第二十六章品质管理

26.1质量概念

- 设计质量:要求、规格书和系统的设计
- 一致性质量:实现
- 用户满意=合规产品+良好质量

  +在预算和进度内交付

定义:软件质量

符合明确声明的功能和性能要求，明确记录的开发标准，以及所有专业开发的软件应有的隐含特征。

26.1质量概念

- **质量成本**
  - 预防成本

    质量规划、正式的技术审查、测试设备、培训……
  - 评估成本

    过程检验、设备校准和维护、测试……
  - 失败的成本
    - **内部故障成本:返工、维修、发货前的故障模式分析**
    - **外部故障成本:投诉解决、产品退换货、帮助线支持、保修工作……**

26.2软件质量保证

- **SQA活动**

1. 准备一个项目的质量保证计划，该计划确定
   - 进行评估
   - 进行审计和审查
   - 适用于项目的标准
   - 错误报告和跟踪的程序
   - 由质量保证小组制作的文件
   - 提供给软件项目团队的反馈数量
2. 参与项目软件过程描述的开发

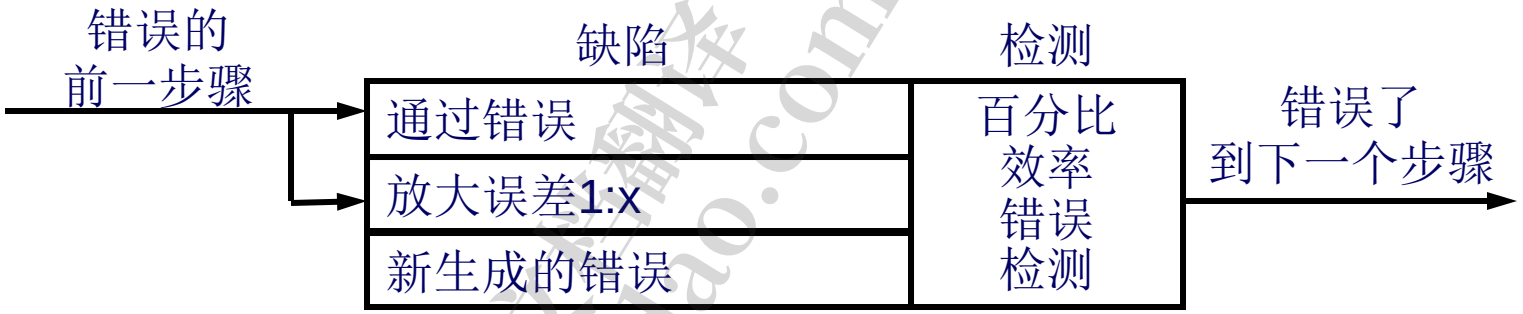
26.2软件质量保证

- **SQA活动(持续)**

3. 评审软件工程活动，以验证与定义的软件过程的合规性
4. 审核指定的软件工作产品，以验证与定义为软件过程一部分的产品的合规性
5. 确保软件工作和工作产品中的偏差形成文件，并按照文件化程序处理
6. 记录任何不合规的情况，并向高级管理层报告

26.3软件的评论

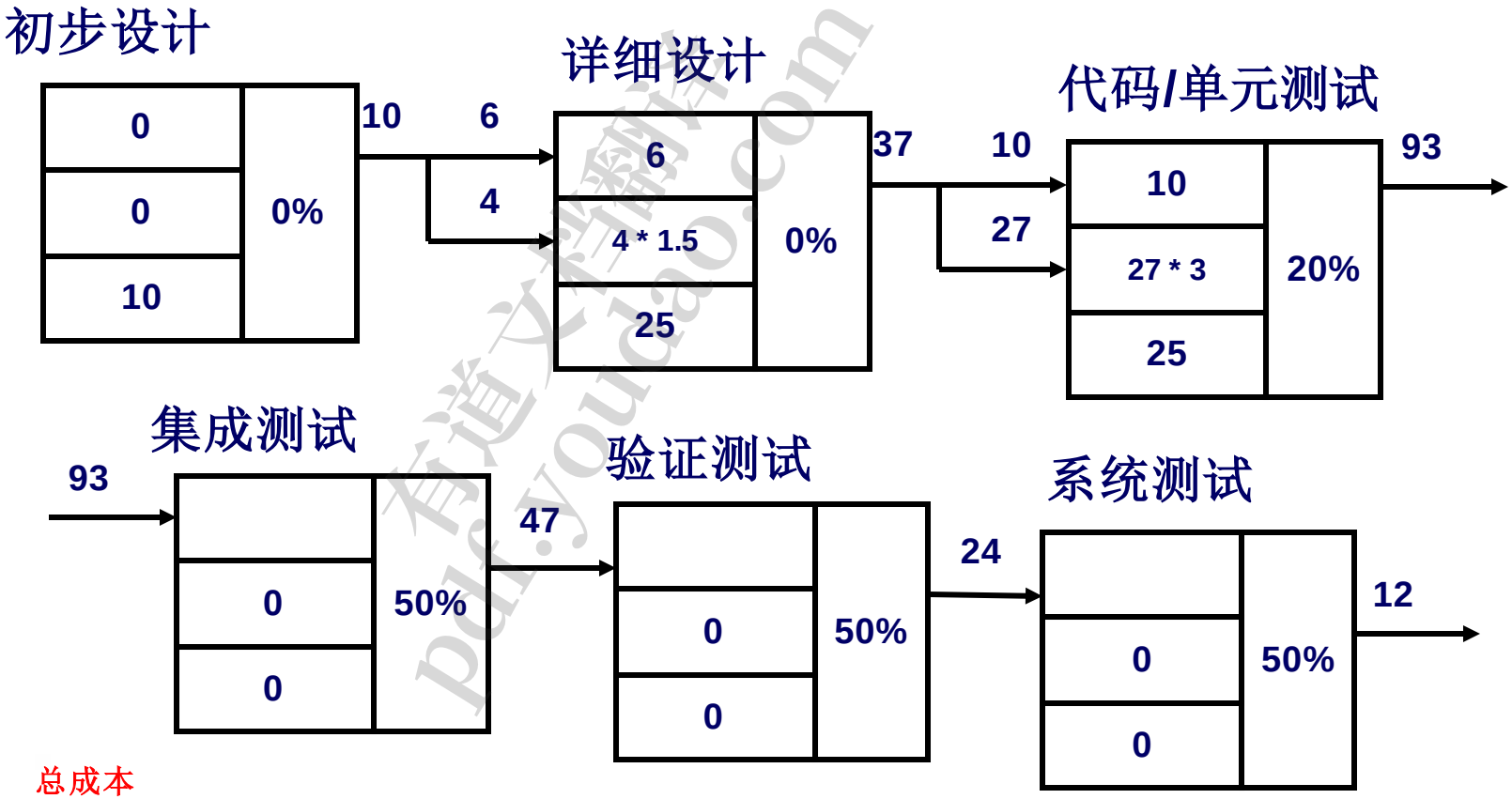
- 缺陷放大模型



- 假设在设计过程中发现的错误将花费1.0货币单位来纠正。相对于这个成本，在测试开始前发现的相同错误将花费6.5个单位;测试期间，15个单位;发布后，60 - 100个单位。
- 大量研究表明，设计活动引入了软件过程中50% - 65%的所有错误。然而，正式审查技术已被证明在发现设计缺陷方面有高达75%的有效性。

26.3软件的评论

示例:缺陷放大-无评论

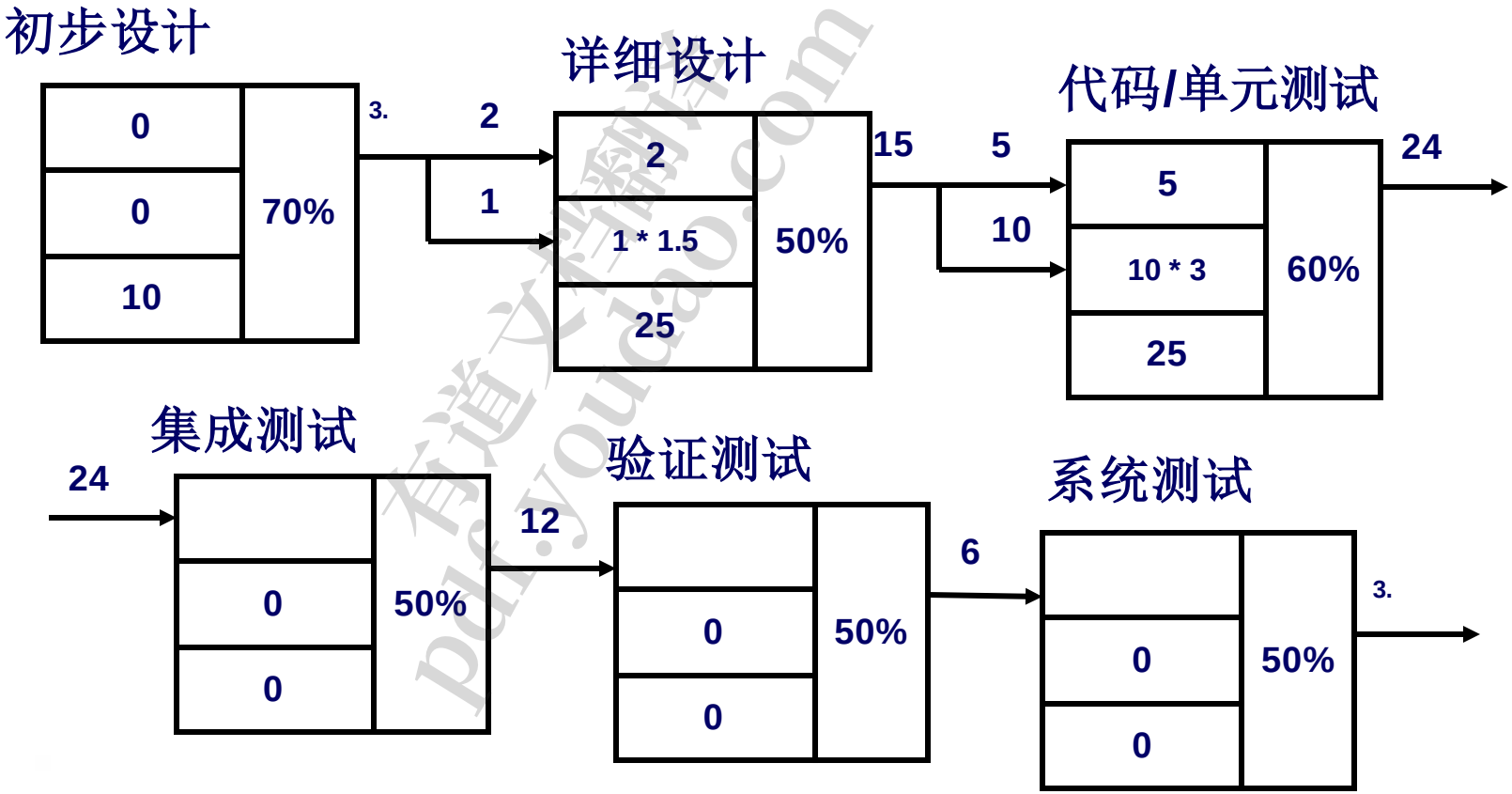


总成本

= (10 + 27 * 3 + 25) * 20% * 6.5 + (93 + 47 + 24) * 50% * 15 + 12 * 67 = 2183.5

26.3软件的评论

例子:缺陷放大-与评论



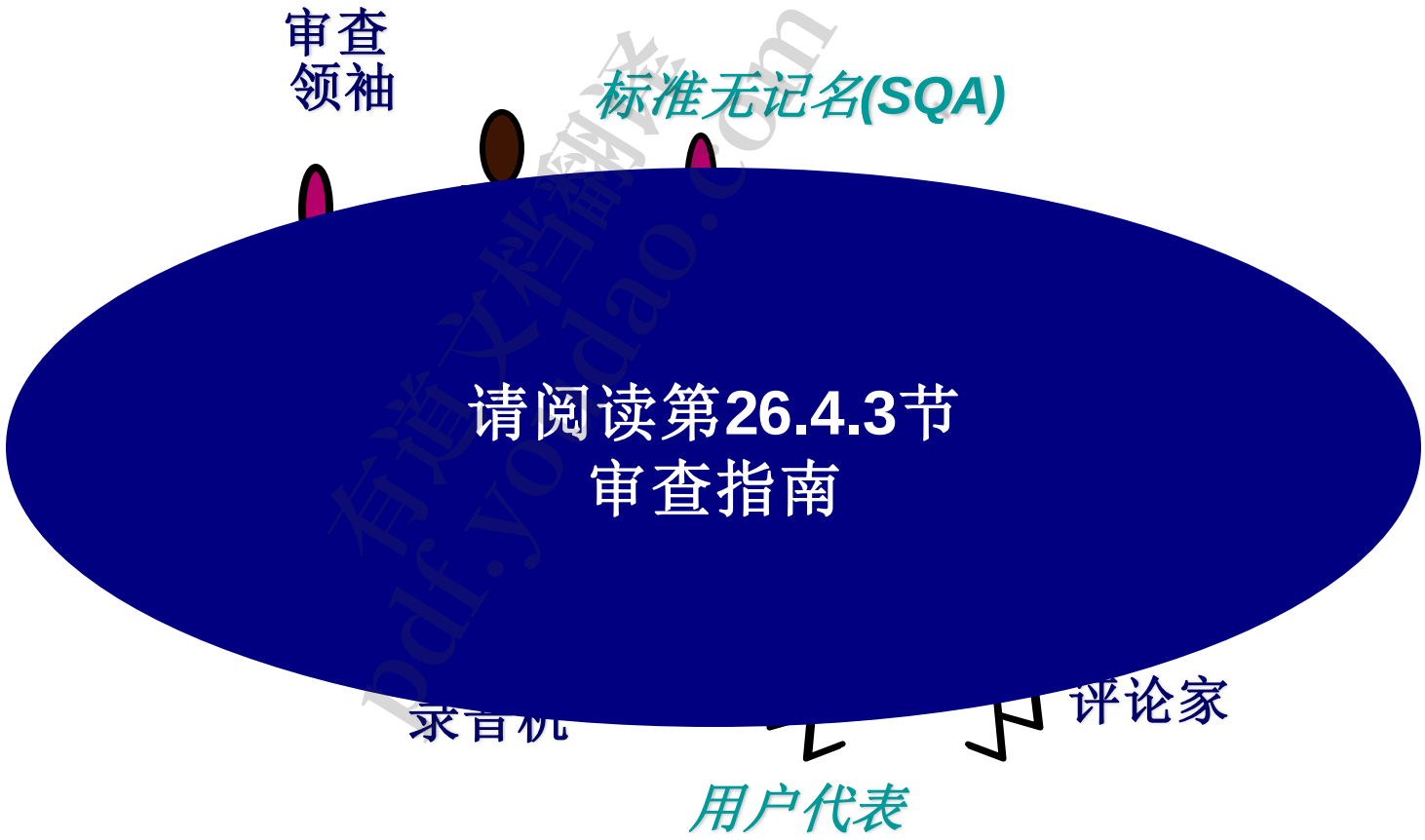
总成本= (10 * 70% + 28.5 * 50%)* 1.0 + (5 + 10 * 3 + 25)* 60% * 6.5
+ (24+12+6)*50%*15 + 3*67 = 771

26.4正式的技术审查

- 目标

1. 发现错误
2. 验证软件是否符合要求
3. 确保软件已按照预先定义的标准表示
4. 以统一的方式实现软件
5. 使项目更易于管理
6. 充当训练场

26.4正式技术审查



26.6统计SQA

产品
&过程

测量表示“

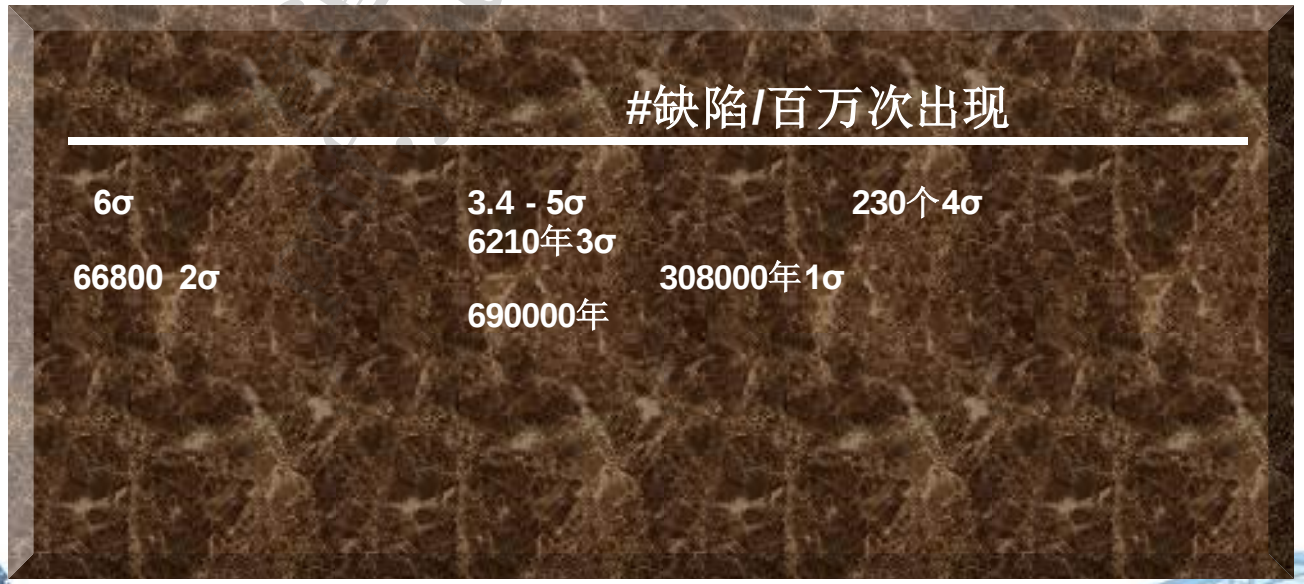
- 收集所有缺陷的信息
- 找出缺陷的原因
- 为流程提供解决方案

...对如何
提高质量.....

26.6统计SQA

- 软件工程的六西格玛(Motorola 1980年)

  6 σ这个术语来源于6□(标准偏差)，意味着极高的质量标准。



26.6统计SQA

- 六西格玛方法定义了三个核心步骤:
  1. *通过明确定义的客户沟通方法来定义客户需求、可交付成果和项目目标*
  2. *测量现有的过程及其输出，以确定当前的质量表现(收集缺陷指标)*
  3. *分析缺陷指标并确定关键的少数原因(80%的缺陷可以追溯到20%的所有可能原因)。*
     - *通过消除缺陷的根本原因来改进过程。*
     - *控制过程，确保将来的工作不会重新引入缺陷的原因。*

     或
     - *设计过程以避免缺陷的根本原因，并满足客户要求。*
     - *验证过程模型将实际上避免缺陷并满足客户要求。*

26.7软件可靠性

可靠性= MTBF = MTTF + MTTR

可用性=(舒曼，1983)

$$\frac{MTTF}{MTTF + MTTR}$$

其中MTTF =平均失效时间=

$$\frac{1}{n}\sum_{i=1}^{n} t_{ui}$$

MTTR =平均修复时间=

$$\frac{1}{n}\sum_{i=1}^{n} t_{di}$$

软件安全的重点是识别和评估可能对软件产生负面影响并导致整个系统失效的潜在危险。

第27章变更管理

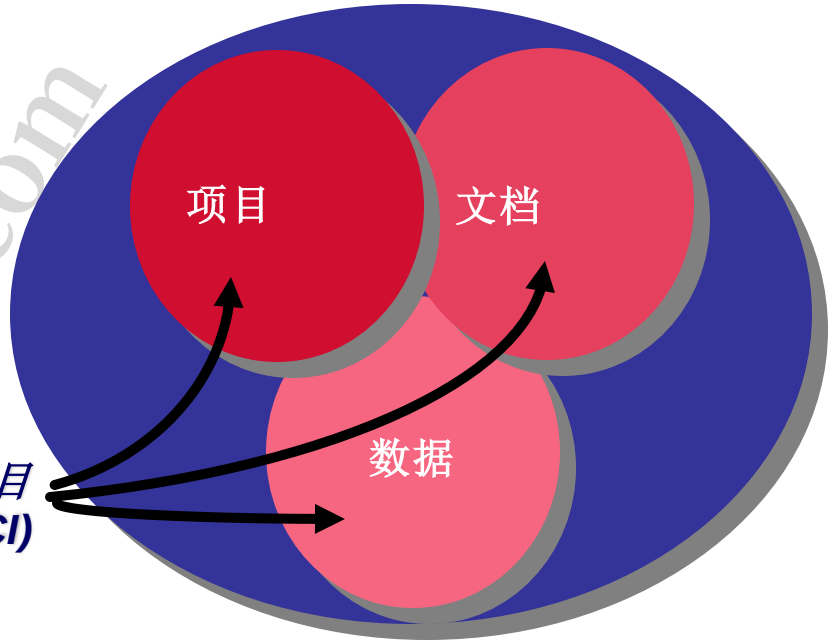
“第一定律”

无论你处于系统生命周期的哪个阶段，系统都会发生变化，而想要改变它的愿望会贯穿整个生命周期。

伯索夫等人，1980年

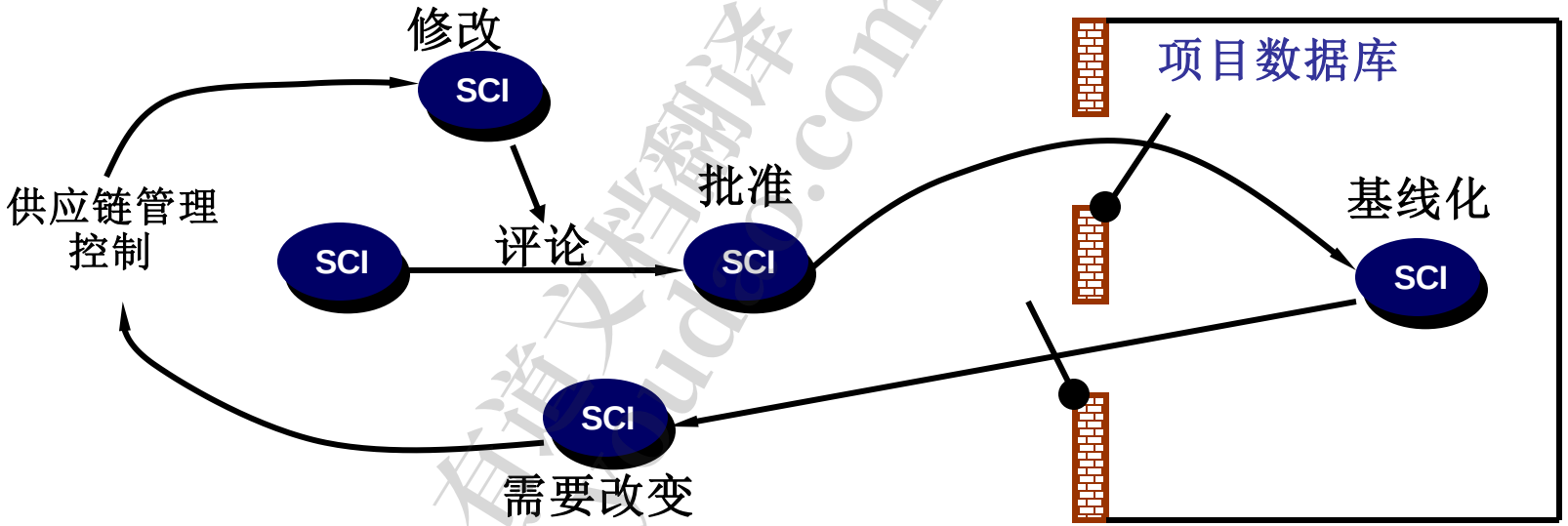
27.1软件配置管理

- 软件配置:包含作为软件过程一部分产生的所有信息(程序、数据、文档等等)的项目统称为软件配置。



- 基线(IEEE Std. No.610.12-1990):一个已经正式审查和商定的规格或产品，此后作为进一步开发的基础，并且只能通过正式的变更控制程序进行更改。

27.1软件配置管理



基线:

系统规格说明软件需求

设计规范源代码

测试计划/程序/数据操作系统

27.3配置SCM流程

1. 识别:组织如何识别和管理程序(及其文档)的许多现有版本，使其能够有效地适应变化?
2. 版本控制:一个组织如何控制软件发布给客户之前和之后的变更?
3. 变更控制:谁负责批准和排序变更?
4. 配置审计:我们如何确保正确地进行了更改?
5. 报告:使用什么机制来评估其他人所做的更改?