每一级都是其上一级的基础,服务于上一级，运用成熟度等级,使得组织之间的比较成为可能，组织成熟度体现于一组过程域.

连续表示法：允许选择改进的次序,使其最适合组织的商业目标,减少组织的风险，以过程域为基础,使得组织之间可以在同一过程域内进行比较，提供一个简便的由 EIA/IS-731 转换至 CMMI 的方式。

比较这不同的表示法：两种表示法都提供了执行过程改进达到组织目标的方法；两种表示法提供的实质内容是相同的,只不过是内容的组织方式不同而已.

过程域能力和组织成熟度的关系：过程域能力和组织成熟度具有相似的概念。二者的区别是：过程域能力只与单一的过程域或实践相关；而组织成熟度包含一组既定的过程域。一般来说,如果一组过程已被评估确认达到某个成熟度等级,那么这些被评估的过程会对应相关的过程域能力水平。

在连续型模型中，每个过程域都被分为 **6 个能力等级**，其特征如下表所示：CL 0 不完备级 Incomplete ，CL 1 已执行 Performed，CL 2 已管理 Managed，CL 3 已定义 Defined，CL 4 量化管理 Quantitatively Managed，CL 5 优化 Optimizing。

CL0 不完备级：过程未执行或执行不完整，特定目标中有不完整的部分。CL1 已执行级：特定目标都能满足，基本活动都得到执行。CL2 已管理级：过程除了得执行，还需要按照计划和组织方针来进行实施；相关的人员得到与执行有关的培训；为了过程的执行分配了相关的资源；生成的工作产品收到控制；利益相关的方面都参与了过程的执行，并且进行了相关的评审以及过程符合度的验证；管理层关心过程的制度化状况和过程的其他目标，例如成本、日程和质量目标。CL3 已定义级：在已管理的过程基础上，还具有如下特征：该过程是组织的标准过程裁减得来的，裁减的依据是组织的裁减指南；该过程还向组织的过程资产库贡献关于工作产品、度量数据以及其他的过程改进信息。CL4 量化管理：在已定义基础上，还具有如下特征：过程是使用统计的以及其他量化管理的手段来进行管理的。；在过程管理中使用了量化管理的质量和过程性能指标作为管理的标准；用统计手段来理解质量和过程性能，并且在整个生命周期内进行管理。CL 5 优化：优化的

过程除了是一个量化管理的过程之外，还具有如下的特征：过程能够及时的变更或采用来满足当前的或预期的业务目标；优化的过程聚焦于使用增量的和创新技术进步手段来达到不断改进过程性能的目的；过程偏差的根本原因得到识别，并且针对这些原因进行采取相应的改进措施；这些措施按照能够度量的方式被识别、评价和实施；这些改进过程的选择是基于对组织量化过程的理解，以及这些改进措施的预期收益、成本以及影响程度；优化过程的性能能够不断提高。

CMMI 与 CMM 的异同：CMMI 有两种表述方式，阶段表述方式与 CMM 兼容，连续表述方式与 ISO/IEC 15504 相似。

CMMI 的新特性①CMMI 模型中比 CMM 进一步强化了对需求的重视。在 CMM 中，关于需求只有需求管理这一个关键过程域，也就是说，强调对有质量的需求进行管理，而如何获取需求则没有提出明确的要求；在 CMMI 的阶段模型中，3 级有一个独立的关键过程域叫做需求开发，提出了对如何获取优秀的需求的要求和方法。②CMMI 模型对工程活动进行了一定的强化。在 CMM 中，只有 3 级中的软件产品工程和同行评审两个关键过程域是与工程过程密切相关的；在 CMMI 中，则将需求开发，验证，确认，技术解决方案，产品集成这些工程过程活动都作为单独的关键过程域进行了要求，从而在实践上提出了对工程的更高要求和更具体的指导。③CMMI 中还强调了风险管理。在 CMM 中把风险的管理分散在项目计划和项目跟踪与监控中进行要求；CMMI3 级里单独提出了一个独立的关键过程域叫做风险管理。④保留了 CMM 阶段式模式的基础⑤增加了连续式模型。可以帮助组织其客户更加客观和全面的了解它的过程成熟度；可以给组织在进行过程改进的时候带来更大的自主性；不用再象 CMM 中 一样，受到等级的严格限制；这种改进的好处是灵活性和客观性强，弱点在于若缺乏指导，一个组织可能缺乏对关键过程域之间依赖关系的正确理解而片面的实施过程，造成一些过程成为空中楼阁，缺少其他过程的支撑；两种表现方式（连续的和阶段的）从他们所涵盖的过程区域上来说并没有不同，不同的是过程区域的组织方式以及对成熟度（能力）级别的判断方式。

CMM：有次序的、基于活动的度量方法与管理规范；与瀑布过程的有非常密切的联系，更适合瀑布型的开发过程。

CMMI 相对 CMM：更一步支持迭代开发过程；支持经济动机推动组织采用基于结果的方法；虽然 CMMI 保留了基于活动的方法，它集成了软件产业内很多现代的最好的实践，但淡化了与瀑布思想的联系。

CMMI 评估方式：自我评估：用于本企业领导层评价公司自身的软件能力；主任评估：使本企业领导层评价公司自身的软件能力，向外宣布自己企业的软件能力。

CMMI 的评估类型：软件组织的关于具体的软件过程能力的评估；软件组织整体软件能力的评估（软件能力成熟度等级评估）。

CMMI 的基本思想 1、解决软件项目过程改进难度增大问题 2、实现软件工程的并行与多学科组合 3、实现过程改进的最佳效益。

简单改进过程：确定目前处于什么现状；确定想改进到什么程度；制定计划；执行计划；汲取经验教训

继续做。

PDCA 过程: 计划、执行、检查、改进。

CMMI 在 IDEAL 模型中的运用：①初始阶段。CMMI 模型能帮助组织了解如何发起并确定改进的基本内容②诊断阶段。用于过程改进的标准 CMMI 过程改进评估方法(SCAMPI^{SM})为基于 CMMI 的过程评估提供了准绳③建立阶段。CMMI 过程域注重于建立过程改进组.④行动阶段。 CMMI 模型为定义和改进过程提供了指南⑤学习阶段。 学习 CMMI 文档是组织进行过程改进的基础.

定义过程：①成熟的过程是文档化的②通常采用两种方式进行过程文档化:描述正式的过程；描述面向用户的过程③描述正式的过程：读者主要是过程专家；详细正规的描述；主要用于开发、剪裁和改进过程④描述面向用户的过程：读者主要是每天使用过程的用户；简单清晰的描述；主要用于执行过程。

过程描述下列事项: 在这个过程中将执行什么活动?谁来完成?为什么要完成它们?什么时候完成?如何完成?哪些输入是必须的?能有哪些输出?怎样测量其性能?

不一样的描述格式不一定都能方便地描述:活动的次序；活动的时间；活动中的数据流；分层次的细节；与标准的出入；叙述性的资料

描述格式的其它特征:灵活性；简单化；易于理解和培训；实用性。

过程描述格式：①通用的。数据流图；流程图；决策树或决策表；检查表；叙述②特殊的：ETVX (入口-任务-确认-出口)；SADT/IDEF0 (结构分析和设计技术)；信息图(Information Mapping) ③或是上述的综合。

活动细节：本次活动的目的是什么？ 谁参加了此活动？ 完成这项活动需要有哪些投入？ 通过这次活动能产生什么工作产品？怎么知道这次活动应在什么时候开始的?怎么知道这次活动到什么时候已经顺利完成的？为本次活动的完成做了些什么?为完成本次活动,列出 3-6 个子活动?怎么确定或测量这次活动的性能？在这次活动之前或之后还有什么活动?