

# 软件能力成熟度模型

## 内容摘要



Ch1 绪论



Ch2 CMM体系结构



Ch3 可重复级(第2级)



Ch4 已定义级(第3级)



Ch5 已管理级(第4级)



Ch6 优化级(第5级)



Ch7 CMM过程控制和ISO9000评价标准的区别

## 第一章 绪论

### 1.1 CMM背景



20世纪60年代，软件危机爆发，其主要表现为：

- 软件开发费用和进度失控
- 软件的可靠性差
- 生产出来的软件难以维护
- 难以满足用户要求
- . . .



失败的软件项目仍然比比皆是，软件灾难频发：



专家们的努力

- 软件专家们花费了数十多年的时间寻找解决软件问题的“silver bullet”
  - 新的程序设计语言
  - CASE 工具
  - 开发方法
- 但是，Many of these solutions became shelfware



人们做了什么？

- 开发新的程序设计语言？
- 发明新的 CASE 工具？
- 发明新的方法？
- 但是，with no significant improvement.



还能做什么？

- 雇用更好的软件专家?
  - 雇用更多的软件专家?
- 为什么不尝试“管理软件过程”?

1984年SEI成立

很多项目失败是由于只关注结果

而不关注过程引起的

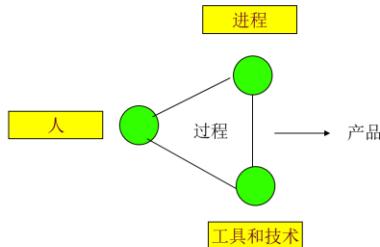


什么是过程

- 什么是“过程”?
- 过程 (process) 是为实现给定目标所执行的一系列操作步骤。

• A process integrates

- People
- Tools
- Procedures



- 过程定义了“做什么” (What People Do)  
应用 procedures, methods, tools 将原材料 (input) 转换为对客户有价值的产品 (output)。
- 过程能带给我们什么?  
人们使用过程的方法和工具按照一定的次序就能将输入变成输出。



软件过程管理

- 对软件过程进行全面的、规范化及标准化的管理。
- 目的：保证软件产品具有相对稳定的质量
- 过程管理与项目管理是相互作用的：  
    过程是项目计划的依据，  
    项目执行所产生的数据又可作为过程改进的依据。



软件过程管理的主要内容：

- 过程定义：对最佳实践加以总结，以形成一套稳定的可重复的软件过程。
- 过程改进：根据过程的使用情况，对过程中有偏差或不够切合实际的地方进行优化的活动。

软件能力成熟度模型 (SW-CMM) 是由SEI研制的一种的软件过程管理方法。



CMM的概念

CMM是由SEI研制的一种用于评价软件生产能力并帮助其改善软件质量的一种方法，也就是评估软件能力与成熟度的一套标准。

CMM侧重于软件开发过程的管理及工程能力的提高与评估，是国际软件业的质量管理标准。

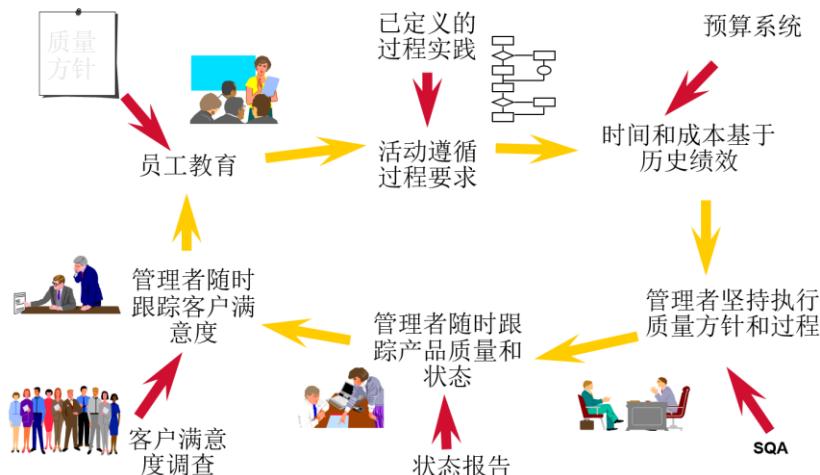


不成熟的软件过程的特征

- 没有定义好的软件过程
- 软件过程是临时准备的
- 管理者无法掌控
- 不能顺利交付成果
- 产品质量难于预测
- 时间和成本总是超出预算
- REVIEW和测试常常被简化



成熟的软件过程的特征



成熟的软件过程

- defined (you know what is done)
- managed (you can control the process qualitatively)
- measured (you know how much is done, and how well)
- controlled (you can control the process quantitatively)
- effective (you can improve the process rationally)

过程被文档化后才能成为过程规范，大家才可以“按规矩办事”。所以如果过程只存在于人的脑子里那是起不了规范作用的。

Maintainable: A good process evolves as technology changes.

Repeatable : Non-repeatable processes are of no use in planning.

Predictable: A process that does not aid in predicting cost, delivery time, and resource needs is of no use in planning.

Pro-active (as opposed to reactive): 积极的，有活力的

Followed: It's of no use if it's only on paper.

要实施一个好的软件过程就必须进行科学的软件过程管理。



从不成熟到成熟，过程改进能给我们带来什么？

- 提高生产力
- 改进质量
- 建立更好的项目控制
- 减少开发时间
- 节约成本



## Why CMM?

- 解决软件危机
- 研究如何有效地对软件开发项目进行管理，以便按照进度和预算完成软件项目计划，实现预期的经济效益和社会效益。
- CMM不是特定的软件开发技术，而是一种管理方法。整个软件开发过程标准规范化管理已成为任何软件开发企业获得成功的基础。



## 从数据看CMM的效果

每千行源代码所含的BUG数，

- CMM1级为11.95个，
- CMM2级为5.52个，
- CMM3级为2.39个，
- CMM4级为0.92个，
- CMM5级则只有0.32个

CMM5的软件开发周期是CMM1的36%，而生产成本是CMM1的19%，平均每个软件开发人员的生产率会提高4倍。

## 1.2 CMM的发展过程



### CMM的理论基础

- 20世纪30年代，Walter Shewhart公布了统计质量控制的原理
- W. Edward Deming提出了丹明链式反应的现象
- 20世纪60年代，Feigenbaum提出了全面质量管理（TQM）的概念
- 1979年，Crosby成立质量管理研究所，提出Quality is Free，应向零缺陷努力



### CMM的产生与发展

1987年9月 SEI发表软件能力成熟度框架和软件成熟度问卷



1991年 SEI推出了CMM 1.0版本



1993年 SEI推出了CMM 1.1版本



目前 CMMI (Capability Maturity Model Integration)



计划  
CMM 2.0



### CMM的主要特点

- 基于实际项目实践
- 最好的反映了实践的情况
- 反映了软件过程改进和评估执行人员的需求
- 形成文档
- 文档可以公开使用
- CMM是活的文档
- 在稳定性需求和不断的过程改进之间提供适当和实际的平衡点

## 1.3 CMM与软件产业



### CMM的用途

- 软件过程评估 (SPA)
- 软件过程改进 ( SPI )
- 软件能力评价 ( SCE )
- 设计CMM，就是为了指导软件组织通过判断当前自身的成熟度，提出自身软件质量和过程提高最为关键的问题，以此来选择过程的提高策略。



### CMM对中国软件产业是非常有价值的

中国软件企业要彻底摆脱困境，提高竞争力，就必须走规范化发展的道路，而CMM无疑是当今国际上最流行的软件企业质量管理标准，被誉为软件企业参与国际竞争的通行证。



### CMM不仅仅是一种认证，而是一种能力的改进和评估

## 第二章 CMM体系结构

### 2.1 CMM基础知识



软件过程(Software Process)是软件开发人员开发和维护软件及相关产品的全部生产活动和工程管理活动。

- 作用对象：软件及其相关产品
- 包括：活动、方法、实践和革新

只要过程上正确以及构成过程的解决方法正确，  
产品就会正确。

软件过程的好坏有事用什么来衡量的呢：

- 软件过程能力(Software Process Capability)  
描述在遵循了某一个软件过程后可能得到的预期结果的范围。
- 软件过程性能(Software Process Performance)  
描述在遵循了某一个软件过程后所得到的实际结果。
- 软件过程成熟度(Software Process Maturity)  
一个具体的软件过程被明确地**定义、管理、评价、控制和产生实效**的程度。
  - 成熟度表明了组织实施软件过程的实际水平
  - 成熟度包含着能力的一种增长潜力



### 软件组织的成熟与不成熟

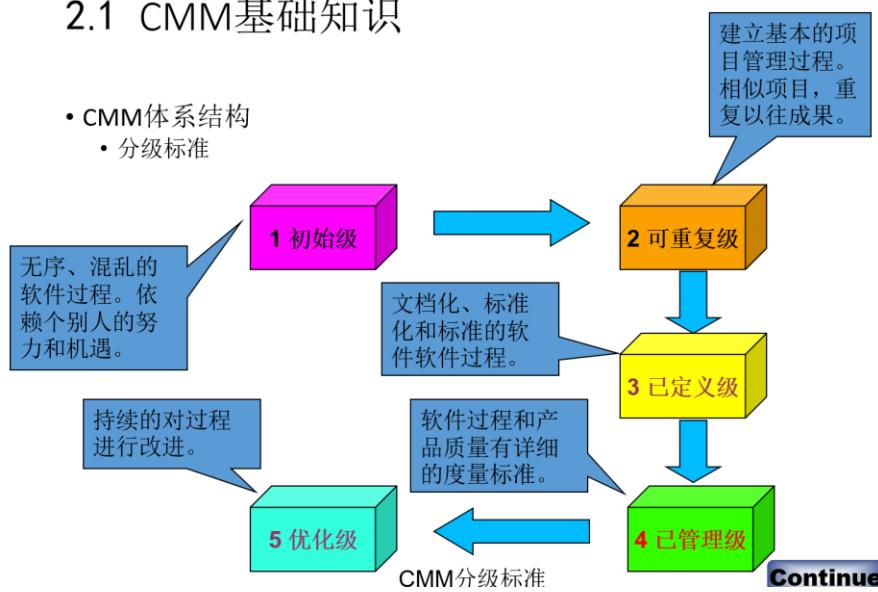
内容	不成熟的组织	成熟的组织
软件过程	没有进行预先计划，临时拼凑、不能贯彻。	有统一标准，切实可行并不断改进；通过培训，全员理解，各司其职，纪律严明
质量管理	问题判断无基准，质量难预测；产品交付前对客户是不可见的	产品质量有保证，软件过程有纪律，有必要的支持性基础设施。
管理方式	反应式（消防式）	主动式，监控产品质量和顾客满意程度。

进度 经费 估计	无实际根据，经常超支超时。硬性限时，常在功能和质量上作让步。	有历史数据和客观依据，比较准确。能实现预期目标。
----------------	--------------------------------	--------------------------



## 2.1 CMM基础知识

- CMM体系结构
  - 分级标准



各个级别的特点：

- (1) 初始级——软件过程的特点是无秩序的，混乱的，进度、预算、功能、质量不可预测，软件过程定义几乎处于无章法和步骤可循的状态，软件产品所取得的成功往往依赖极个别人的努力和机遇。常常在遇到问题的时候，就放弃原定的计划而只专注于编程与测试。
- (2) 可重复级——已建立了基本的项目管理过程，可用于对成本、进度和功能特性进行跟踪。对类似的应用项目，基于以往的项目的经验来计划与管理新的项目，有章可循并能重复以往所取得的成功。达到此级别的企业过程已制度化，有纪律，可重复。
- (3) 已定义级——软件过程均已文档化、标准化，并形成了整个软件组织的标准软件过程。全部项目均采用与实际情况相吻合的、适当修改后的标准软件过程来进行操作。
- (4) 已管理级——软件过程和产品质量有详细的度量标准，能够得到定量的认识和控制。企业要对所有项目的重要的过程活动进行生产率和质量的度量。软件产品因此具有可预期的高质量。达到该级的企业已实现过程量化。
- (5) 优化级——通过对来自过程、新概念和新技术等方面的各种有用信息的定量分析，能够不断地、持续性地对过程进行改进。

这种分级方式使得 CMM 模型具有了可操作性，软件组织也可以通过这种方式达到自己的目标。

模型以产品质量的概念和软件工程的经验教训为基础，指导企业如何控制开发、维护软件的生产过程和如何制定一套与之相适应的软件工程及管理体系。指导软件企业通过判断自身当前的过程成熟度，针对软件质量和过程提高中最为关键的问题，来选择过程的提高策略。将注意力放在具体的和可实现的目标上，并努力地通过模型中提供的措施和手段去实现这些目标。以此使企业不断地、渐进地和平滑地向高级阶段过渡，并最终实现软件企业的高速发展。



软件过程可视性 VS. CMM

### ■ 等级1

软件过程是一种混浊体一个黑盒。过程的可视性受到限制。由于活动阶段的划分做的差，管理者们极难确定项目进程和活动状态。需求以不受控的方式进入软件过程然后产出产品。软件开发常被看作为不可知的魔术。

- 等级2  
顾客需求和工作产品受到控制，建立了基本的项目管理过程。这些管理控制使得项目在规定点可视，软件过程可看作一连串黑盒子。随着活动在盒子间流动，在各过渡点的项目里程碑上具有管理可视性。尽管管理者可能不知道盒子内发生事情的细节，但是过程的产品和用以证实过程正在运行的检验点都是确定和已知的，当问题出现时管理者能对它们作出反应。
- 等级3  
盒子的内部结构，即项目规定的软件过程中的作业是可视的，反应了组织的标准软件过程。项目参加者了解他们在过程中的岗位和责任，知道他们的活动如何在相当的细节层次上相互影响。由于已定义的过程提供对项目活动很好的可视性，所以项目外的人能准确而及时地得到状态更新信息，管理者对可能产生的风险预先有准备。
- 等级4  
已定义的软件过程具备定量特性并受到定量控制。管理者们能够度量进度和问题，他们作决策时有一个客观的定量的基础，随着过程的变异性越来越小他们预测结果的能力稳定增长，变得越来越准确。
- 等级5  
为了提高生产率和质量，团队以受控的方式不断尝试新的和改进的构造软件的方法。无效的或者已出错的活动得到标识和替代或修改，有纪律的更改成为一种生活方式，洞察力延展到现行过程以外，并且能深入了解潜在变更的作用。管理者们能定量的估计更改的影响和有效性然后跟踪它们。可见，在过程成熟度的每个等级上显示给管理者的项目状态和表现的可视性程度随成熟度等递增软件过程可视性亦提高。

## 2.2 CMM主要内容



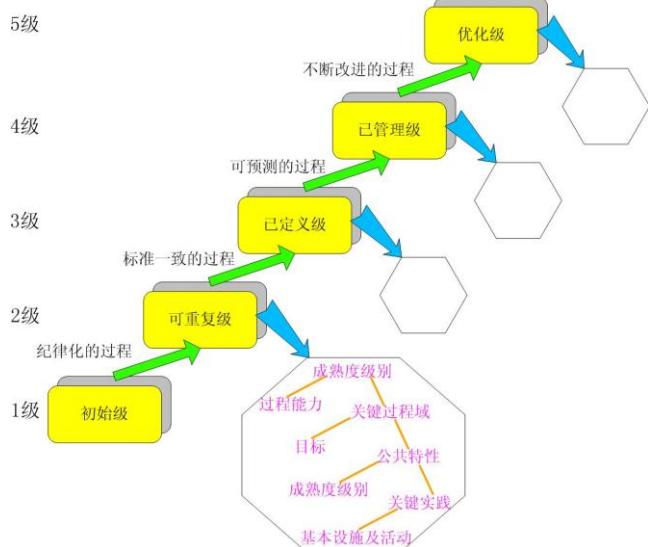
### CMM体系结构

- 内部结构  
除第1级以外每个成熟度级都由若干个关键过程域组成，各关键过程域中规定了执行约定、执行能力、执行活动、度量和验证的标准等。
- 组织保证  
管理者：经理、各级经理、领导、职员和个人  
软件小组：软件工程组、软件工程过程组、软件相关组等



### CMM的内容

- CMM为软件企业的过程能力提供了一个阶梯式的进化框架，采用分层的方式安排它的组成部分，以适应不同机构使用的需要





## 各级别改进方向

### ■ 初始级

类型	内容
过程特征	软件过程不稳定，项目执行无序、混乱，没有稳定的开发环境
工作组	可能存在
度量	没有进行数据集成和分析
改进方向	<ul style="list-style-type: none"><li>● 建立项目管理</li><li>● 完善需求</li><li>● 建立软件项目计划</li><li>● 开展SQA</li></ul>

### ■ 可重复级

类型	内容
过程特征	规则化的
工作组	系统测试组、软件评估组、软件质量保证组、软件配置管理组、合同管理组、文档支持组和培训组
度量	每一个项目建立资源计划
改进方向	<ul style="list-style-type: none"><li>● 总结项目成功经验</li><li>● 确定全组织的标准软件过程</li><li>● 建立SEPG</li><li>● 积累数据</li><li>● 加强培训</li></ul>

### ■ 已定义级

类型	内容
过程特征	标准的、一致的
工作组	增加了：软件工程过程组、软件工程活动组、软件评估组
度量	<ul style="list-style-type: none"><li>● 全过程中收集使用数据</li><li>● 全项目中系统性的共享数据</li></ul>
改进方向	<ul style="list-style-type: none"><li>● 软件过程的定量分析</li><li>● 通过质量管理达到软件的质量目标</li></ul>

### ■ 已管理级

类型	内容
过程特征	可预测的
工作组	增加了：软件相关组、定量过程管理活动
度量	<ul style="list-style-type: none"><li>● 全组织内进行数据收集与确定</li><li>● 度量标准化</li><li>● 数据用于定量的理解软件过程及稳定软件过程</li></ul>
改进方向	<ul style="list-style-type: none"><li>● 缺陷防范</li><li>● 主动进行技术改革管理、标识、选择和评价新技术</li><li>● 过程变更管理</li></ul>

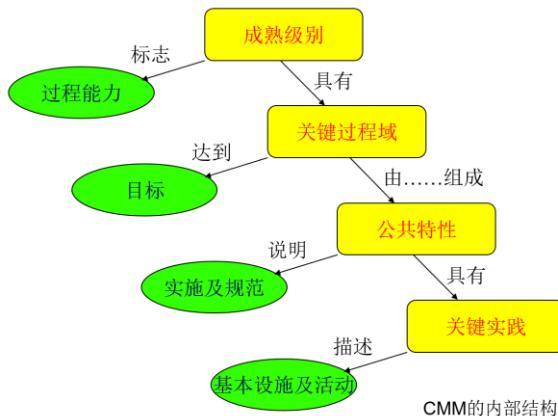
### ■ 优化级

类型	内容
过程特征	不断改进
工作组	增加了：软件相关组、缺陷防范活动协调组、技术改革管理活动组、软件过程改进组
度量	数据评估，选择过程改进
改进方向	保持持续不断的软件过程改进



### CMM的内部结构

- CMM由5个成熟度级别组成
- 每个成熟度级别（除级别1）包含了实现该级别的若干个关键过程域（KPA）
- 每一个KPA进一步被分为称为公共特征的5个部分
- 这些公共特征包括了关键实践（KP），即每一个KPA包括5类KP
- 实现了这些KP后，就实现了关键过程域的目标



### 关键过程域 (KPA, Key Process Area)

- KPA是一系列相互关联的操作活动
- KPA是某一级别的一组目标，用以衡量是否具有此级别的能力。
- 每个KPA的目标总结了它的关键实践（KP），目标说明了每一个KPA的界限、范围、内容和关键实践
- 不同级别的KPA（项目、数目、内容）是不同的，但其中很多项都有深层次的联系（上级是下级的深化和延伸）
- 18个关键过程域，分布在2~5级

过程等级	管理方面	组织方面	工程方面
优化级		技术改革管理 过程变更管理	缺陷防范
可管理级	定量过程管理		软件质量管理
已定义级	集成软件管理 组间协调	组织过程焦点 组织过程定义 培训程序	软件产品工程 同级评审
可重复级	需求管理 软件项目计划 软件项目跟踪与监控 软件转包合同管理 软件质量保证 软件配置管理		
初始级	无序过程	每一级成熟度都由若干 关键过程域组成，共18个	

每个关键过程域（KPA）都与一些目标相关，代表某种对过程的要求

KPA 标明了某级成熟度所要求的条件（也就是企业需要努力达成的目标）和评估标准  
KPA 说明了要达到此级成熟标准所需解决的具体要点



级别2的关键过程域（6个）  
主要涉及项目管理方面的内容

需求管理(RM)	对分配需求进行管理。作出详尽的需求，并指出若需求变动时，对时间及金钱的追加
软件项目计划(SPP)	制定实施软件工程与管理软件项目的合理的计划。 <b>重点：可操作性</b>
软件项目跟踪和监控(SPTO)	按照软件项目计划对软件完成情况和结果进行跟踪和评审，并在必要时作一些纠正
软件转包合同管理(SSM)	选择高质量的软件分包商，并进行有效的管理，是对分包商的一种约束
软件质量保证(SQA)	对软件项目和软件产品质量进行监督和控制，提供了适度的可见性，表现为黑盒之间的断点
软件配置管理(SCM)	保证软件项目生成的产品在软件生命周期中的完整性



级别3的关键过程域（7个）  
主要涉及项目和机构的问题

组织过程焦点(OPF)	为改进机构的整体软件过程能力，建立负责软件过程活动的机制
组织过程定义(OPD)	开发和维护机构标准软件过程及相关资源
培训程序(TP)	提高个人的技能和知识，更有效、更好的完成工作
集成软件管理(ISM)	将软件工程和管理活动结合成为密切相关、定义完整的软件过程
软件产品工程(SPE)	严格定义及过程控制
组间协调(IC)	内部协调为主（如定期小组会议），外部协调为辅
同级评审(PR)	同一级别的其他软件人员对该软件项目产品系统评审的方法，以便尽早、有效的排除缺陷



级别4的关键过程域（2个）  
主要是定量检测，属于技术活动

定量过程管理(QPM)	以过程为中心进行管理， 定量的控制软件项目的过程效能
软件质量管理(SQM)	以产品为中心进行管理， 定量的评价软件产品的质量， 并实现具体的质量目标



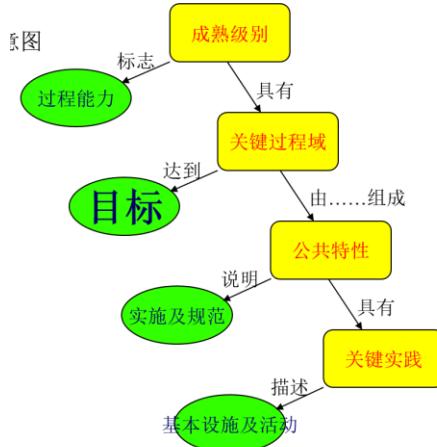
级别5的关键过程域（3个）  
主要解决可控制问题，进行问题预防

缺陷防范(DP)	明确产生缺陷的原因并预防它们再次发生
技术改革管理(TCM)	确定新技术（如工具、方法和过程），并有序地将这些技术引入机构内
过程变更管理(PCM)	不断改进机构中所使用的软件过程， 提高软件质量和生产率，缩短生命周期



### 关键过程域的目标

- 表示关键过程域的范围、边界和意图
- 目标概述了某KPA中的关键实践
- 检验关键过程域是否满足的指标



### 关键过程域的公共特性

- 是一些属性，指明一个KPA是否有效、可重复和可持续
- 用来组织关键实践
- 五个部分

- 执行约定：为保证过程得以建立和持续发挥作用必须采取的行动
- 执行能力：前提条件
- 实施活动：必须执行的任务和步骤
- 度量和分析：用以确定、改进和控制过程的状态
- 验证实施：验证实施活动与确立的过程是否遵循已制定的步骤

公共特性		含义
AC	实施活动	做什么？
AB	执行能力	有能力做吗？
CO	执行约定	怎样做？
VE	验证实施	做了没有？
ME	度量分析	做得怎样？



### 关键过程域的关键实践 (key practices)

- 描述了对关键过程域的有效实施和制度化起最重要作用的基础设施和活动
- 是为了达到一个KPA的目标而要做的事

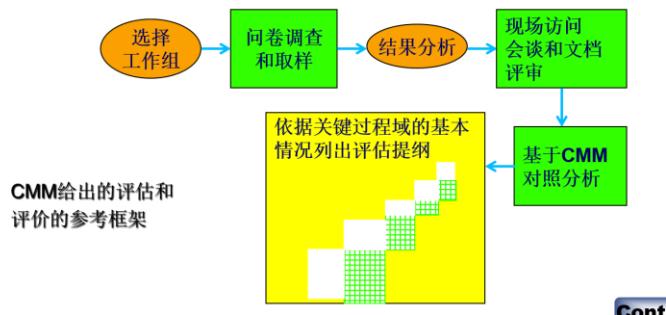


### 运用CMM

- 软件过程评估

用来判断一个组织当前的软件过程的能力状态，判断一个组织所面对的更高层次上的与软件过程相关的课题，以及利用组织的鼎力支持来对该组织的软件过程进行有效的改进

-  针对软件组织自身内部软件改进
- 软件能力评价
  -  用来判断有意承担某个软件项目的软件组织的软件过程能力，或已进行的软件过程所处状态是否正确或正常
  -  针对接受评价者
- 软件过程评估和软件能力评价方法



- 软件过程评估和软件能力评价之间的差异

### 2.3 CMM各级之间的关系



#### 软件企业自身现状与CMM级别

- 从不成熟到成熟：时间过程，不提倡跨成熟度级别的进化
- 企业成熟度提高的表现
  - 项目的计划目标和实际结果的差距减小
  - 项目的计划目标与实际结果的变化减小
  - 计划目标结果的提高
- 努力地提高软件过程，其结果将是质量和产量的显著提高



#### 从初始级向可重复级过渡

- 初始级是CMM的起点
- 无序、混乱的过程中开发超出预算和时间进度的产品
  - 过程成熟度的焦点问题不是技术问题，而是软件开发过程中的管理问题
  - 需求管理和需求变动的应对
- 经验教训文档化 → 过程可重复



#### 从可重复级向已定义级过渡

- 可重复级的需求管理
- 等级2定义了管理的基本过程，而没有定义具体的执行步骤标准
- 等级3制定企业范围的工程化标准
  - 定义明确的过程应包括：
    - 妥当的依据、输入、完成工作的标准和步骤，
    - 审核的方法、输出和完成的数据
    - 建立在项目管理的基础上
    - 整个软件过程的定义、集成和文档化



#### 向已管理级和优化级过渡

- 等级3：软件人员工作时有一定的自由度
  - 规范问题
  - 产品检验问题
- 等级4：过程的定量控制
- 等级5：不断改进  
通过过程执行的反馈信息  
实现形式：逐渐改进现有的过程、不断创新技术与方法

## 2. 4 CMM实施的人员及组织机构



### 人员的构成

- 经理 (manager)
- 各级经理
- 工作人员



### 组织机构的划分

- 组织 (Organization)：一个公司或其他实体内的一个单位
- 项目 (Project)：一项要求共同努力的任务，产品可包括硬件、软件和其他成分
- 组 (Group)：负责一组任务或活动的部门、经理和个人的集合，组的规模可变。



### 组织机构的划分

- 主要的软件工作组
  - 软件工程组 (Software Engineering Group)
  - 软件相关组 (Software Related Group)
  - 软件工程过程组 (Software Engineering Process Group)
  - 系统工程组 (System Engineering Group)
  - 系统测试组 (System Test Group)
  - 软件质量保证组 (Software Quality Assurance Group)
  - 软件配置管理组 (Software Configuration Management Group)
  - 培训组 (Training Group)

## 2. 5 CMM初始级

- 未加定义的一种随意过程
- 无序、混乱的软件过程
- 特点：能力指的是个人的行为特征而不是机构特征
- 结果：外在表现是不能保证后续产品具有前一产品的质量
- “手工作坊式的生产方式”

## 第三章 可重复级（第2级）



### 可重复级的基本特征

- 优势

已建立了项目管理的方针和规定  
对项目已设置基本的软件管理控制  
组织的过程能力体现在有纪律  
当有转包商时，通过转包合同建立有效的供求关系

- 缺陷：依赖经验管理项目



### 达到的目标

- 软件质量保证活动独立于软件开发
- 按照严格的步骤估计软件项目时间、成本和进行软件项目开发管理评审
- 有一种机制控制需求和代码的变更以及高层管理机构对软件开发项目状态进行正规的评审



### 关键过程域

- 需求管理
- 软件项目计划
- 软件项目跟踪和监控
- 软件转包合同管理
- 软件质量保证
- 软件配置管理

## 3.1 需求管理



### TRM公司软件项目发现的BUG分布结果：

- 在软件整个生命周期发生的错误中，45%是在需求分析和设计阶段产生的。



### 需求管理容易出问题的原因何在？

- 缺乏良好的需求规格说明编写模板
- 较严重地忽略了非功能性需求
- 缺乏对需求文档的配置管理
- 需求规格说明缺乏可测性
- 缺乏较好的需求规格说明转化规范

### 什么是需求？



### Rational 的定义：

- 系统必须符合的条件或具备的功能。



### Merlin Dorfman 和 Richard H. Thayer 的定义：

- 用户解决某一问题或达到某一目标所需的软件功能。系统或系统构件为了满足合同、规约、标准或其他正式实行的文档而必须满足或具备的软件功能。



需求不总是显而易见的，而且它可来自各个方面



需求并不总是容易用文字明白无误地表达。



存在不同种类的需求，其详细程度各不相同。



如果不加以控制，需求的数量将难以管理。



需求相互之间以及与流程的其他可交付工件之间以多种方式相关联。



需求既非同等重要，处理的难度也不同。



需求涉及众多相关利益责任方，这意味着需求要由跨职能的各组人员来管理。



需求发生变更。需求可能对时间敏感。



需求管理 (RM, Requirements Management)

■ 对分配给软件的系统需求进行管理

- 非技术性需求（主要指合同条款）
- 技术性需求（分为功能需求和非功能需求）
- 接收标准

- 在客户和实现客户需求的软件项目之间达成共识
- 控制系统软件需求，为软件工程和管理建立基准线
- 保持软件计划、产品和活动与系统软件的一致性

参与者：系统工程组和软件工程组

- 分配需求是制定软件开发计划的根据，是整个软件生命周期中估算、计划、执行和跟踪软件项目活动的基础。
- 当分配需求改变时，必须调整受影响的软件计划、工作产品和活动，使其与更新后的需求保持一致。

### 3.1.1 需求管理的目标



需求管理的主要工作

- 在客户和将处理客户需求的软件项目之间建立对客户需求的共同理解。
- 和顾客一起建立和维护“分配需求”的协议。



目标

- 确保分配需求是受控的，建立供软件工程和管理使用的基线。
- 软件计划、产品和活动与分配给软件的系统需求保持一致。

### 3.1.2 需求管理的执行约定



项目遵循一个书面的、组织上的方针去管理分配给软件的系统需求。

- 为了在开发过程中有章可循，要为分配需求建立文档
- 为了使分配需求切实可行，必须由软件经理和其他受影响组成员进行审查
- 当分配需求变更时，为了保持一致，软件计划、工作产品和活动也要随之更改

### 3.1.3 需求管理的执行能力

- 对每个项目建立分析系统需求并将其分配到硬件、软件和其它系统成分的职责。
- 对分配需求建立文档
- 提供足够的用以管理分配需求的资源和投资
- 软件工程组和其它有关组的成员受到培训以便完成他们的需求管理活动。

### 3.1.4 需求管理需实施的活动



包括三个方面的内容

- 需求定义的管理：在分配需求被纳入软件项目前，由软件工程组对分配需求进行评审
- 需求实现的管理：软件工程组采用分配需求作为制定软件计划、开发产品和开展管理活动的基础
- 需求变更的管理：评审分配需求变更，并将变更加入软件项目计划书中

### 3.1.5 需求管理的实施过程



软件开发过程各阶段的需求管理

- 可用一个数据库来标志需求在各个阶段的状态，来明确分配需求管理活动的状态



需求管理涉及三个方面的内容

- 需求定义的管理、需求实现的管理、需求变更的管理

#### 1) 需求定义的管理



**需求定义**，也被称为需求开发。对需求定义的管理包括以下几方面：

- 需求获取
- 需求分析
- 需求处理
- 需求验证



**需求获取阶段的工作：**

确定系统的用户需求（建立系统模型）：

确定用户的业务目标

分析业务的环境与内容（全面认识整个业务领域）

展开软件需求调查

如果原有的业务流程或组织架构等本身存在问题，需进行业务重组与改造业务。

需求获取Step1：建立一张项目视图

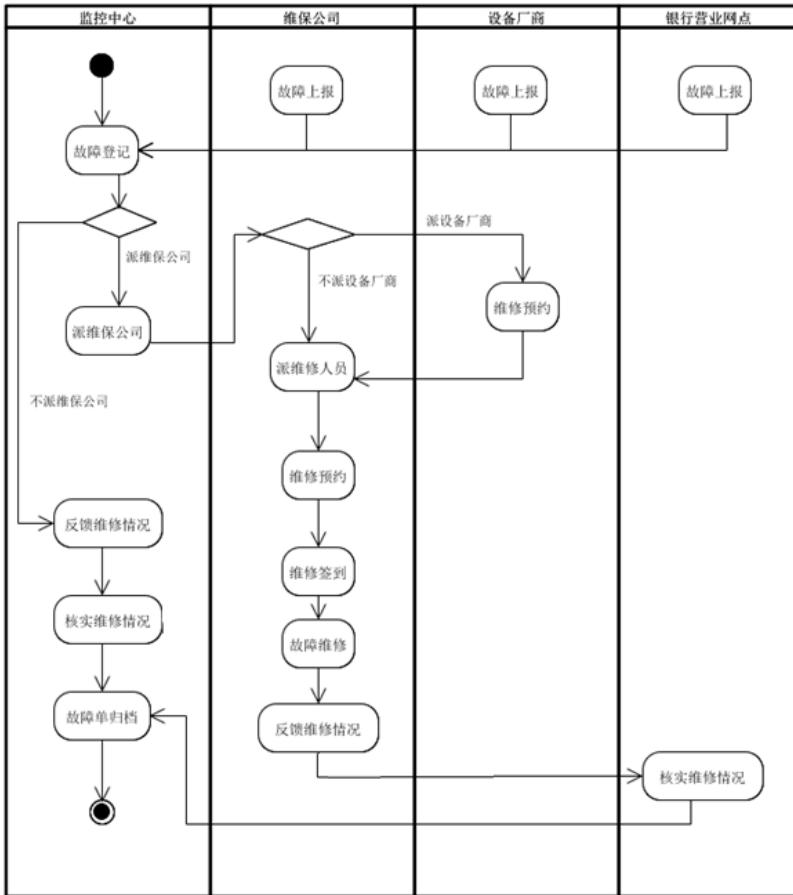
		1	2	3	4	5	6
A	业务需求	背景	项目 机遇	项目 目标	市场 需求	客户 价值	项目 风险
B	方案描述	功能 视图	主要 特征	假设 依赖			
C	范围局限	首次发 行范围	随后发 行范围	局限性 专用性			
D	系统环境	用户 概貌	项目 优先级				
E	成功因素						

需求获取Step2：建立业务模型

首先完成以下分析工作，在此基础上建立业务模型。

- **领域中的业务角色（Business Role/Actor）**
- **角色间的业务功能等关系**
- **业务组织架构（Business Organization）**
- **业务规则（Business Rule）**
- **业务实体（Business Entity）**
- **业务事件（Business Event）**
- **以业务角色为主角的业务用例（Business Use Case）**

需求获取Step3：分析工作流，产生信息流



#### 需求获取Step4：分析业务实体

- **业务实体**就是组织活动的对象，是业务的有效实体。  
例如：库存系统中的存货、订单等。
- 实体的逐步细化、具体属性化，就成为系统数据字典的雏形。  
例如：存货的属性有：名称、种类、数量、单价、入库日期、等。

#### 需求获取阶段项目经理的关注点

- 技术经理，重点控制上述几个需求获取过程。
- 项目经理，重点关注需求范围（项目视图），而不是系统功能的细节。其中，特别要关注非功能软件需求，主要包括：
  - 质量需求、环境需求、设计约束
  - 开发策略、问题信息、沿用信息

这几个问题也是需求，或者是往往容易被忽略的需求的外延，控制需求范围（外延）是项目经理的责任。



#### 需求分析阶段的工作：

确定系统的功能需求（建立功能模型）

功能模型，可以映射出软件产品核心的需求，包括：

- 与业务功能对应的组织的特性，
- 与业务流程对应的内外部关系特性等；

系统用例模型，是在已经建立的业务模型的基础上建立的系统模型。

软件产品本身可能还存在与业务无直接关系的另类需求，一般与硬件、软件环境相关等，通常两类需求构成软件需求的总集。

#### 需求分析Step1：建立系统用例

- 系统用例模型，引入了角色、用例和边界的概念
  - 角色是业务之外与业务交互的人或事 ----> 系统外部

- 用例是业务模型中，业务的活动 ----> 系统内部
- 用例间及角色间的关系表达
  - 泛化关系：由带封闭箭头的实线表示，由子级指向父级。
  - 关联关系：用实线表示，由参与者指向用例
  - 包含关系：用带开放箭头的虚线和版型表示，由基用例指向子用例。
  - 扩展关系：用带开放箭头的虚线和版型表示，由子用例指向基用例。
- 用例模型描述事件流，  
包括主事件流、其他事件流、前提条件、事后条件等。

#### 需求分析Step2：描述对象关系

描述系统中的实体对象及其关系，建立静态模型。

- 定义系统中的实体类
- 表示实体类之间的联系

#### 需求分析 Step3：描述对象间关系及行为

- 用 Interaction 图描述对象关系
- 用带泳道的活动图描述用户跟系统间的交互行为：



#### 需求处理阶段的工作：

产生系统的需求规格说明书

- 阐述软件系统必须提供的功能和性能，以及他们必须考虑的限制条件。
- 是测试、用户使用和维护文档的基础
- 是子系统规划、设计和编码的基础。

**应尽可能详细地描述系统外部化的（向用户展示的）行为，也要为项目组内部，尽可能详细地讲明系统功能上的考虑。**

#### 需求规格说明书模版

- 引言
  - 目的、文档约定、产品范围、参考文献
- 概述
  - 产品前景、产品功能、用户特征、运行环境、设计和实现上的限制、假设和依赖条件
- 外部接口
  - 用户界面，软硬件接口、通信接口
- 系统特性
  - 说明和优先级、响应序列、功能需求
- 其他非功能需求
  - 性能需求、安全性需求、质量属性，业务规范、用户文档
- 其他
  - 词汇表、分析模型、待确定问题清单

#### 需求处理阶段项目经理关注的主要问题

- 需求来源
  - 项目经理要让所有的责任者和风险承担者明白这份需求各内容的来源。  
来源（要求）被责任人确认是接受了的。
- 需求形式化
  - 应该把需求分解为层和项  
需求数据库的支持
- 需求跟踪矩阵
  - 实现对需求实现的确认、需求变更影响的分析评估等



### 需求验证阶段的成果

- 编写测试计划与测试用例
- 编写用户使用手册
- 编写系统验收标准
- 通过需求评审

需求验证阶段：

确定需求评审的对象

- **非技术需求**（例如：协议、条件和/或合同条款）。具体实例有：要交付的产品、交付日期、里程碑。
- **技术需求**

实例有：功能需求；性能需求；设计约束条件；程序设计语言；界面要求。

用于确认软件产品是否能满足给定需求的**验收标准**。

需求验证阶段：

实施需求评审：

- 确定不完整和遗漏的给定需求；
- 评审给定需求以确定他们是否：可行、适用于软件实现、说明清楚、适当、彼此一致、可测试。
- 有负责分析和分配系统需求的小组对确认可能有问题的给定需求进行评审并进行必要的修改。
- 相关小组协商由给定需求所得出的约定。

良好的需求规格说明的特点：

- 不含糊性
- 完整性
- 可检验性
- 一致性
- 可跟踪性
- 可使用性

## 2) 需求实现的管理



### 需求实现管理的重要工作

- 需求的形式化与需求基线的建立
- 需求状态的变化
- 需求状态变化的追踪



需求的形式化和需求基线的建立， 建立软件架构——建立需求基线的前提

- 以构架为中心的软件开发模式
- 软件构架与系统的软件需求往往不是一一对应的关系，通常软件构架的设计应当基于业务领域模型。
- 真正健壮的软件构架应当面向整个业务领域



需求实现：

设计软件体系结构

软件架构，确定系统整体结构、层次划分，不同部分之间的协作等

根据关键的功能需求、质量需求、环境需求， 综合主要的设计约束和开发策略，来定义系统的总体构架。

**需求的形式化**

- 形式化的目的  
首先通过记录使需求固定下来，减少口头的误传。

解决完整性和无歧义性。

### ■ 定义需求属性 —> 需求属性化

方便地对需求的变化做出完善的记录。

评估需求的状态

建立需求数据库，分层次进行需求分解（需求分配）

需求属性		含义	说明
名称	*	需求名称	用最简洁的语言表示需求的核心含义
描述与定义	*	对需求的描述和定义	需求的本质内容、可以用模型、图、表表示
编号(层/序)	*	需求的顺序号	可根据系统结构或任务的WBS编排
来源	*	需求的提出来源	用户需求的更高层依据、来源
提出/决策人		需求的提出人	当需求变化或受到影响时最易于讨论和决定的人
优先级		需求的优先级	表明高、中、低，以备需要取舍或先后响应时
实体	*	需求实现的实体	表明需求与实现实体的对应关系
状态	*	需求所处的状态	包括：提出、批准、实施、实现、完成或拒绝、推迟、等待、丢弃等。
稳定性		需求的稳定性	按稳定性定义描述稳定性的高、中、低
验收标准		验收标准	需求实现的验证方式和标准
负责人		需求实现的负责人	
备注		对需求的附加说明	
作者		需求的提交者	
版本号	*	需求的版本号	
变更记录	*	本版本的变更内容	描述本版本的变更原因、内容、影响
更新日期	*	需求的变更日期	

### 建立需求基线

- 基线是指在软件开发过程中的里程碑，这些里程碑的标志是一项或多项经过正式的技术评审并一致认同的软件制品的提交。
- 项目开发过程的制品经过正式评审并被相关人员一致同意，对基线的修改必须要通过正式的变更控制流程。

记录需求状态的变化：需求受各种因素的影响，会产生不可预料的变化。

状态	定义
被建议	根据需求来源，责任、相关人提出了需求。
被拒绝	在一系列需求开发过程后，该需求没有被认可。
被批准	在需求（特别是变更需求）被分析，评估了合理、可行、成本、影响等要素，被确认可接受，被标注了新的版本号、给出了新的标号等需求属性、被加入到需求基线库中，进入实现过程。
被实现	已实现设计、编码、单元测试。
被验证	根据验收标准，已经通过集成以上的测试，被验证实现了需求的要求，被放置进配置基线库。表明需求已经被实现。
被丢弃	被批准的需求已从基线库中被丢弃。记录下丢弃的原因和决定责任人。
被交付	通过用户的验收测试，需求以交付物的形式，向用户提交。

### 追踪需求状态的变化

通过跟踪定义了的需求，掌握需求在实现过程中的具体实现细节与目标的距离，建立需求追踪链。

### 建立需求追踪矩阵

功能需求	用例			
	增加用户	修改用户	终止用户	注销用户
用系统属性定义用户属性	✓	✓		
用户属性可批量修改		✓	✓	✓
用户属性检查范围可采用剔除法		✓	✓	
用户属性统计可按指定方法		✓		

用例	功能需求	设计元素	实现代码段	测试用例
Ucase 0021	查询	Check 0021类	Check 0021( )	CTest 0021. 1
				CTest 0021. 2
Ucase 0022	插入	Insert 0031类	Insert 0031( )	Itest 0031. 1
				Itest 0031. 2

需求实现阶段项目经理关注的主要问题

- 需求要尽量形式化、数据库化；
- 在需求形式化的基础之上，建立需求追踪矩阵，对需求实现追溯和回溯；
- 需求追踪能力的好坏，是软件项目管理水平的标志。

### 3) 需求变更的管理

需求变更的常见原因：

- 因竞争、成本等因素，工期已经确定且极不合理
- 用户在需求期提不出需求、或用户的需求不明确
- 项目组对业务不熟悉、或者没有与用户密切结合、需求分析工作不细致
- 项目组没有很好地实施需求管理



6大需求变更控制活动

- 确定需求变更控制过程
- 建立需求变更控制委员会
- 进行需求变更影响分析
- 跟踪所有受需求变更影响的工作产品
- 调整需求基线
- 维护需求变更记录和文档



评估需求稳定性

- 统计基线需求的数量
- 统计每月/每周基线需求项的增加、删除、修改的数量
- 对需求不稳定的因素和来源进行分析，做出的评估

对需求稳定性的关注，是软件项目经理经常需要做的工作。

需求不稳定，是项目组的“危险警报”。

轻则是项目组工作量的增大，

重则可能导致项目的严重延误，甚至失败。

### 3.1.6 需求管理



#### 度量和分析

进行测量，并将测量结果用于对分配需求的管理活动状态的确定。

##### ■ 度量内容：

- 每个分配需求的状态、
- 分配需求的变更情况、
- 分配需求的变更次数



#### 验证实施

- 高级管理者参与定期评审那些管理分配需求的活动
- 项目经理既定期地又事件驱动地参与评审那些管理分配需求的活动
- 软件质量保证组审查管理分配需求的活动和工作产品，并报告其结果。

## 3.2 软件项目计划



#### 软件项目计划的作用？

- 建立估计值，即建立和维护项目计划因素的估计值。
- 开发项目计划文档，即文档化项目计划，维护项目计划，并以此作为项目管理的基线。
- 获得并维持所有项目干系人对项目的承诺。



#### 软件项目计划 (SPP, Software Project Planning)

- 估计待完成的工作，建立必要的缩写，确定进行该工作的计划。
- 软件项目计划提供完成和管理软件项目活动的基础，并按照软件项目的资源、约束和能力，阐述对软件项目的顾客作的约定。



#### 目的

为执行软件工程和管理软件项目制定合理的计划。



#### 软件项目计划的步骤

- 估计产品规模和所需的资源
- 制定时间表
- 鉴别和评估软件风险
- 协商约定



#### 目标

- 对计划和跟踪软件项目用的软件估计已建立档案
- 软件项目的活动和约定是有计划的并已建立档案
- 受影响的组和个人同意他们的关于软件项目的约定



### 执行约定

- 指定软件项目经理负责协商约定和制定项目软件开发计划
- 项目遵守书面的、组织的用于策划软件项目的方针。包括:
  - 将分配的软件需求作为计划软件项目的基础
  - 对软件项目约定的协商程序
  - 对估计的软件规模、工作量、成本、进度和其它约定的评审程序
  - 对项目的软件开发计划进行管理和控制



### 执行能力

- 对软件项目有文档化的且经批准的工作陈述
- 项目经理直接的或委托代表，协调项目软件计划
- 为制定软件项目计划提供足够的资源和投资
- 对参与软件计划的人员进行软件估计和计划规程培训



### 实施的活动

- 软件项目的策划、建议与评审
- 识别或确定易于管理的软件生命周期，即软件过程模型
- 按照文档化的规程，制定项目的软件开发计划
- 识别软件工作产品
- 按照文档化的规程进行软件工作产品的规模、软件项目的工作量、成本和关键计算机资源的估计，并建立文档，进行评审，使得到承认。
- 按照文档化的规程编制软件进度表
- 鉴别和估计与项目的成本、资源、进度和技术方面相联系的软件风险，建立文档。
- 制定项目软件工程设施和支持工具的计划
- 记录软件计划数据



### 度量和分析

- 对实际情况进行测量，将测量结果用于确定软件计划活动的状态。
  - 与计划相比，软件项目计划的里程碑的完成情况
  - 与计划相比，软件项目计划活动完成的工作、使用的工作量、消耗的资金等。

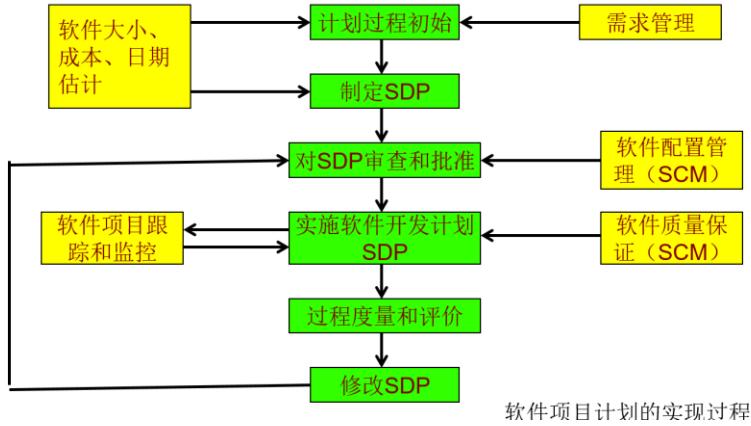


### 验证实施

- 高级管理者定期参加评审软件项目计划的活动
- 项目经理定期地和有事件驱动地参与评审软件项目计划的活动
- 软件质量保证组审查软件项目计划活动和工作产品，并报告其结果。



### 实现过程



### 制定软件项目计划的技巧

- 从粗粒度的计划开始
- 实施者应该是计划人员
- 不要忘记“不该忘记的事”
- 将任何设想和约束编入文档
- 认识到不同的资源意味着不同的计划
- 创建现实的计划
- 只规划有价值的事
- 适当使用项目管理工具

### 3.3 软件项目跟踪和监控



### 软件项目跟踪和监控 (SPTO, Software Project Tracking and Oversight)

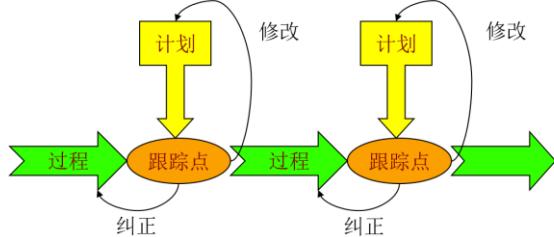
对照已文档化的估计、约定和计划，评审和跟踪软件完成情况和结果，基于实际的完成情况和结果调整这些计划。

管理者在软件项目完成时或里程碑出监控软件活动，将实际的软件规模、工作量、成本和时间表与计划相比较，确定进展情况。



### 目的

- 建立对实际进展的适当的可见性，使管理者能够在软件项目性能明显偏离软件计划时采取有效措施。



在软件开发过程的若干关键点上进行软件项目跟踪和监控



### 目标

- 对照软件计划，跟踪实际结果和性能
- 当实际结果和性能明显偏离软件计划时，采取纠正措施并加以管理直到结束。

- 对软件约定的更改应得到受影响组合个人的认可。



### 执行约定

- 指派软件项目经理，对项目的软件活动和结果负责。
- 遵循书面的组织用于管理软件项目的方针。包括：
  - 采用并维护已文档化的软件开发计划作为跟踪的基础
  - 随时向项目经理汇报软件项目的状态和问题
  - 软件计划未实现时，采取纠正措施，调整计划



### 执行能力

- 具有被批准且已文档化的软件项目开发计划
- 软件项目经理明确地分配关于软件工作产品和活动的任务
- 为跟踪软件项目提供足够的资源和投资
- 对软件项目经理进行项目管理和技术方面的必要培训



### 实施的活动

- 利用SDP跟踪活动，需要时按照已文档化的规程修订项目的开发计划
- 跟踪实际的开发过程，必要时采取纠正措施
  - 软件产品的规模、工作量、成本
  - 关键计算机资源；项目进度
  - 软件工程技术活动
  - 与成本、资源、进度和技术相关的风险
- 记录软件项目的实际度量数据，并重新计划数据
- 审查：定期的内部审查和项目里程碑处审查  
项目跟踪过程                           项目计划过程

由项目计划得出跟踪需求，并通过  
管理升级的标准

收集当前实际值并与计划值比较

评价与计划相关项目的性能与状态

报告状态和调整或建议

需要调整吗？

是  
调整项目目标、计划或资源（包括约定控制过程）

否  
文档化，并发布已修订的计划

项目结束？

否  
是  
结束

项目跟踪过程流程图



### 度量和分析

- 对实际情况进行测量，将测量结果用于确定软件项目跟踪和监控活动状态。

- 执行跟踪和监控活动中花费的工作量和其他资源
- 软件开发计划更改活动



#### 验证实施

- 上级部门定期审查
- 项目经理定期或有事件驱动时审查
- 软件质量保证组审查和核算软件跟踪和监控的活动和工作产品，并报告结果



#### 跟踪与监控的六大要素

- |       |         |
|-------|---------|
| ■ 要素一 | 进度      |
| ■ 要素二 | 成本      |
| ■ 要素三 | 风险      |
| ■ 要素四 | 规模      |
| ■ 要素五 | 关键计算机资源 |
| ■ 要素六 | 工作量     |



#### 如何进行项目的跟踪与监控

- Step1 估算、制定计划
- Step2 确定项目跟踪与监控策略
- Step3 在关键里程碑节点进行跟踪
- Step4 分析跟踪数据
- Step5 采取行动

### 3.4 软件转包合同管理



#### 软件转包合同管理 (SSM, Software Subcontract Management)

- 选择合格的软件转包商、与转包商建立承诺、跟踪和审查转包商执行合同的结果。
- 目的 选择合适的软件子承包商并有效地管理他们



#### 目标

- 主承包商选择合适的软件子承包商
- 主承包商和软件子承包商认同他们相互的约定
- 主承包商和软件子承包商保持不断通信
- 主承包商对照约定定期跟踪软件子承包商的实际结果和性能



#### 执行约定

- 为软件转包合同的管理制定书面的组织方针
- 指定一名转包合同经理负责建立和管理软件转包合同



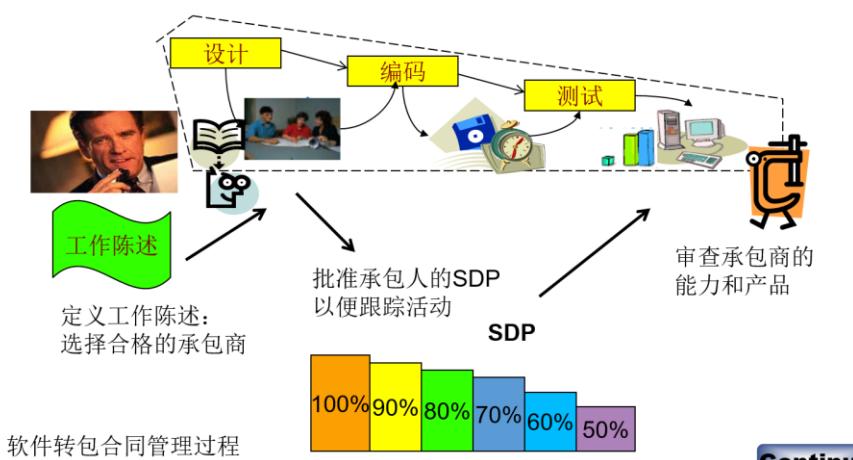
#### 执行能力

- 为选择子承包商和管理转包合同提供足够的资源和投资
- 培训涉及建立和管理软件转包合同的软件经理和其他人员
- 参与管理软件转包合同的软件经理和其他人员接受技术方面的定向培训



## 实施的活动

- 选择合格的转包商，并与之签订合同
  - 根据文档化的规程，定义并计划将被转包的工作
  - 基于对转包合同竞标者完成工作能力的评估，选择转包商
  - 管理转包合同的基础是主承包商和软件转包商之间的**合同协议**
- 主承包商审查转包商的软件开发计划，并用于跟踪其软件活动
- 评审、评价转包商
  - 主承包商的管理部门同软件转包商的管理部门一起进行定期的**状态或协调审查**
  - 双方一起进行定期的**技术审查**和交流
  - 根据文档化的规程，在选定的**里程碑**处进行正式评审，评价转包商的软件工程的完成情况
  - 定期评估软件转包商的成绩，并与转包商一起审查此评估
- 监督、验收转包商的软件活动
  - 主承包商的**软件质量保证组**根据文档化的规程监控软件转包商的**质量保证活动**
  - 主承包商的**软件配置管理组**根据文档户的规程监控转包商的**软件配置管理活动**
  - 作为转包商软件产品交付过程的一部分，主承包商根据文档化的规程指导接收测试
  - 定期评价软件转包商的性能，并对该评价活动进行评审



## 度量和分析

- 对实际情况进行测量，将测量结果用于确定软件转包合同管理活动状态。
  - 转包合同的成本与计划相比较
  - 转包产品的实际交付日期与计划相比较
  - 主承包商的交付产品交付给转包商的实际日期与计划先比较



## 验证实施

- 上级部门定期审查
- 项目经理定期或有事件驱动时审查
- 软件质量保证组审查和核算软件转包合同管理的活动和工作产品，并报告结果

## 3.5 软件质量保证



## 软件的六大品质要素

- 正确性 (correctness)：
  - 实现的功能达到设计规范，并满足用户需求的程度
- 可靠性 (reliability)：

在规定的时间和条件下，仍能维持其性能水准的程度

■ 易用性 (usability) :

用户掌握软件操作所要付出的时间及努力程度

■ 效率 (efficiency) :

软件执行某项功能所需电脑资源（含时间）的有效程度

■ 可维护性 (maintainability) :

当环境改变或软件发生错误时，执行修改或恢复所做努力的程度

■ 可移植性 (portability) :

从一个电脑系统或环境移到另一电脑系统或环境的容易程度

即参照一定的质量标准、目标及各项软件流程、规范来监督，管理公司产品的质量；

在许多质量体系还不是很成熟的公司，维护和发展这些质量标准、流程规范等也是由质量保证人员进行。

行内有个这样的说法：“软件质量保证并不能够保证软件的质量”，

事实也是如此，软件质量的好坏不是一个人，一个部门能够决定的。

但是，我们可以把提高软件的质量作为我们从事软件质量保证工作的目标。



### 软件质量保证 (Software Quality Assurance, 简称SQA)

即参照一定的质量标准、目标及各项软件流程、规范来监督，管理公司产品的质量；在许多质量体系还不是很成熟的公司，维护和发展这些质量标准、流程规范等也是由质量保证人员进行。行内有个这样的说法：“软件质量保证并不能够保证软件的质量”，事实也是如此，软件质量的好坏不是一个人，一个部门能够决定的。但是，我们可以把提高软件的质量作为我们从事软件质量保证工作的目标。



### 软件测试 (Software Test)

尽早、尽可能多地发现软件系统中存在的缺陷及问题。但是，好的软件并不是“测”出来的，而是“做”出来的，所以，每一个测试人员都应该清楚这样一点：任何人都不可能找出软件中隐含的所有缺陷和问题，正如世界上没有人是十全十美的。

■ SQA重点是对软件开发过程进行监督、管理、控制；

■ SQC重点是对软件开发的成果进行检查、控制。

■ 软件质量保证并不能够保证软件的质量

- QA: Quality assurance 质量保证

- QC: Quality Control 质量控制

- SQA: 软件质量保证

- SQC: 软件质量控制

	<b>QA</b>	<b>QC</b>
<b>全称</b>	<b>Quality Assurance</b>	<b>Quality Control</b>
<b>定义</b>	为确保软件开发过程和结果符合预期的结果，依照过程和计划采取的一系列活动及其结果评价	为发现软件产品的错误而进行工作的过程
<b>目标</b>	减少并纠正实际的软件开发过程和软件开发结果与预期的软件开发过程和软件开发结果的不符合情况	寻找错误，并尽可能的为修复错误提供更多信息
<b>职责</b>	监控公司质量保证体系的运行状况，审计项目实际执行情况与公司规范之间的差异。出具改进建议和统计分析报告，是过程和产品质量的审计者。	对每一个阶段或关键点的产出物进行检测，评估产出物是否符合预计的质量要求，是产品质量检查者。

阶段	<b>QA</b>		<b>QC</b>	
	工作内容	工作产品	工作内容	工作产品
项目建设	定义产品质量指标 参与项目规划评审	产品质量指标 <b>QA工作报告</b>		
项目计划	编制 <b>QA计划</b> 过程审计	项目 <b>QA计划</b> <b>QA工作报告</b>	测试管理计划	测试计划
需求分析	需求评审 过程审计	<b>QA工作报告</b>	分析测试需求	测试需求
设计	设计评审 测试设计评审 过程审计	<b>QA工作报告</b>	设计测试用例	测试用例
编码	代码评审 过程审计	<b>QA工作报告</b>	单元测试 集成测试	<b>Bug记录</b> 测试报告
测试	用户手册验证 过程审计	用户手册验证报告 <b>QA工作报告</b>	集成、系统、性能、 回归测试 测试工作管理	测试脚本 <b>Bug记录</b> 测试报告
实施	产品质量状态评估 过程审计	产品质量评估报告 <b>QA工作报告</b>	内部接收测试 验收测试	<b>Bug记录</b> 测试报告

- SQA 负责软件开发流程的质量，是软件质量保证人员
- SQC 负责软件开发过程中各个阶段产出成果的质量，是软件测试人员
- SQA 与 SQC 的工作都是为了保证软件的质量，但是，
  - SQA 是通过控制过程来保证软件产品的质量
  - SQC 是通过控制每个阶段的结果来保证软件产品的质量
  - 只有 SQA 没有 SQC，无法保证最终产品的质量
  - 只有 SQC 没有 SQA，无法保证软件项目成功
- 有效地保证软件产品质量 SQA 与 SQC 缺一不可，二者必须相互配合，在过程和结果都正确的基础上才能有效改善软件产品的质量。



软件质量保证 (SQA, Software Quality Assurance)

- 评审和审计软件产品和活动以验证它们符合适用的规程和标准，给项目和其它有关的 经理提供这些评审和审计的结果。
- 目的 向管理者提供对软件项目所有的过程和正被开发的产品的适当的监控



软件质量保证组

- 项目初期参与软件项目计划、标准和规程的制定
- 整个生命周期中审查项目活动和审核软件工作产品

- 向管理者提供可视性



#### 目标

- 软件质量保证活动是有计划的
- 软件产品和活动遵守适用的标准、规程，需求的情况得到客观的验证
- 受影响的组和个人接到软件质量保证活动结果的通知
- 高级管理者处理在软件项目内部不能解决的不符合问题



#### 执行约定

- 项目遵循书面的实施SQA的组织方针，包括：
  - 落实SQA到各个适当阶段
  - SQA有直接向上级汇报的渠道
  - 上级部门定期审查SQA的活动和结果



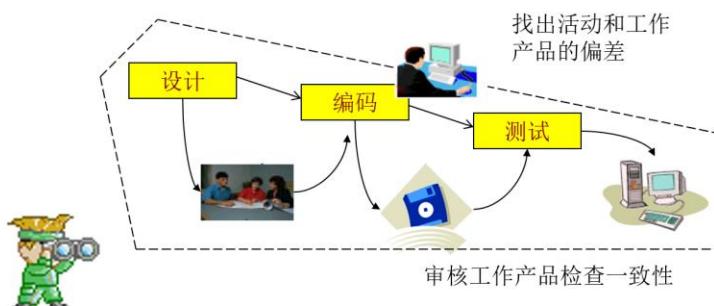
#### 执行能力

- 存在SQA小组
- 为执行SQA活动提供足够的资金和资源
- 培训SQA组的成员
- 项目组成员对SQA组工作的认同



#### 实施的主要活动

- 根据文档化的规程制定软件项目的SQA计划
- SQA工作组按照SQA计划来开展活动
- SQA组参与项目软件开发计划、标准和规程的制定和审核
- SQA组评审软件工程活动，以检验一致性
- SQA组审核制定的软件工作产品，以检验一致性
- SQA工作组定期向软件工程组报告其活动结果
- 根据文档化的规程对在软件活动和软件工作产品中所找出的偏差建立文档
- 在合适的时候，SQA组和客户的SQA人员一起对SQA活动和调查结果进行定期审查



软件质量管理贯穿产品生产的全过程



#### 度量和分析

- 对实际情况进行测量，将测量结果用于确定软件质量保证活动状态。
  - 参照计划比较SQA活动里程碑的完成情况；
  - 参照计划比较SQA活动完成的工作、工作量和资金的使用情况；
  - 参照计划比较产品审核和活动审查的次数 等



## 验证实施

- 上级部门定期审查SQA的活动
- 项目经理定期又有事件驱动地审查SQA的活动
- 独立于SQA组的专家定期审查SQA组的活动和软件工作产品

## 3.6 软件配置管理



### 为什么要进行软件配置管理?

忽视软件配置管理可能导致混乱现象

- 发错了版本
- 安装后不工作
- 异地不能正常工作
- 已经解决的缺陷过后又出现错误
- 开发人员把产品拿出去出售赢利
- 找不到最新修改了的源程序
- 找不到编程序的人



### 软件配置管理能帮助我们解决什么问题?

- 多人同时修改程序或文档
- 人员流动
- 软件维护中的历史重现
- 控制软件的复杂性
- 影响项目进度的特殊因素
- 已修复的错误仍然存在
- 协同开发中的工作重复



### 软件配置管理的管理对象:

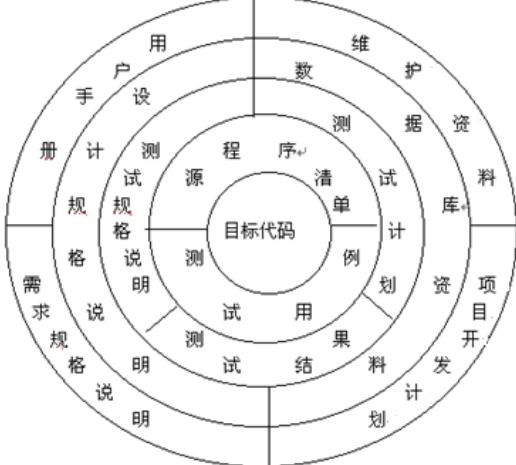
软件配置项 (SCI, Software Configuration Item )



### 软件配置项是软件生命

周期内产生的各种工作

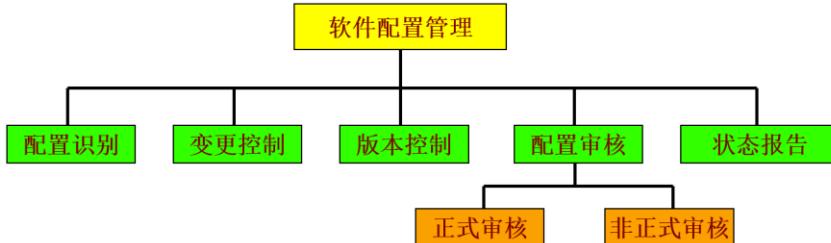
产品。包括：程序、文档、数据、软件工具库标准和规约





## 软件配置管理 (SCM, Software Configuration Management)

- 标识在给定时间点上软件的配置、系统的控制对配置的更改，以及维护在整个软件生命周期中配置的完整性和可跟踪性。
- 是在开发和维护的各个阶段管理软件演进过程的方法和规程。
- 目的 建立和维护在项目的整个软件生命周期中软件项目产品的完整性



任务一：配置标识：要配置标识，首先必须明确项目生命周期内所要产生的工作产品，然后确定工作产品的名称和标识规则。总体原则是，保证配置管理工具检索便利，让项目组成员容易记住标识规则，同时要确保组织一级的标识规则的一致性。

任务二：版本管理：版本管理一般是使用工具来完成的，如 Rational ClearCase、Merant PVCS Version Manager、Microsoft Visual SourceSafe 等。使用这些工具时，容易被忽视的一点是制定所使用工具的版本规则。如果直接采用工具的内部版本号，会给产品发布带来一些困难。通常采用“X.Y.Z”方式进行版本标识，明确 X、Y 和 Z 各位数字递增的规则，然后结合工具标签（Label）功能，便可实现高效的版本管理。

任务三：变更管理：变更管理是项目管理的一个重点和难点，涉及的范围很广。实施高效的变更管理至少应该包括两个部分：“定义合理的变更管理流程”、“采用自动化工具作为支持”。在具体的实践中，应该对变更进行分类和分层，建立起处理不同变更的“变更控制委员会”（CCB），既保证项目组成员有一定的自主权，又不会耽误高层经理对关键问题的把握。

任务四：配置审核：配置审核包括两方面的内容：“配置管理活动审核”、“基线审核”。

“配置管理活动审核”用于确保项目组成员的所有配置管理活动，遵循已批准的软件配置管理方针和规程，如检入（Check in）/检出（Check Out）的频度、工作产品成熟度提升原则等。实施“基线审核”，要保证基线化软件工作产品的完整性和一致性，并且满足其功能要求。基线的完整性可从以下几个方面考虑：基线库是否包括所有计划纳入的配置项？基线库中配置项自身的内容是否完整？（如，文档中所提到的参考或引用是否存在？）此外，对于代码，要根据代码清单检查是否所有源文件都已存在于基线库。同时，还要编译所有的源文件，检查是否可产生最终产品。一致性主要考察需求与设计以及设计与代码的一致关系，尤其在有变更发生时，要检查所有受影响的部分是否都做了相应的变更。审核发现的不符合项要进行记录，并跟踪直到解决。

在实际操作过程中，一般认为审核是一种事后活动，很容易被忽视。但是“事后”也是有相对性的，在项目初期审核发现的问题，对项目后期工作总是有指导和参考价值的。为了提高审核的效果，应该充分准备好检查单，如表 1 所示。

任务五：报告配置状态：报告配置状态的目的，是向项目所有成员提供基线内容和状态、基线变更信息（如表 2 所示），这也是实现资源共享的前提。此外，在项目生命周期中进行对配置项的变更数据统计分析，有利于评估项目风险，有效控制项目的执行。在变更请求被批准、基线版本发生变化及项目组提出任何需要时，可以采用 Email 等方式进行报告。

任务六：发布管理：实施了规范的配置管理，发布就显得很从容了。但是必须要注意的是：发布的产品应该是从软件基线库中提取出来的；在软件发布给最终用户之前，要准备发布记录，为软件产品分配发布版本号，同时要对它进行发布评审并确认其得到批准。一般来说，高层经理、项目经理、软件质量保证人员和测试组都应该参加发布评审。



## 目标

- 软件配置管理活动是有计划的

- 所选定的软件工作产品是已标识的、受控的和适用的
- 对已标识的软件工作产品的更改是受控的
- 受影响的组和个人得到软件基线的状态和内容的通知



#### 执行约定

- 明确地分配每个项目的SCM责任
- 在项目的整个生命周期中实施SCM
- SCM为外部交付的软件产品、指定的内部软件工作产品、指定的用于项目内部使用的支持工具都实施SCM
- 项目建立或使用一个仓库（如数据库）用于存放配置项/单元和相关的SCM记录
- 定期审核软件基线和SCM活动



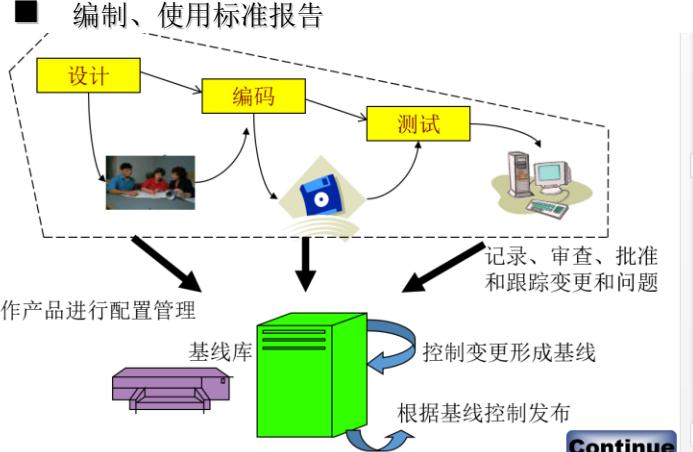
#### 执行能力

- 存在或建立一个有权利管理项目软件基线的委员会，即软件配置控制委员会（SCCB）
- 存在一个负责协调和实施项目的软件配置管理组（SCM组）
- 为执行SCM活动提供足够的资源和投资
- 对SCM组的成员进行SCM活动的对象、规程和方法方面的培训
- 对软件工程组和其他软件相关组的成员进行培训，以便执行SCM活动



#### 实施的主要活动

- 制定SCM计划
  - 每个软件项目制定SCM计划
  - 将已文档化且经批准的SCM计划作为执行SCM活动的基础
- 标识、更改、维护配置
  - 建立一个配置管理库系统作为存放软件基线的仓库
  - 标识置于配置管理之下软件工作产品
  - 根据文档化的规程，提出、记录、审查、批准和跟踪所有配置项/单元的更改要求和问题报告
  - 根据文档化的规程记录配置项/单元的状态
- 更改、控制、审核软件基线
  - 根据文档化的规程控制基线的更改
  - 根据文档化的规程生成由软件基线库制造的产品，并控制它们的发行
  - 根据文档化的规程进行软件基线审核
- 编制、使用标准报告



#### 度量与分析

- 对实际情况进行测量，将测量结果用于确定软件配置管理活动状态。

单位时间内处理更改申请的次数;  
在SCM计划中，SCM活动重要事件的完成情况;  
在SCM活动过程中完成的工作、花费的工作量和资金 等



### 验证实施

- 上级定期审核
- 项目经理定期或有事件驱动时审核
- SCM组定期审核软件基线以检验其是否与定义它们的文档相符
- 软件质量保证组审查和审核SCM的活动和工作产品



### 软件配置管理的观念要求 ■

- (1) 软件配置管理不只是收集程序代码。 ■
- (2) 有效的软件配置管理应是有利于设计部门和设计人员提高工作效率。 ■
- (3) 软件配置管理应以过程清楚、可跟踪为度，尽量简化操作程序，方便操作。



### 配置管理的主要任务

- 制定项目的配置管理计划
- 对配置项进行标识
- 版本管理
- 配置支持
- 变更管理
- 构造管理
- 过程支持和团队支持
- 状态报告和审计控制



### 配置管理应该明确的问题

- 你识别出的配置项有哪些？
- 配置管理的主要任务是什么？
- 你熟悉的配置管理工具有哪些？
  - 1、阅读课文 P.93—101，学习：
  - 软件配置管理的概念
  - 软件配置管理的五个公共特性



### 常用配置管理工具

- 元老：CCC, SCCS, RCS
- 中坚：Rational Clear Case
- 新秀：Hansky Firefly
- 开源奇葩：CVS, SVN
- 小工作组级：Merant PVCS
- 入门级：VSS



### 配置管理中应注意的问题

- 评估开发团队当前配置管理状况
- 定义实施的范围

- 计划资源要素
- 像你的老板一样思考



## 配置管理的误区

- 版本控制=软件配置管理
- 编码水平最差=配置管理员
- 采用配置管理工具=有效的配置管理

### 1、版本管理

#### 1.1 软件配置项(software configuration item):

含义：在软件生存周期内所产生的各种应纳入管理范围的系统构成成分。

包括各种管理文档和技术文档，源程序与目标代码，以及运行所需的各种数据等（配置管理的资源对象）。

形态：在通常的软件配置管理系统中，最基本的软件配置项是以磁盘文件的形式进行存放和管理的。

#### 1.2 版本管理是配置管理的基础：

应当记录每个软件配置项的所有历史记录，并记录该软件配置项由何人创建，何人在何时因何原因进行了修改等信息，以及对这些软件配置项版本的进行的检索和信息查询等活动。

#### 1.3 版本树：

可以对软件系统的不同演化方向进行管理。

1.4 软件配置项的版本管理 记录一棵带有时间标记的配置项版本演化的树结构信息。

### 2、配置支持

#### 2.1 软件配置(software configuration): 所有软件配置项在不同时期的组合、结构与关系定义。

2.2 系统建模：通过定义配置来表示整个系统或其中的子系统。

2.3 依赖性追踪：例如：查找与某个源文件版本对应的设计文档的版本。

2.4 影响分析：分析对系统一个部分的修改可能影响哪些其它部分。

### 3、变化管理

#### 3.1 变化：软件版本演化的来源与过程

来源：需求变化、增加功能、修改错误 .....

生命周期：请求、审批、实施、验证、审核、结束。

3.2 变化控制：记录和控制对软件配置项的每一次修改。

3.3 变化跟踪：一个变化生命周期进行到哪一步了？已经改掉的 bug 又出现了，怎样找出原因。

3.4 变化传播：帮助将对产品一个版本的修改传播到其它版本中。

### 4、构造管理（Build）

4.1 系统的构造和重新构造（Build）：帮助开发人员正确和快速地构造和重新构造产品的任何版本。

4.2 软件发布管理（Release）：为不同的用户提供不同的版本，避免其中发生混乱。

4.3 软件部署管理（Deployment）：帮助在分布式环境中部署整个系统。

### 5、过程支持

#### 5.1 过程控制

5.2 预定义的过程模版 和 可剪裁的过程实例

### 6、团队支持

6.1 工作区管理：不同的开发人员拥有独立的不相互影响的工作空间。

6.2 并行开发：支持多个开发人员同时开发一个项目。

6.3 远程开发：开发人员在物理上可以分布在相距较远的位置上。

## 7、状态报告

依赖性报告 影响报告 构造报告 变化状态报告 差异报告 历史报告 访问控制报告 冲突检测报告

## 8、审计控制

8.1 验证软件配置管理过程

8.2 验证系统管理的所有配置项的完整性

8.3 基本的审计控制是记录配置管理过程中执行的所有活动，并提供检索机制——日志

### 可重复级总结1



可重复级的中心在于项目而非组织

需求管理(RM)	明确需求、同意需求
软件项目计划(SPP)	文档化估算/计划/承诺
软件项目跟踪与监控(SPTO)	管理计划、实施纠正措施
软件转包合同管理(SSM)	选择/管理有资格的承包商
软件质量保证(SQA)	提供管理的可见性
软件配置管理(SCM)	维持产品的完整性

关键过程域	实施的障碍
需求管理(RM)	<ul style="list-style-type: none"><li>需求变更</li><li>未经正式批准的需求变更</li><li>客户总是不愿把需求文档化</li></ul>
软件项目计划(SPP)	<ul style="list-style-type: none"><li>作计划花费时间并且要承担义务</li><li>估算需要经验和数据</li><li>存在不切实际的要求</li></ul>
软件项目跟踪与监控(SPTO)	<ul style="list-style-type: none"><li>计划或许是不现实的</li><li>管理者未经充分的培训</li><li>工作太忙，没有时间管理</li><li>发生返工</li></ul>

关键过程域	实施的障碍
软件转包合同管理(SSM)	<ul style="list-style-type: none"><li>发包商与承包商的差异：语言、文化、情报</li><li>缺乏沟通</li></ul>
软件质量保证(SQA)	<ul style="list-style-type: none"><li>怀疑SQA的价值</li><li>软件工程师缺乏对SQA的尊重</li><li>SQA本身不合格</li></ul>
软件配置管理(SCM)	<ul style="list-style-type: none"><li>进度压力</li><li>自动化的支持不足</li><li>有人认为不需要SCM</li></ul>



项目具有基于组织方针的被定义好的过程



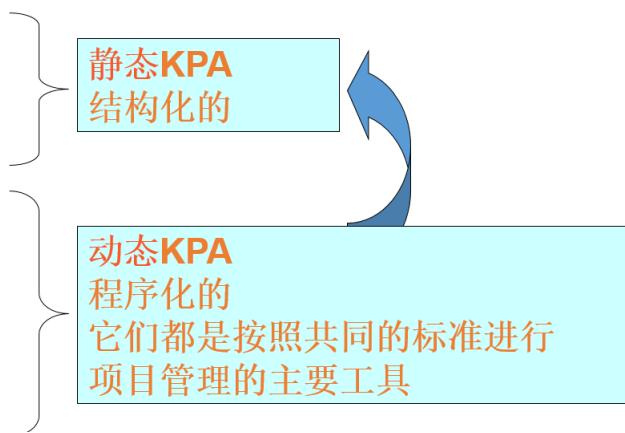
### “已定义”

- 用于开发、维护软件的过程已经得到了系统的阐述并能付诸于实施，包括：
    - 组织标准软件工程过程
    - 组织标准软件管理过程
  - 定义的组织标准过程集，适用于整个组织。
  - 项目管理更加一致，策划与预测更为准确，资源的协调更加流畅。
- 系统的阐述：
- 准备妥当的判据
  - 输入
  - 完成工作的标准和步骤
  - 审核的方法
  - 输出
  - 完成的判据



### 等级3的七个关键过程域

- 组织过程焦点
- 组织过程定义
- 培训程序
- 集成软件管理
- 软件产品工程
- 组间协调
- 同级评审



### 4.1 组织过程焦点



### 组织过程焦点 (organization process focus, OPS)

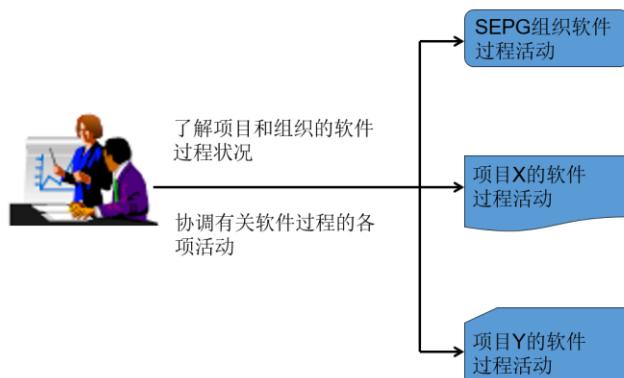
- 目的
  - 建立组织对软件过程活动的责任
  - 为组织的整体软件过程能力的提高提供组织上的保证
- 目标
  - 在整个组织内，有关软件过程的活动是协调的
  - 识别出一个具体软件过程与一个过程标准相比较的强处和弱处
  - 在组织层上，有关软件过程的活动是有计划的



### 组织过程焦点的基础

- 执行约定
  - 遵循一个书面的方针
  - 高级管理者主持组织的软件过程制定和改进
  - 高级管理者监督组织的软件过程制定和改进活动
- 执行能力
  - 存在一个负责组织软件活动的组

为开展软件过程活动提供足够的资金和资源  
负责组织的软件过程活动的组员接受有关培训  
软件工程组和其他工程组的成员接受有关软件过程活动及相关角色的定向培训  
组织过程焦点通常由软件工程过程组（SEPG）负责。



#### 组织过程焦点的活动

- 定期评估软件过程并制定相应的更改计划
- 组织制定和维护有关软件过程制定和改进活动的计划
- 协调组织的标准软件过程和项目定义软件过程的制定和改进活动
- 协调组织的软件过程数据库的使用
- 新过程、新方法、新工具的评价、监控和推广
- 对有关组织和项目的软件过程的培训工作加强统一管理
- 及时将有关软件过程制定和改进的活动通知到与实施软件过程有关的组和人员



#### 组织过程焦点的评价

- 度量和分析
  - 包括：已经完成的工作、花费的工作量以及消耗的资金与计划的比较；每次软件过程的评估结果与以往的评估结果和建议的比较
- 验证实施
  - 组织过程焦点的验证由高级管理者承担
  - 验证的一般内容

## 4.2 组织过程定义



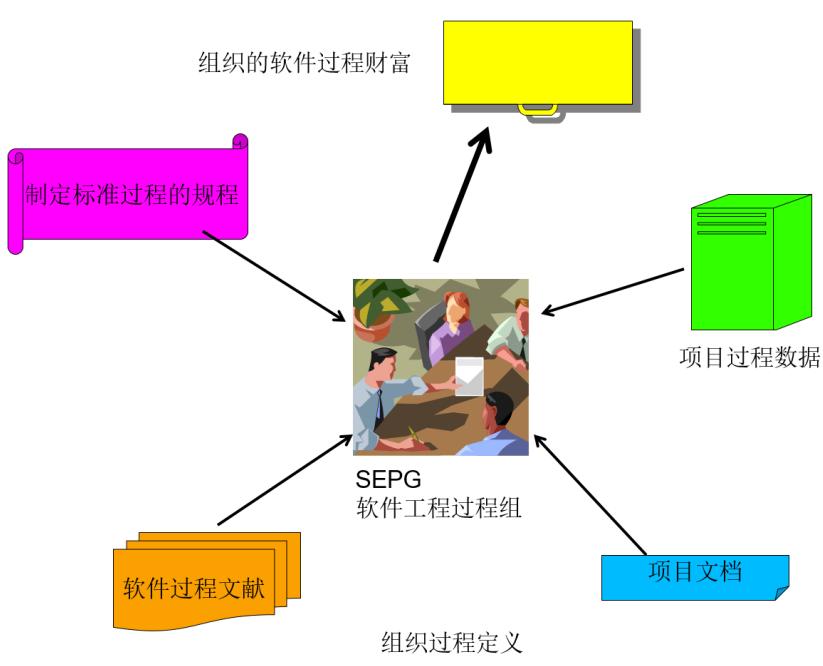
#### 组织过程定义 (Organization Process Definition, OPD)

- 负责软件过程活动的组在组织层上定义软件过程
  - 包括
    - 制定和维护组织的标准软件过程
      - 软件过程元素的描述
      - 软件过程体系结构的描述
    - 制定和维护相关的过程财富
    - 目的 开发和维护一组便于使用的软件过程财富



#### 组织过程定义的基础

- 执行约定 遵循书面的方针
- 执行能力
  - 提供足够的资源和资金
  - 必要的培训：软件工程实践和方法、过程分析和建立文档的方法以及过程建模等



#### 组织过程定义的活动

- 按照已文档化的规程制定和维护组织标准软件过程
- 为组织的标准软件过程建立文档
- 对经批准供使用的软件生命周期，建立文档并进行维护
- 制定和维护项目裁减组织标准软件过程的指南和准则
- 建立和维护组织的软件过程数据库
- 建立并维护与软件过程有关的文档库



#### 组织过程定义的评价

- 度量和分析  
包括：过程开发和维护进度表中一个重要阶段的状态、过程定义活动的成本等
- 验证实施  
由软件质量保证组（SQA）承担  
验证内容包括：在开发、文档化和维护组织的标准软件过程和有关财富时遵循适当的标准；组织的标准软件过程和有关财富被控制和恰当的使用

#### \*\*组织过程焦点与组织过程定义

##### •SEPG的核心工作

组织过程焦点	组织过程定义
创建和维护组织的软件过程改进计划（SPI）	创建和维护组织的软件过程财富



#### 组织过程财富

- 组织的标准软件过程
- 组织的软件生命周期

- 裁剪标准软件过程的指南和准则
- 软件过程数据库
- 软件过程文档库
- 组织的培训工作

KPA	对象	内容
组织 过程 焦点	SEPG	制定SPI计划，使用CMM，过程管理
	联络员	使用SPI计划，与SEPG协同工作， CMM定向培训
	员工	SPI计划定向培训，CMM定向培训
组织 过程 定义	SEPG	定义SSPS，定义生命周期，
	联络员	定义裁剪指南
	员工	使用SSPS，使用生命周期，使用裁剪指南， 使用测量数据库，使用SSPS库

### 4.3 培训程序



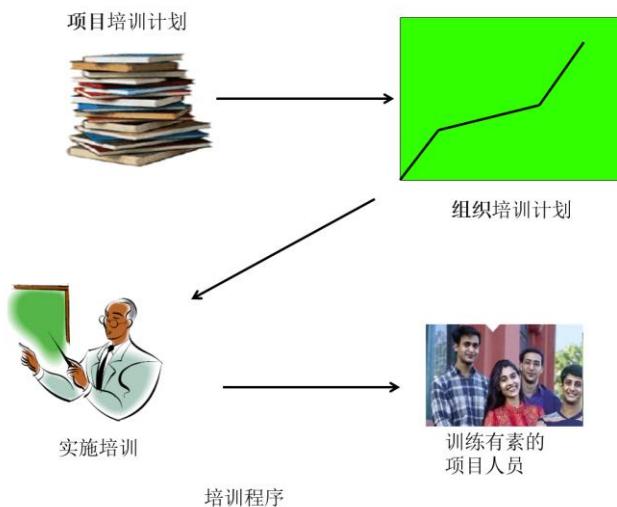
培训程序 (Training Program, TP)

- 培训程序步骤
  - 确定所需培训
  - 开发培训内容
- 培训途径：正式培训 VS. 非正式培训
- 目的：提高软件开发者和软件管理者的知识技能



培训程序的基础

- 执行约定 遵循书面的方针：
  - 需要掌握的技能和知识、获得培训的途径；提供培训；
  - 选择组织内部培训或者组织外部培训
- 执行能力
  - 建立一个责任实现组织培训需求的小组
  - 为实施培训提供足够的资源和资金
  - 培训组成员具有完成其培训活动所需的技能和知识
  - 软件经理接受有关培训程序的定向培训





### 培训程序的活动

- 每个软件项目制定和维护满足其培训需求的培训计划
- 按照文档化的规程制定和修订组织的培训计划
- 按照组织培训计划实施组织培训
- 根据组织标准开发和维护组织层上培训课程
- 制定所需培训的免修规程，并据此确定哪些员工已具备了做好本职工作所应有的技能和知识
- 维护培训记录



### 培训程序的评价

- 度量和分析
  - 确定培训程序活动的状态和质量
  - 度量培训程序活动状态
  - 度量培训程序质量
- 验证实施
  - 高级管理者定期参与评审培训程序的活动
  - 定期地、独立地评价培训程序是否与组织需要相一致、相关联
  - 评审、审计培训程序的活动及其工作产品并报告审核结果

## 4.4 集成软件管理



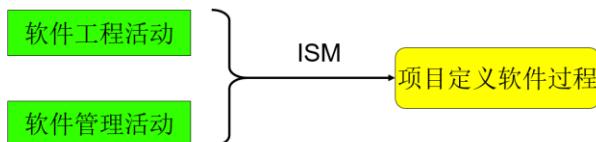
### 集成软件管理 (Integrated Software Management, ISM)

- 目的：协调软件项目的工程活动和管理活动

集成的含义：

集成软件管理 (ISM, Integrated Software Management)

把软件的工程和管理活动集中到持续的和确定的软件过程中来，它主要包括组织的标准软件过程和与这相关的操作，这些在组织过程定义中已有描述。当然，这种组织方式与该项目的商业环境和具体的技术需求有关。



目标：基于组织标准软件过程定义项目定义软件过程，并据此对项目进行计划和管理。



### 集成软件管理的基础

- 执行约定
  - 制定一个组织方针来计划和管理软件项目
- 执行能力
  - 为项目定义软件过程的制定和使用足够的资源和资金
  - 负责指定项目定义软件过程的人员接受必须的培训
  - 对软件经理的培训



### 集成软件管理的活动（管理内容）

- 按照文档化的规程裁剪组织标准软件过程、制定项目定义软件过程
- 按照建档的规定进行项目定义软件过程的修订工作
- 按照文档化的规程制定、修订项目软件开发计划

- 按照项目定义软件过程管理软件项目
- 利用组织软件过程数据库计划和估计数据
- 按照已建档的规程管理软件工作产品的规模（或工作变更的规模）
- 按照已建档的规程管理项目软件工作量和成本
- 按照已建档的规程管理项目的关键计算机资源
- 按照已建档的规程管理项目软件进度中的关键依赖关系和关键路径
- 按照已建档的规程确定、评价、建档和管理项目的软件风险
- 定期审核软件项目并确定相应的行动，使软件项目的性能和结果与经营的客户和最终用户的需求相一致



#### 集成的软件管理的评价

- 度量和分析 进行度量并确定集成软件管理活动的效果
- 验证实施
  - 高级管理者定期参与审查软件项目的管理活动
  - 项目经理定期的和需要时审查软件项目的管理活动
  - 软件质量保证组评审、审计软件项目的管理活动及其工作产品、并通报结果

### 4.5 软件产品工程



#### 软件产品工程 (Software Product Engineering, SPE)

- 目的 协调一致地执行一个妥善定义的工程过程，并有效地生产正确的、一致的软件产品
- 任务 采用项目定义软件过程及适当的方法和工具实施软件的工程任务



#### 软件产品工程的基础

- 执行约定 遵循文档化的、有关软件工程活动的组织方针
- 执行能力
  - 为实施软件产品工程任务足够的资源和资金
  - 为软件工程技术人员提供与项目及应用领域有关的内容培训
  - 为软件工程技术人员提供有关软件工程科目的定向培训
  - 为项目经理和所有软件经理提供项目技术方面的定向培训



#### 软件产品工程的活动

- 将适当的软件工程方法和工具集成到项目定义软件过程中去
- 按照项目定义软件过程制定基于分配需求的软件需求，并对其维护、验证和建档
- 按照项目定义软件过程开发、维护和审查软件设计以适应软件需求，形成编码框架
- 按照项目定义软件过程开发、维护、建档并验证软件代码，以实现软件需求和设计
- 按照项目定义软件过程进行软件测试
- 按照项目定义软件过程计划和实施软件的集成测试
- 设计并实施软件的系统测试和确认测试
- 按照项目定义软件过程编写并维护将要用于运行和维护软件的文档
- 按照项目定义软件过程收集并分析统计评审和测试中发现的缺陷及其有关数据
- 维护软件工作产品间的一致性



#### 软件产品工程的评价

- 度量和分析
  - 进行度量并判断软件产品的功能和质量

进行度量并判断软件产品工程活动的状态

## ■ 验证实施

高级管理者定期参与审查软件产品工程活动

项目经理定期的和需要时审查软件产品工程活动

软件质量保证组评审、审计软件产品工程活动及其工作产品、并通报结果

## 4.6 组间协调



组间协调 (Intergroup Coordination)

- 对于一个软件项目来说，一般要设置若干工程组
- 任务

软件工程组与其他工程组一起参与阐述系统级的需求、目标和问题  
计划和管理组间协作的技术界面和交互行为

组间协调 (Intergroup Coordination) 描述了各个工程组之间在进行协同工作时应当执行的关键实践。

组间协调的目的是建立一种软件工程组与其他工程组一起积极参与项目协作的方式，通过这种方式的协作使得项目能够有效并高效地满足客户的需求。

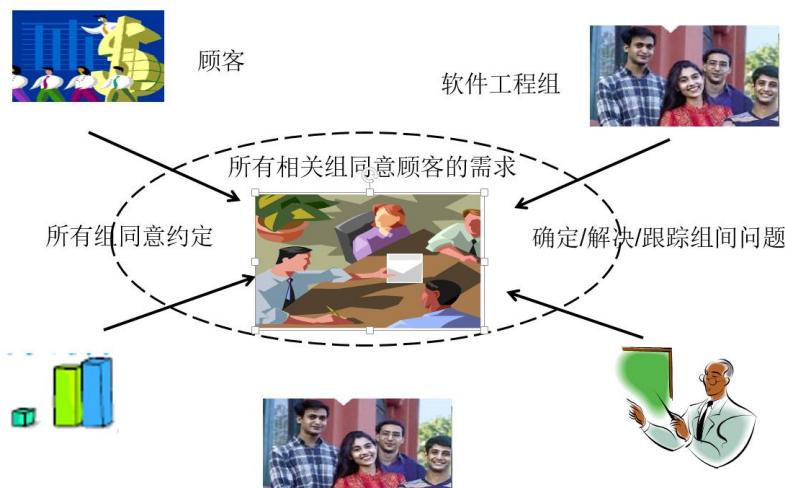
在组间协调过程中强调要实现三个目标，

- 一是客户的需求要得到全部受影响的组的认同；
- 二是工程组之间的约定要得到受影响的组的认同；
- 三是工程组要识别、跟踪和解决组间问题。

从项目启动开始，项目组内的各工程组之间及项目内工程组与项目外部有关工程组间就可能存在需要进行协调的关系，组间协调活动贯穿在整个项目生命周期之中。

东软在实施组间协调过程时，根据自己的组织结构和项目类型的特点，将组间协调过程分成三个阶段进行：

1. 在需求建立时，受影响的软件工程组和其他工程组一起参与确认和阐述系统层的需求、对象，识别、分析问题。必要时，顾客和最终用户也要参与建立系统层的需求和目标的工作，这些需求和目标将成为项目中全部工程活动的基础。
2. 在项目的策划阶段，对组间的技术工作接口和相互的关键依赖关系进行识别和策划，以保证整个系统的质量和统一性。
3. 在项目执行过程中，各工程组的代表参与定期的技术评审和内部交流，以保证所有工程组都清楚各组的状态和计划，各组代表共同管理和评审组间协调活动，保证系统和组间的问题受到恰当的关注，并协商解决组间协调中存在的问题。



相关组：使用者、测试人员、SQA组



### 组间协调的基础

- 执行约定 文档化的、关于建立跨学科工程组的组织方针
- 执行能力
  - 提供足够的资源和资金
  - 各工程组使用的支持工具是相互兼容
  - 组织内所有经理接受关于团队合作的培训
  - 每一个工程组的所有任务领导接受有关其他工程组所使用的过程、方法及标准等方面培训
  - 工程组成员接受有关团队合作方面内容的定向培训



### 组间协调的活动

- 软件工程组和其他工程组（合适时包括客户和最终用户）一起参与确定系统需求
- 项目软件工程组的代表与其他工程组代表一起监督、协调技术活动，解决技术问题
- 按照文档化的计划，交流组间约定，协调、跟踪所进行的工作
- 按照文档化的规程确定、协商和跟踪工程组之间的关键依赖关系
- 由接受组评审工作产品，以保证工作产品满足他们的要求
- 按照文档化的规程处理由项目工程组代表无法解决的组间问题
- 项目工程组代表定期开展技术审查和交流工作



### 度量与分析

- 用于确定组间协调活动的状态



### 验证实施

- 高级管理者定期组间协调活动
- 项目经理定期的和需要时审查组间协调活动
- 软件质量保证组评审、审计组间协调活动及其工作产品、并通报审查结果

## 4.7 同级评审



### 同级评审 (Peer Reviews)

- 由于软件工作产品生产者处于同一级别的人员系统的检测软件工作产品，找出其中错误并确定需要更改的领域

是指处于同一级别其他软件人员对该软件项目产品系统地检测的一种手段，其目的是为了能够较早和有效地发现软件产品中存在的错误并改正它们。它是一种在软件产品工程中非常重要的和有效的工程方法。



### 同级评审的基础

- 执行约定
  - 项目遵循一个文档的、实施同级评审的组织方针
- 执行能力
  - 提供足够的资源和资金
  - 同级评审领导者接受必要的关于如何领导同级评审的培训
  - 同级评审参与者接受必要的关于同级评审目标、原理和方法的培训



### 同级评审的活动

- 制定文档化的同级评审计划
- 按照文档化的规程执行同级评审
- 记录同级评审的行为和结果



## 同级评审的评价

### ■ 度量与分析

包括：完成的同级评审数及计划数；同级评审所花的工作量及计划工作量；被评审的工作产品数与计划数

### ■ 验证实施

软件质量保证组评审、审计同级评审的活动及其工作产品，并报告结果



## 同级评审经常遇到的问题

- 评审时组间争论过多或过少
- 缺乏心理训练
- 竞争与合作意识不充分

## 考虑不全面

### ◆ 评审时组间争论过多或过少

这一问题在不同企业的表现也不同。调查表明，争论较多的情况是工作产品的输入/输出不清楚，组间缺乏沟通的公共平台，因此组间只有通过较多的讨论甚至争论才能弄清其他组的需求。遗憾的是，事后大家并没有坐下来认真讨论如何改进原工作产品的模板形式或表现形式，因而也就无法从根本上解决问题。另一种较极端的情况是，评审一个组的工作产品时，其他组很少发表意见，尽管有些问题是十分明显的。通过调研发现，这实际上是企业文化的问题。一种普遍的想法是“等我们实际做的时候自然就清楚了”，但实际情况往往事与愿违，这使企业的工作效率大打折扣，但又不易被管理层意识到。无论是哪种情况，最终的原因是：“项目甚至企业缺乏持续改进过程管理的意识。”

### ◆ 缺乏心理训练

做好同行评审的最大挑战是克服心理障碍。简单地说，同行评审就是被别人挑错或挑别人的错。因此，评审会就像是答辩会，必须做好充分的准备。当角色互换，自己成了挑错方时，则应该把被评审的工作产品看成是自己在较早前完成的，现在再做一次修改，且修改完成后，自己要拿它去参加评审答辩。经历几次这种心理角色换位，就会逐渐适应。如果大家都这么做，同行评审就会形成良好的氛围，这对形成健康的企业文化将起促进作用。

### ◆ 竞争与合作意识不充分

从另一个角度看，同行评审又是竞争与合作的最佳表现场所和形式，凡在这种场合讲话有理且意见中肯的人逐渐会成为团队的核心人物。在这种竞争的环境中，合作是基础，同行评审的目的就是在合作的前提下尽早且有效地排除工作产品中的缺陷。把握好竞争与合作的尺度，将有益于企业文化的发展，否则有可能出现恶性循环。如何把握呢？从大量的案例看，多数消化少数是较好的方法，因为文化是不可创造的。

### ◆ 考虑不全面

同行评审存在的另一个问题是评审时仅注意工作产品内容本身，大家面对面地弄清内容后，却忽略了如何改进工作产品的表现形式，使新表现形式下的工作产品可更好地用书面形式表示，进而可减少面对面沟通的需求。当然，面对面的沟通并不是不好，但如果一个工作产品需要太多的口头表达才能被理解，则原因只有两个：书写不清楚或模板定义不好，如果是后者则情况更糟。



## 同级评审的陷阱

- 陷阱 1：参与评审的人不了解评审过程
- 陷阱 2：评审过程没有被遵循
- 陷阱 3：适合的人没有参加评审
- 陷阱 4：评审会陷入到对问题的解决上
- 陷阱 5：评审的焦点放在文档形式而不是内容本身上
- 同级评审实践常常会落入一些陷阱而导致失败，例如：缺乏明确过程定义的非正式评审由于容易引起误解而无法达到跟踪。当评审组长不能有效地发挥其作用时，许多这样的问题就会发生。如何识别和避免这些陷阱？本文将介绍最常见的 5 个陷阱及其对策。

### 陷阱 1：参与评审的人不了解评审过程

这种陷阱的一个症状是开发组成员不能使用准确的、一致的词汇来描述各种类型的同级评审，或者各评审小组之间不能遵循一致的评审过程。因此培训、实践和文档化的评审过程是必需的。所有潜在的评审人员都必须理解什么是评审，为什么、怎样、何时和谁来进行评审。Stephen Allott 关于高效审查组的习惯的描述向我们提供了深入的经验报告（Allott 1999），这些习惯包括：

- | 把审查放在自己的工作之上
- | 谨慎地选择审查人员
- | 要求审查员努力、勤奋和优秀
- | 控制讨论过程，以最佳的评估标准记录缺陷
- | 信任作者，但是还是要核查改正情况
- | 确保度量的可信度

### 陷阱 2：评审过程没有被遵循

在采取纠正措施之前，找出评审过程没有被遵循的原因。在找出潜在的原因之后选择适当的措施纠正。如果评审过程过于复杂，参与者可能会放弃或采用其它的方法来替代。如果管理者没有通过政策来传达他们的期望，参与者就只会在方便的时候或当从个人的角度对其进行重要作用时进行评审。如果质量对于一个项目来说不是成功的驱动力，那么从同级评审中所获得的质量的提高就不会成为推动实施同级评审的主要原因。然而，同级评审所带来的对生产力的提高能够支持项目满足目标并加快产品的提交。在一个已经陷入进度超期、需求混乱、人员疲惫之困境的项目中引入同级评审是很艰苦的，但是如果评审能将项目引入正常轨道，那么这是完全值得的。

### 陷阱 3：适合的人没有参加评审

不合适的参与者包括没有被作者邀请的管理人员和没有明确目的的观察人员。参加者不是来学习的，而是应当能发现问题的人。

对于小项目，一个人要担任几个角色，因此可以邀请一些同事代表其他人的观点。对于有些评审来说，若代表关键视角的人没有出现，那么评审是不全面的。例如：对某需求规格说明书的评审需要从用户的角度判断其正确性和完整性，并快速解决含糊和冲突问题，客户可以是真正的最终用户或其替代者，如市场人员。

### 陷阱 4：评审会陷入到对问题的解决上。

除非特别要求评审成为一次头脑风暴会议，评审组应该把焦点放在发现问题而不是解决问题上。当评审会转向寻找问题解决方案时，对产品的检查变得停止了。如果参加者对于所讨论的问题不感兴趣，他们就停止了思考。当评审者意识到评审会的结束时间到了，他们会很快的翻过剩下的材料，宣布评审的成功。事实上，这些材料中很可能隐藏了主要缺陷，将会在未来长期侵扰开发组。评审组长的失败是产生这一问题的主要原因。

### 陷阱 5：评审的焦点放在文档形式而不是内容本身上

一份只记录文档形式问题的问题日志表明评审者被形式问题搞乱了，他们缺乏充足的准备，或者只做了表面的检查。为了避免这种陷阱，应当对其他文档事先定义编码标准、标准模板。编码规范规定了代码版面格式、命名约定、注释、避免采用的编程语句和其他一些提高可读性和可维护性的规则。作为检查该工作成品是否符合准入条件的一项任务，可指定一名标准核查员检查该产品是否符合相关的标准。标准的代码版面格式可以使评审者将注意力放在重要的逻辑、功能和语义问题上。



### 同级评审成功的准则

- 准则一：让同级而不是用户来发现缺陷
- 准则二：获得管理方面的支持
- 准则三：培训评审人员和评审组长
- 准则四：确保给评审和返工活动分配时间
- 准则五：为评审过程设定目标
- 准则六：评选一位从评审活动中受益的评审冠军
- 准则七：制订计划，尽量在早期并经常性地进行正式和非正式的评审
- 准则八：分析早期的评审，持续改进评审实践

将同级评审机制融入一个组织的文化中需要花费大量的时间。新的评审过程往往非常脆弱，有时我

们花费了很多时间却没有发现一个主要缺陷。那么，有什么方法可以有效地实施同级评审？本文将介绍七条进行评审的关键成功准则。

#### 准则一：让同级而不是用户来发现缺陷

参与评审的人员以及他们对于质量的态度最能决定评审的成功与否。其中首要因素是让开发组成员愿意让同级而不是用户来发现缺陷 (Wieger 1996)。“用户”包含任何以所提交的产品为基础而进行工作的那些人，如：以需求规格说明书为基础编写测试用例的测试工程师。评审的参与者必须认同同级评审所带来的许多好处，包括早期缺陷的消除、减少后期的返工、文档的质量得到评估、交叉培训和缺陷预防过程的改进等 (Gilb 2000)。一旦开发组成员理解了质量成本和投资回报的概念，他们就会克服心理上的一些障碍，例如：认为增加评审会拖延开发的进度。

#### 准则二：获得管理方面的支持

如果没有获得管理方面的支持，即使是目标明确的开发组成员也会抵制进行评审。管理的支持不仅是简单的给一个许可或简单的说“每个人都应实施评审”。管理层的支持包括建立评审策略和目标、提供资源、时间、培训和激励，并遵守评审小组的决定。

#### 准则三：培训评审人员和评审组长

48%的被调查者认为未接受培训将阻碍他们对审查的最初运用(Brykczynski 1994)。培训可以告诉人们为什么和如何进行审查，但是只有实际的经验才能给他们提供深刻的认识。

#### 准则四：确保在项目计划中给评审和返工活动分配时间

尽管人们动机良好，可当时间压力增大时他们将压缩评审的时间，而随着后来潜在缺陷的不断暴露，这将导致更大的时间压力。在项目中投入百分之几的努力用于进行同级评审是管理对质量进行支持的有力的标志。

#### 准则五：为评审过程设定目标

例如，某小组被要求评审需求规格说明书的全部内容，60%的设计说明书，75%的代码等等 (Wigers 1996)。设定数字目标可以迫使我们跟踪我们所创建的每一类成果的数量，以便测量向目标不断接近。这样我们完成了目标，同时也使同级评审成为日常工作的一部分。评审的更高目标可以是：对已完成审查的工作产品中所保留的缺陷数定义一个量化的目标，这样做将促使更细致的审查和质量测量。或者是：将返工占整个开发费用的百分比从某一已知的起点开始降至一个更低的某一点。要确保所设定的目标是可以达到的、可测量的，并符合组织的整体目标。

#### 准则六：评选一位从评审活动中受益的评审冠军

让评审冠军充当一名倡议者，他可以从经验角度而不是理论知识角度来说服其他人。一名受团队成员尊敬的评审冠军，不管他是技术人员、管理人员还是质量工程师，能够克服最初施加给同级评审的压力。一名卓越的开发者如果要求进行评审，等于传递出这样的信息：每个人都能够从同级评审中获益。

#### 准则七：制订计划，尽量在早期并经常性地进行正式和非正式的评审

应该使用那些能够满足目标的最便宜的评审方法。有时一次快速的随机评审就能达到要求，而有时却需要多种方式，如轮查、走查、头脑风暴会议、小组评审、或严格的审查。对一个正在进行开发的工作产品实施非正式的持续不断的评审可以快速并省力地过滤出大量的缺陷。其中的一个风险在于评审者可能会厌烦对文档的不断重复，或许他们会认为他们是在做作者做的事；另一个风险是：由于其他文档的变更，你不得不重复地进行评审。尽管如此，按顺序使用多种方式的评审是提高文档质量的有效的方法。

#### 准则七：分析早期的评审，持续改进评审实践

你的第一次评审可能不会象想象中那样顺利。为了改进有效性，分析你做过的早期评审，持续不断地改进评审步骤、检查表和一些经验表单。将已知的最好的审查实践，如下表所示，加入到你的评审过程中。



#### 软件审查的最佳实践 (Gilb1998, 2000)

- 制订审查计划，明确项目目标和审查目标
- 使用严格的，定量的准入和准出条件
- 首先审查上游的文档
- 在文档生成的初始阶段就开始对其进行审查
- 核对源代码和相关的文档

- 在最适当的时机进行准备和审查
- 把焦点放在主要缺陷上
- 测量审查所带来的好处
- 强调缺陷预防和过程改进

#### 第一类 文化问题

- 问题一：一些开发人员拒绝对他们的工作进行评审
- 问题二：一些开发组成员拒绝评审其他人的工作
- 问题三：评审有多余的无理事情发生，人身攻击和讽刺很普遍，作者处于防卫状态
- 问题四：评审组长不能有效地控制评审会议

#### 第二类 计划编制问题

- 问题一：评审阶段根本没有在项目计划中出现
- 问题二：评审被认为会拖延项目进度
- 问题三：当在关键时刻或时间紧张的时候就跳过评审
- 问题四：人们不能评审产品中适当的部分

#### 第三类 效率问题

- 问题一：评审人员选择了不恰当的准备方法和分析技术
- 问题二：在评审会议上重新讨论很久以前的决定，或质疑工作产品的背景
- 问题三：参加者没有为会议做好足够的准备
- 问题四：评审能发现某种缺陷但是常常漏掉其他的缺陷，很多缺陷无法通过评审被发现
- 问题五：评审总是发现同样类型的缺陷
- 问题六：评审时发现了太多的缺陷
- 问题七：审核发现了很多次要缺陷但是很少发现主要缺陷
- 问题八：评审工作实施得太晚。纠正所有的缺陷需要过多的返工

#### 第四类 管理问题

- 问题一：管理层不支持或反对进行同级评审
- 问题二：管理层没有为评审确定明确的期望目标
- 问题三：管理者要求参加他们不应参加的评审
- 问题四：管理人员不适当的使用了度量数据或要求看他们不应看的数据
- 问题五：评审中收集的数据没有在其他任何地方使用

### CMM3总结

CMM2仅定义了管理的基本过程没有定义执行的步骤标准。

CMM3则定义了一套文档化的企业范围的工程化标准，并将这些标准集成到企业软件开发的标准过程中去。这些标准过程使得管理人员和一般成员工作得更有效率。

要达到等级 3 的成熟度，必须从这两个 KPA 开始。有了这两个 KPA 等级 3 的其他 5 个 KPA 才能真正地建立起来。这一点与等级 2 是不同的，等级 2 的 6 个 KPA 在许多方面是同时的，彼此并行地建立起来，但是在等级 3 某些因素改变了这种并行性。

组织在过程改进活动方面汇聚到一起，甚至有所交叉，这导致两个巨大的变化：

第一，在组织范围内，组建了一个过程改进组（SEPG），以监督和协调软件过程改进（SPI）活动。

第二，建立起一组共同的软件开发过程与实践并被组织的全部项目所使用。

按照此思路，CMM3 的实施步骤是：

先将组织集中于组织的过程焦点，再定义组织的标准软件过程集，在此基础上，等级 3 的其他方面才能够得以实施。

请阅读课文 P.155--P.159 内容，

1、理解组间协调的目标？

2、组间协调关键过程域应实施哪些活动？

## 第五章 已管理级（第4级）

从CMM1到CMM2

CMM1	CMM2
未加定义的一种随意过程	确立了基本的项目管理控制 组织通过政策设定期望 项目重复以前成功项目的过程

从CMM2到CMM3

CMM2	CMM3
工作的焦点在项目 <b>Focus on Project</b>	重点转移到了组织 <ul style="list-style-type: none"><li>• Common Processes</li><li>• Common Measurements</li><li>• Training</li></ul>

从CMM3到CMM4， CMM5

CMM3	CMM4	CMM5
定义了度量方法，并且系统地收集度量结果。	基于收集到的数据进行决策。 <ul style="list-style-type: none"><li>• 数据测量，收集</li><li>• 分析</li><li>• 用数字进行过程决策</li></ul>	持续的过程改进成为了一种生活方式



组织为软件产品和软件过程定了**量化的质量标准**；  
量化控制将使软件开发真正成为一种工业生产活动  
两个关键过程域

- 定量过程管理
- 软件质量管理

### 5.1 定量过程管理

#### 5.1.1 定量过程管理

##### ● 定量过程管理的焦点是“过程”

- 过程能力被量化地展示
- 当过程性能不能达标时：
  - 识别原因
  - 采取纠偏行动

##### ● 定量过程管理的核心是“数据”

- 数据反映事实
- 以事实为基础的管理



#### 定量过程管理

- 为项目定义软件过程实施制定目标，测量过程的性能，以及分析得到的变量，并通过变量的调整使过程性能处于可接受的范围之内
- 目的：定量控制软件项目的过程性能
- 目标：提供统一衡量软件质量的标准，并促使软件质量的不断提高



#### 定量过程管理的基础

- 执行约定 遵循一套测量和定量控制的方针、遵循分析组织的标准软件过程能力的方针
- 执行能力
  - 具有一个工作组负责协调组织的定量管理过程活动
  - 提供足够的资源和资金
  - 具有为选定的过程和产品测量数据进行采集、记录、分析提供支持的能力
  - 实施和支持定量管理的人员接受必要的培训
  - 软件工程组和相关组成员认同定量过程管理目标和价值



#### 实施定量过程管理的活动

- 制定定量过程管理计划
- 根据定量过程管理计划，实施定量过程管理活动
- 确定数据采集的策略和相应的定量分析方法
- 采集用来定量控制软件过程的测量数据
- 分析和定量控制软件过程
- 准备并提交关于软件项目定量过程管理活动结果的书面报告
- 建立和维护组织的标准软件过程的过程能力基线



#### 评价

- 度量和分析：用于确定**软件度量管理活动**的状态

- 验证实施
  - 高级管理者定期参与评审软件定量过程管理活动
  - 项目经理既定期的又事件驱动地参与评审软件定量过程管理活动
  - 软件质量保证组评审和审计软件定量过程管理的活动和工作产品

## 5.2 软件质量管理

### 5.2 软件质量管理

- 软件质量管理的焦点是“产品”
  - ▶ 产品具有明确的、量化的质量目标
  - ▶ 质量好坏最终由用户说了算
    - 客户
    - 终端用户
    - 组织

•CMM2: Focus on “conformance to requirements”  
•CMM4: Focus on “customer satisfaction”



#### 软件质量管理

- 目的：建立对项目软件产品质量的定量了解，实现特定的质量目标
  - 包括：确定软件产品的质量目标；
  - 制定实现这些目标的计划；
  - 监控及调整软件计划、软件工作产品、活动和质量目标
- 基于三个关键过程域：集成软件管理和软件产品工程、定量过程管理



#### 软件质量管理的基础

- 执行约定：书面的管理软件质量组织策略
- 执行能力：
  - 提供足够的资源和资金
  - 实施和支持软件质量管理的人员应接受开展活动所要求的培训
  - 软件工程组和其他软件相关组成员接受在软件质量管理方面的必要培训



#### 实施软件质量管理的活动

- 项目的软件质量计划是项目质量管理活动的基础，需要根据文档化的规程制定和维护项目的软件质量计划
- 确定、监控和修订项目软件产品的定量质量目标
- 对项目软件产品的质量进行测量、分析和比较
- 将软件项目的产品定量质量目标恰当地分配给那些向项目交付软件产品的转包商



#### 评价

- 度量和分析：进行度量并将结果用于确定软件质量管理活动的状态
- 验证实施
  - 高级管理者定期参与评审软件质量管理活动
  - 项目经理既定期的又事件驱动地参与评审软件质量管理活动
  - 软件质量保证组评审和审计软件质量管理的活动和工作产品



定量过程管理的目的是定量地控制软件项目的过程性能。

软件过程性能表示遵循一个软件过程所得到的实际结果。焦点是在一个可测的稳定的过程范围内鉴别出变化的特殊原因，并且适当时改正那些促使瞬时变化出现的环境。

定量过程管理给组织过程定义、集成软件管理、组间协调、和同行评审的实践附加一个内容丰富的测量计划。

软件质量管理的目的是建立对项目的软件产品质量的定量了解和实现特定的质量目标。

软件质量管理对软件产品工程中所描述的软件工作产品实施内容丰富的测量计划。



定量过程管理的步骤：

- 确定能够表示软件质量的各种属性和指标；
- 分析软件，收集数据
- 运用公式换算软件的各种指标值
- 通过这些指标就可以分析软件的质量

- 定量过程管理的目的是定量地控制软件项目的过程性能。
- 软件过程性能表示遵循一个软件过程所得到的实际结果。焦点是在一个可测的稳定的过程范围内鉴别出变化的特殊原因，并且适当时改正那些促使瞬时变化出现的环境。
- 定量过程管理给组织过程定义、集成软件管理、组间协调、和同行评审的实践附加一个内容丰富的测量计划。
- 定量过程管理和软件质量管理——是互相紧密依赖的：
- 定量过程管理的目的是定量地控制软件项目的过程性能。
- 软件过程性能表示遵循一个软件过程所得到的实际结果。焦点是在一个可测的稳定的过程范围内鉴别出变化的特殊原因，并且适当时改正那些促使瞬时变化出现的环境。定量过程管理给组织过程定义、集成软件管理、组间协调、和同行评审的实践附加一个内容丰富的测量计划。
- 软件质量管理的目的是建立对项目的软件产品质量的定量了解和实现特定的质量目标。软件质量管理对软件产品工程中所描述的软件工作产品实施内容丰富的测量计划

软件度量是对软件开发项目、过程及其产品进行数据定义、收集以及分析的持续性定量化过程，目的在于对此加以理解、预测、评估、控制和改善。没有软件度量，就不能从软件开发的暗箱中跳将出来。通过软件度量可以改进软件开发过程，促进项目成功，开发高质量的软件产品。

软件度量贯穿整个软件开发生命周期，是软件开发过程中进行理解、预测、评估、控制和改善的重要载体。软件质量度量建立在度量数学理论基础之上。软件度量包括 3 个维度，即项目度量、产品度量和过程度量

- 软件质量管理的目的是建立对项目的软件产品质量的定量了解和实现特定的质量目标。
- 软件质量管理对软件产品工程中所描述的软件工作产品实施内容丰富的测量计划。

- 定量过程管理的步骤：
  - 确定能够表示软件质量的各种属性和指标；
  - 分析软件，收集数据
  - 运用公式换算软件的各种指标值
  - 通过这些指标就可以分析软件的质量

软件度量工作首先需要确定能够表示软件质量的各种属性和指标；然后，分析软件，收集数据；接着运用公式换算代码的各种指标值；最后，通过这些指标就可以分析代码的质量。

确定哪些属性和指标可以表示软件质量，收集哪些数据，如何用公式推导指标，都是软件度量这门科学的研究重点。它所确定的各种软件度量指标为我们了解软件属性，衡量软件质量提供了科学的依据。

# 定量过程管理与软件质量管理

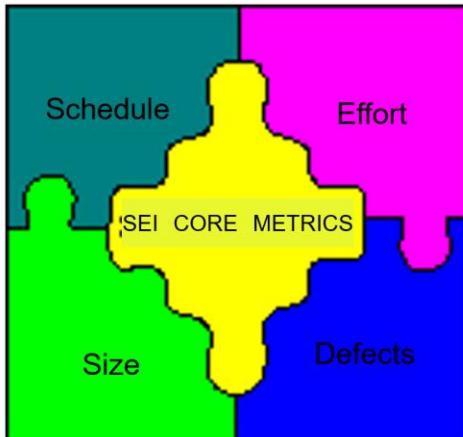
- 软件度量3维度

度量维度	侧重点	具体内容
项目维度	理解和控制当前项目的情况和状态； 项目维度具有战术性意义， 针对具体的项目进行。	规模、成本、工作量、进度、 生产性、风险、顾客满意度等。
产品维度	理解和控制当前产品的情况和状态； 用于对产品质量的预测和控制。	以质量度量为中心，包括功能性、可靠性、易用性、效率性、 可维护性、可移植性等
过程维度	理解和控制当前过程的情况和状态，包含对 过程的改善和未来过程的能力预测； 过程维度具有战略性意义，在整个组织范围 内进行。	成熟度、管理、生命周期、生 产率、缺陷植入率等

## 定量过程管理与软件质量管理

- SEI的软件核心度量项目

- 规模
- 作业量
- 进度
- 缺陷



涉及到需要量化管理的领域非常多，从事前管理和事后管理的角度来分，可以分为估算和度量两大类。

估算以实际统计调查资料为基础，根据事物的联系及其发展规律，间接地估算和预计有关事物的数量关系和变化前景。度量是依据特定的标准，衡量当前的事物与标准之间的差异。

项目管理范围内，有如下阶段需要应用估算技术：

1. 项目范围估算：项目预期的范围进行评估是项目的基础，范围估算失误将给项目带来不可挽回的损失。
2. 项目成本估算：成本估算估计完成项目各活动所需每种资源成本的近似值，成本预算的过程是把估算的总成本分配到各个工作细目，建立基准成本以衡量项目执行情况。可见成本估算的准确性直接决定成本的预算情况。
3. 项目进度估算：项目管理的关键要素之一就是时间管理，也即进度控制。准确地估算对制定项目计划、监督项目执行都有重要的意义。
4. 项目风险估算：对风险识别不到，或对风险可能造成的影响估计不足都可能导致项目失败，因此对项目风险的量化估算更是至关重要。

通常需要度量的项目要素包括：

1. 项目进度度量：对项目进度进行定期的跟踪度量，及时发现当前进度与计划的偏差，可以及时采取措施，及时赶工或调整进度计划。
2. 缺陷度量：项目的成败直接取决于客户满意度，客户满意度是个难以量化的指标，而项目成果——产品的缺陷密度直接影响着客户的满意程度。度量产品的缺陷密度，可以有效地了解项目完成的质量。
3. 项目工作量度量：工作量是衡量项目成本、人员工作情况的基础，准确地度量出项目真实的工作量，既可以

掌握当前项目的情况，对于今后估算其它项目数据也有重要意义。

4. 人员生产率度量：人力资源是项目中最为重要的资源，掌握人员的生产能力对于项目管理中人员管理、资源管理都有重要的参考价值。

定义项目、制定项目计划的时候需要进行项目估算，而项目执行过程中的跟踪监督过程则离不开度量。

良好的项目管理主要针对项目要素进行跟踪度量，通过分析度量数字就可以及时发现项目进展中存在的问题，从而有针对性地制定解决方案。

- 软件度量方法举例

- 构件级度量

- 集中评价软件构件的特性，它主要包括 3C 测量，既：内聚度（cohesion），耦合度（coupling），和复杂度（complexity）。
- 考察构件的输入，输出参数的性质和个数，全局变量，被调用模块的个数，调用的外部模块数，得出测量数据评价内聚度和耦合度。
- 分析构件控制流图得出构件的复杂度数据。

软件度量经历了几十年的发展，在软件的各个方面和领域都开发出了各种度量套件。

有针对分析模型的度量，体系结构设计的度量，构件级设计的度量，界面设计的度量，以及源代码级的度量等等，要了解这些度量套件如何进行质量测量可以参考相关的资料。

不过，这些度量套件并不是都具备实际的操作意义，有些度量就太复杂不可操作，或者脱离实际很难理解。但是也有一些度量套件具备很实际的指导意义。这里打算介绍的构件级度量套件和源码级度量套件都是很有意义的。构件级的软件度量集中评价软件构件的特性，它主要包括 3C 测量，既：内聚度（cohesion），耦合度（coupling），和复杂度（complexity），这组测量指标贴切的描述了构件设计质量，我们也能看出各种设计模式和设计原则都在想办法平衡这些元素。

内聚度和耦合度可以通过考察构件的输入，输出参数的性质和个数，全局变量，被调用模块的个数（扇出），调用的外部模块数（扇入），得出测量数据。分析公式就不列举在这里了，公式的大概含义是如果一个模块有少的输入参数，且都是数据型的参数，没有访问全局数据，被单一的模块调用，可以预计这个模块将有低的耦合度。关于复杂度还有一个著名的测量指标——环复杂度，它基于构件控制流图进行分析。

源码级的软件度量主要评价代码复杂度，Halstead 测量套件被称为“最著名和研究最完全的软件复杂度复合度量之一”。它通过研究源代码中的操作符和操作数，开发出了一系列指标，可以描述代码的实际体积，开发工作量，开发时间，甚至软件中被预测的错误数。很多软件测量工具都提供象耦合度，环复杂度，Halstead 测量套件，扇入扇出数，等指标的自动统计，透过这些指标，我们可以掌握代码的内部特性，分析每次代码改动对代码质量的影响。一些停留在定性描述上的质量改进，完全可以通过数据得到印证。比如某部分代码采用设计模式以后提高了内聚力，降低了耦合度，某块职责众多，特别复杂的代码被拆散，所拆出的各个模块的复杂度都很低，易于维护。那么通过度量数据一定可以反应出这些特性，比如，源码级的复杂度指标降低，构件级的耦合度指标降低等等。

源码级的软件度量主要评价代码复杂度，Halstead 测量套件被称为“最著名和研究最完全的软件复杂度复合度量之一”。它通过研究源代码中的操作符和操作数，开发出了一系列指标，可以描述代码的实际体积，开发工作量，开发时间，甚至软件中被预测的错误数。

很多软件测量工具都提供象耦合度，环复杂度，Halstead 测量套件，扇入扇出数，等指标的自动统计，透过这些指标，我们可以掌握代码的内部特性，分析每次代码改动对代码质量的影响。

一些停留在定性描述上的质量改进，完全可以通过数据得到印证。比如某部分代码采用设计模式以后提高了内聚力，降低了耦合度，某块职责众多，特别复杂的代码被拆散，所拆出的各个模块的复杂度都很低，易于维护。那么通过度量数据一定可以反应出这些特性，比如，源码级的复杂度指标降低，构件级的耦合度指标降低等等。

可度量性是学科是否高度成熟的一大标志，度量使软件开发逐渐趋向专业、标准和科学。尽管人们觉得软件度量比较难操作，且不愿意在度量上花费时间和精力，甚至对其持怀疑态度，但是这无法否认软件度量

的作用。

定量过程管理是建立在软件度量基础之上的一种过程管理方式，它引领我们用更加科学和严谨的态度来管理软件产品质量。

美国卡内基·梅隆大学软件工程研究所在《软件度量指南》(Software Measurement Guidebook)中认为，软件度量在软件工程中的作用有三：

- (1)通过软件度量增加理解；
- (2)通过软件度量管理软件项目，主要是计划和估算、跟踪和确认；
- (3)通过软件度量指导软件过程改善，主要是理解、评估和包装。

CMM 第 4 级最大的困难在于量化，从软件开发开始到结束，都需要大量的数据来说明。

无数的科学事实都说明，如果因为目标太难达到就不作任何工作，就不可能有任何进步。在达到最终目标之前的过程中，会有很多有益的小发现，这些发现又在不断促进新的发现，最后使不可能变成可能。

软件度量科学的发展同样在追求最终目标的过程中为我们带来了众多的有益发现，让我们用更加科学和严谨的态度来看待软件质量问题；让我们对代码的认识从定性描述阶段，进入到定量描述阶段；让我们感受到科学和美学的统一所展现出的巨大魅力

## 第六章 优化级（第5级）



CMM中的最高层次

工作重点：对已有的软件过程进行深层次的改进和过程成熟能力的不断提高



企业以“预防”、“改革”和“完善”为目标



三个关键过程域

- 缺陷预防
- 技术改革管理
- 过程变更管理

### 6.1 缺陷预防



缺陷预防(DP, Defect Prevention)

在软件过程中能识别出产生缺陷的原因，并且以此采取防范措施，防止它们再次发生

分析问题找出隐患

对可能出现错误的情况加以分析跟踪



缺陷预防的基础

- 执行约定
  - 作为一个组织开展缺陷预防活动应该遵循书面的缺陷预防策略
  - 作为一个项目开展缺陷预防活动应该遵循书面的用于遵循缺陷预防活动的策略
- 执行能力
  - 具有组织级的缺陷预防活动协调组
  - 具有协调软件项目的缺陷预防活动协调组
  - 项目层和组织层都要提供足够的资源和资金
  - 各种组织和成员需要接受有关实施缺陷预防活动的培训



实施缺陷预防的活动

- 软件项目为缺陷预防活动的开展建立一个计划
- 准备工作

- 按照已文档化的规程进行原因分析会议
- 建立定期例会制度，检查和协调措施的实施情况
- 将缺陷预防活动的有关数据文档化，并进行跟踪
- 根据文档化的规程，对组织的标准软件过程进行修订
- 对项目定义软件工程进行修订
- 对反馈信息的处理



#### 评价缺陷预防活动的方法

- 度量和分析：将结果用于确定缺陷预防活动的状态
- 验证实施
  - 高级管理者定期参与评审缺陷预防活动
  - 项目经理既定期参与软件项目的缺陷预防活动评审检查
  - 软件质量保证组评审和审计缺陷预防活动和工作产品



#### 缺陷的定义

缺陷指软件与需求的不一致，常常指软件的功能和特性与设计说明书或用户需求不一致，也可称之为bug。  
缺陷与bug不同，缺陷不仅指代码级别的错误，而且可以是在需求分析，设计和测试阶段发现的错误。  
缺陷发现越晚，修改成本越大，进度延误越严重。



#### 缺陷预防的工作重点

##### 目的

- 系统化消除缺陷，满足客户的质量需求
- 改变工作方式，由反应式向主动式转变

##### 任务重点

- 建立缺陷预防组织结构
- 定义缺陷类型
- 流程定义
- 获得支持
- 跟踪分析预防结果



#### 缺陷预防组织结构

##### 组织级DP小组

- 由SEPG、技术组、设计组和测试组负责人构成
- 侧重于一般性的原因分析，提出改善建议或制定行动计划
- 跟踪缺陷预防活动的结果

##### 项目级DP小组

- 3-4个有经验的项目组成员构成
- 侧重于缺陷数据的收集和原因分析
- 根据组织级DP小组的建议采取具体的行动措施



#### 缺陷预防的方法

缺陷预防的着眼点在于缺陷的共性原因。通过找寻、分析和处理缺陷的共性原因，实现缺陷预防。

- 分析共性原因及其类型
- 帕累托图的应用
- 鱼骨图的应用



#### 缺陷类型定义

##### 原则

- 与产品特征相结合
- 与人员成熟度水平相适应
- 必须完备

方法

- 选择一种缺陷类型定义作为初始分类
- 分析项目的缺陷数据，选择一组缺陷作为预防重点
- 利用所选缺陷数据修正初始分类
- 发布结果

## 6.2 技术改革管理



技术改革管理

- 内容：在组织内成立一个技术改革管理组，识别、选择和评估新技术，使其有效的融合到组织中
- 目标：改善软件质量，增加生产率，缩短产品开发周期



技术改革管理的基础

- 执行约定
  - 组织以书面方式写出技术改革管理的目标，并对目标的内容进行详细阐述
  - 由高层管理人员负责技术改革管理活动的主要内容
  - 由高层管理人员监督技术改革管理活动的主要方面
- 执行能力
  - 需要有技术改革管理活动组
  - 提供足够的资源和资金
  - 对数据的收集和分析的支持
  - 可以得到软件工程和软件工作产品的正确数据
  - 做好必要的培训



实施技术改革管理的主要活动

- 组织制定和维护技术改革管理的计划
- 识别技术改革的区域
- 采取一定的措施使软件管理人员和技术人员不断地了解新技术
- 分析标准软件过程中采用新技术可能收益的情况
- 为组织和软件项目挑选和获得新技术
- 为新技术的引入做一些先导性的试验工作
- 将合适的新技术纳入组织的标准软件过程和项目定义软件过程中



评价技术改革管理的方法

- 度量和分析
  - 度量技术改革活动的数量、类型、规模、实施技术改革的效果和对比预期目标所进行的分析
- 验证实施
  - 高级管理者定期对组织的技术改革管理活动进行检查
  - 有软件质量保证组对技术改革管理活动和工作产品进行检查或审计

在成熟度的各个级别都会进行技术创新，但是，CMM5这个级别的技术创新有如下特点：

- 以有序的方式有计划地将新技术引入过程
- 制度化，被引入的新技术要纳入组织标准软件过程和项目定义软件过程中

## 6.3 过程变更管理



## 过程变更管理

- 内容：改进组织所用软件过程的实践活动
- 活动
  - 定义过程改进目标
  - 不断地、系统地识别、评价、改进和完善组织的标准软件过程和项目定义软件过程
  - 制定培训和激励计划，促使组织中的每个人都参与过程改进活动。



## 过程变更管理的基础

- 执行约定
  - 书面的过程变更管理活动的策略
  - 高级管理人员负责组织软件过程改进活动
- 执行能力
  - 足够的资源和资金
  - 要进行软件过程改进方面的培训



## 实施过程变更管理的主要活动

- 制定软件过程改进大纲
- 组织软件过程活动组负责协调软件过程改进工作
- 制定和修改软件过程改进计划
- 按照软件过程改进计划执行软件过程改进活动
- 处理软件过程改进的建议
- 组织的成员积极参加到小组中，为指定的过程域进行软件过程改进
- 做好软件过程改进纳入标准实践之前的准备工作
- 当软件过程改进的任务确定后，依据文档化的规程开展实施改进工作
- 做好软件过程改进活动的记录工作
- 了解反馈信息



## 评价过程变更管理的方法

- 度量和分析
  - 度量改进对质量、生产率、开发周期的影响，将度量结果用于确定软件过程改进活动的状态
- 实施验证
  - 高级管理者定期参与评审软件过程改进活动
  - 软件质量保证组对软件过程改进的活动和工作产品进行评审和审计
  - CMM5并不是终点



## LEVEL5是进一步过程改进的基础

- LEVEL5的组织本身也在持续不断地改进：改善、创新
- LEVEL5组织中的每一个人都需要参与过程改进过程改革任重而道远

## 总结：理解CMM应注意的问题



它仅指明该做什么，而没有指明如何做。



它仅指明该做的关键内容，仅描述软件过程的本质属性，而并非面面俱到。



软件过程是指软件工程过程、软件管理过程和软件组织过程三者的有机结合。



它是从软件过程的角度考虑问题，而并非关注软件开发工具，与框架软件生存周期无关，也与所采用的开发技术无关。



CMM为改善整个企业的软件过程提供了指南，而并非针对某个具体项目。



基于CMM的过程改善投资力度大、周期长，而技术投资则可能在短期内有较快回报。

## 第七章 CMM与CMMI



CMMI的全称为：Capability Maturity Model Integration，即能力成熟度模型集成。它是SEI开发的一种评估和测算一个组织的软件开发与集成过程的方法。



CMMI集成了以下四个模型部件：

- SW-CMM（软件工程成熟度模型）
- SE-CMM（系统工程成熟度模型）
- IPPD-CMM（集成产品和过程开发成熟度模型）
- SS-CMM（采购供应成熟度模型）



CMMI的表示方法

- 阶段式表示
  - 初始级
  - 受管理级
  - 已定义级
  - 定量管理级
  - 持续优化级
- 连续式表示
  - 四类过程区域：过程管理、项目管理、过程、支持
  - 每类过程区域又分为基本的和高级的



CMMI的目标

- 成本效益：减少理解和培训上的成本
- 改进模型：统一模型利于统筹进行分析和计划
- 避免封闭的过程改进：过程按照学科单独进行，没有顾及整体效益
- 交流：跨越部门学科的过程带来更多的交流，从而利于紧密的、有效的、精简的、继承的过程



CMMI框架结构的基本思想

- 根据信息的不同作用进行分类，划分为十二种构件
- 整个模型由此十二种构件组成
- 每个构件由一个或者多个资料组成
- 整个模型汇编了数千个大的资料
- 模型的不同表示法共同使用这些资料，不同表示法通过构件的不同结构来体现



CMM 到 CMMI的各级映射



CMM和瀑布思想相联系



CMMI和叠代思想联系得更紧密



选择CMM还是CMMI？

- 实施企业的业务特点
  - 软件开发: CMM
  - 软件开发+硬件开发: CMMI
- 实施企业对过程改进的熟悉程度
  - 未涉及过程改进: CMM
  - 已实施过程改进: CMMI
- 实施企业对过程改进项目的预算
  - 实施CMMI的费用肯定要比实施CMM高

#### CMM与ISO9000



ISO9000标准系列框架

- ISO: International Standard Organization
  - 1974年成立，日内瓦
  - 工作领域涉及除电工、电子以外的所有学科
- ISO9000: ISO于1987年公布，组成
  - 质量术语标准
  - 质量保证标准
  - 质量管理标准
  - 质量管理和质量保证标准的选用和实施指南
  - 支持性技术标准



- 对质量管理领域中常用的质量术语进行定义
- 包括（67条术语解释）
  - 基本术语（13个）
  - 与质量有关的术语（19个）
  - 与质量体系相关的术语（16个）
  - 与工具和技术相关的术语（19个）



支持性标准

- 8个标准
- 4个正在制定的标准



ISO9000质量保证标准

- ISO9000系列的核心内容
- 包括三个模式
  - ISO9001: 设计、开发、生产、安装和服务的质量保证模式；20个要素；包括ISO9002和ISO9003；标准多、评估费用高
  - ISO9002: 生产、安装和服务的质量保证模式；19个要素
  - ISO9003: 最终检验和试验的质量保证模式；16个要素



相同点

- 在国际上有影响的质量评估体系
- 在降低软件开发风险、诊断与评价软件产品质量等方面都做出了突出贡献



### 不同点

- 1、适用范围不同
  - ISO9000的标准涉及从原料供应到产品销售的每一个环节
  - CMM侧重软件开发和改进过程
- 2、标准的侧重面不同
  - ISO9000标准的论证结果：通过和不通过
  - CMM将软件成熟能力评价为五个级别
- 3、论证结果包括的层次不同
  - ISO9000标准的论证结果：通过和不通过
  - CMM将软件成熟能力评价为五个级别
- 4、质量管理应用的程度不同
  - ISO9000-3属于软件质量保证的水平；ISO9000论述了可接受的产品质量的最小集合
  - CMM强调过程控制和过程管理，更符合软件产品的开发特点
- 5、应用的领域不同
  - ISO9000标准：质量评估机构的主要工具
  - CMM：帮助软件开发企业自我诊断，软件质量评估机构咨询、诊断、评价的重要工具

### 再看模型之间的差异



取得ISO 9001认证并不意味着完全满足CMM某个等级的要求

- ISO 9001标准只是质量管理体系的最低可接受准则，ISO 9001认证合格的企业至少能满足CMM第2级的大部分要求以及第3级的一部分要求。



通过CMM第2级（或第3级）评估并不代表满足ISO 9001的要求



CMM是专门针对软件开发企业设计的，因此在针对性上比ISO 9001要好，但需要注意的是，CMM强调的是软件开发过程的管理，对于国内软件企业涉及较多的“系统集成”并没有考虑，如果单纯按照CMM的要求建立质量体系，则应该注意补充“系统集成”方面的内容。

### Term Paper

针对CMM的某一个关键过程域的内容展开深入研究，探讨该关键过程域的实施方针、步骤、关键问题及其解决方案。要求：

- 论文不少于3000字
- 要有自己的独立见解
- 2012年7月10日之前提交打印版