软件配置管理

主讲：梁跃虹

2013年6月

配置管理概念

- 什么是配置管理？
- 为什么要做配置管理？

软件项目中典型的SCM问题

- 找不到某个文件的历史版本
- 工作成果被覆盖
- 开发人员使用错误的程序版本
- 开发人员未经授权修改代码或文档
- 人员流动，交接工作不彻底
- 已经解决的缺陷过后又出现
- 因协同开发，或者异地开发，版本变更混乱导致整个项目失败

配置管理概念

- **软件配置管理**（Software Configuration Management，SCM）是对软件开发和维护过程中的变更进行跟踪和控制的管理过程。其最终目标是实现软件产品的完整性、一致性、可追溯性。

配置管理的目标

SCM活动的目标就是为了

① 标识变更

② 控制变更

③ 确保变更正确实现

④ 向其他有关人员报告变更

一、配置标识

- 软件配置管理的对象就是SCI—**软件配置项**

配置标识是软件生命周期中划分选择各类配置项、定义配置项的种类、为它们分配标识符的过程。配置项标识的重要内容分下面两方面：

- 配置项选择和划分
- 配置项标识和命名

配置项选择和划分

一个软件配置项是项目中一个特定的、可文档化的**工作产品集**。

表1 软件配置项的分类、特征和举例

分类	特征	举例
环境类	软件开发环境及软件维护环境	编译器、操作系统、编辑器、数据库管理系统、开发工具（如测试工具）、项目管理工具、文档编辑工具
定义类	需求分析及定义阶段完成后得到的工作产品	需求规格说明书、项目开发计划、设计标准或设计准则、验收测试计划
设计类	设计阶段结束后得到的产品	系统设计规格说明、程序规格说明、数据库设计、编码标准、用户界面标准、测试标准、系统测试计划、用户手册
编码类	编码及单元测试后得到的工作产品	源代码、目标码、单元测试数据及单元测试结果
测试类	系统测试完成后的工作产品	系统测试数据、系统测试结果、操作手册、安装手册
维护类	进入维护阶段以后产生的工作产品	以上任何需要变更的软件配置项

常见的软件配置项

1、 系统规格说明
2、 软件项目计划
3、 软件需求规格说明书
4. 初步用户手册
5. 设计规格说明书
   a. 数据设计描述
   b. 体系结构设计描述
   c. 模块设计描述
   d. 接口设计描述
   e. 对象描述（采用面向对象技术时）
6. 源代码清单
7、 测试规格说明
   a. 测试计划和步骤
   b. 测试用例和记录的结果
8、操作和安装手册
9、 可执行程序
10、数据库描述
   a. 模式和文件结构
   b. 初始内容
11、联机用户手册
12、维护文档
   a. 软件问题报告
   b. 维护请求
   c. 工程变更指令

配置项标识规则

- 每个配置项都必需被唯一地标识，这个唯一的标识被用于与其它配置项进行区分，跟踪和报告该配置项的状态。配置项的标识包括两个方面：
  - 配置项命名
  - 配置项版本标识

例：命名规则

a. 文档

对所有文档而言，文件名就作为配置项的命名。例如：RDMIS_需求规格说明书

b. 代码

使用“项目名-模块名＋代码”或者“项目名＋代码”的方式进行命名。项目名、模块名：根据软件设计文档对软件结构的划分选择缩写名。例如：AA项目组的BB模块，代码配置项命名为：AA-BB代码

c. 工具

以工具本身的名称命名。

版本标识

- 单个配置项在每一次修改后都会发生变化，为了标识配置项在两次修改之间的不同，需要对配置项的版本进行标识。

例：版本标识规则

✓ 配置项的版本标识建议采用的形式为：xx.yy的十进制标识符，其中xx起始为“1”，yy起始为“0”。如： RDMIS_详细设计说明书 V1.0

✓所有数字均是阿拉伯数字，并且单调递增。如果发生了重大的修改，xx递增；如果只有小修改，递增yy。

✓对于软件产品发布版本建议以xx.yy.zz.pp的形式标识，即：**主版本号.次版本号.维护版本号.补丁版本号**。所有上述数字都是阿拉伯数字，并且单调递增。

建立配置管理环境

配置库是指在软件生命周期的某一阶段结束时，存放作为阶段产品而发布的、与软件开发工作有关的信息的库。

配置库存储配置项（SCI）、修改请求、变化记录等，并提供对库中所存储文件的版本控制

配置库的作用

- 记录与配置相关的所有信息
- 利用库中的信息可评价变更的后果
- 利用库中的信息查询

- 运行一个给定的系统版本需要什么硬件和系统的哪些版本？
- 一个系统到目前已生成了多少版本，何时生成的？
- 如果某一特定的构件变更了，会影响到系统的那些版本？
- 一个特定的版本曾提出过那几个变更请求？
- 一个特定的版本有多少已报告的错误？

三库管理

开发库

- 用于存放项目期间处于开发状态的相关代码。库中的信息可能有较为频繁的修改，只要开发库的使用者认为有必要，无需对其做任何限制。因为这通常不会影响到项目的其他部分。

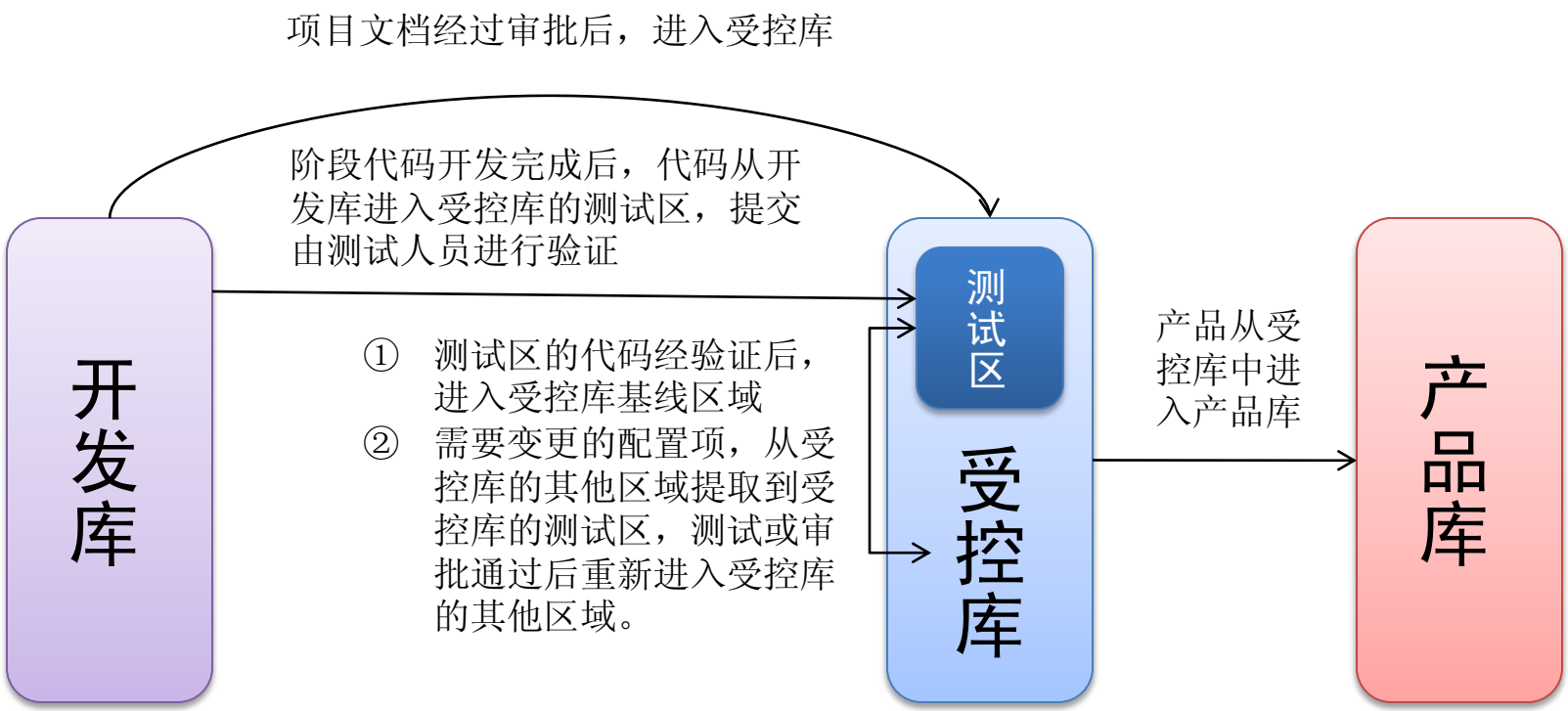
受控库

- 在软件开发的某个阶段工作结束时，用于存放经过验证或审批的产品。建立测试区，用于存放开发工作结束后需要进入测试的配置项，以及为变更实施提供工作空间。应该对库内信息的读写和修改加以控制。

产品库

- 在开发的软件产品完成系统测试之后，作为最终产品存入库内，等待交付用户或现场安装。库内的信息也应加以控制。

三库管理示例


项目文档经过审批后，进入受控库
阶段代码开发完成后，代码从开发库进入受控库的测试区，提交由测试人员进行验证
开发库
测试区
受控库
① 测试区的代码经验证后，进入受控库基线区域
② 需要变更的配置项，从受控库的其他区域提取到受控库的测试区，测试或审批通过后重新进入受控库的其他区域。
产品从受控库中进入产品库
产品库

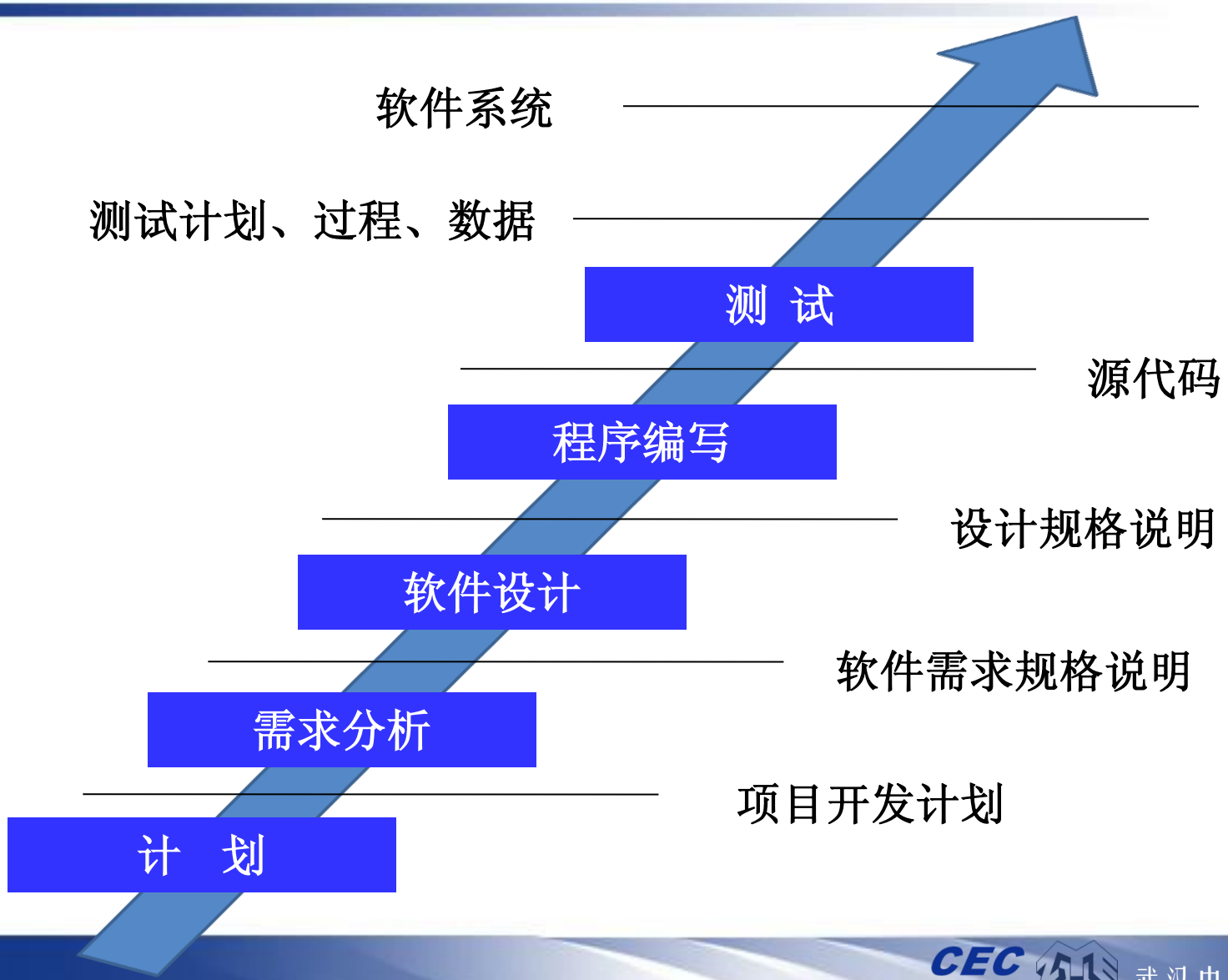
基线

- **基线**是软件生存期各开发阶段末尾的特定点，也称为里程碑，它以一或多个软件配置项的交付为标志。
- 基线由已经通过正式评审和批准的某规约或产品组成，因此可以作为进一步开发的基础。
- 基线**必须经过正式的变更控制程序才能够改变**。
- 基线的作用是把各阶段工作的划分更加明确化，以便于检验和肯定阶段成果。

基线的特点

- 通过正式的评审过程建立
- 基线存在于基线库中，对基线的变更接受更高权限的控制
- 基线是进一步开发和修改的基准和出发点
- 进入基线前，不对变化进行管理或者较少管
- 进入基线后，对变化进行有效管理
- 不会变化的东西不要纳入基线
- 变化对其他没有影响的可以不纳入基线

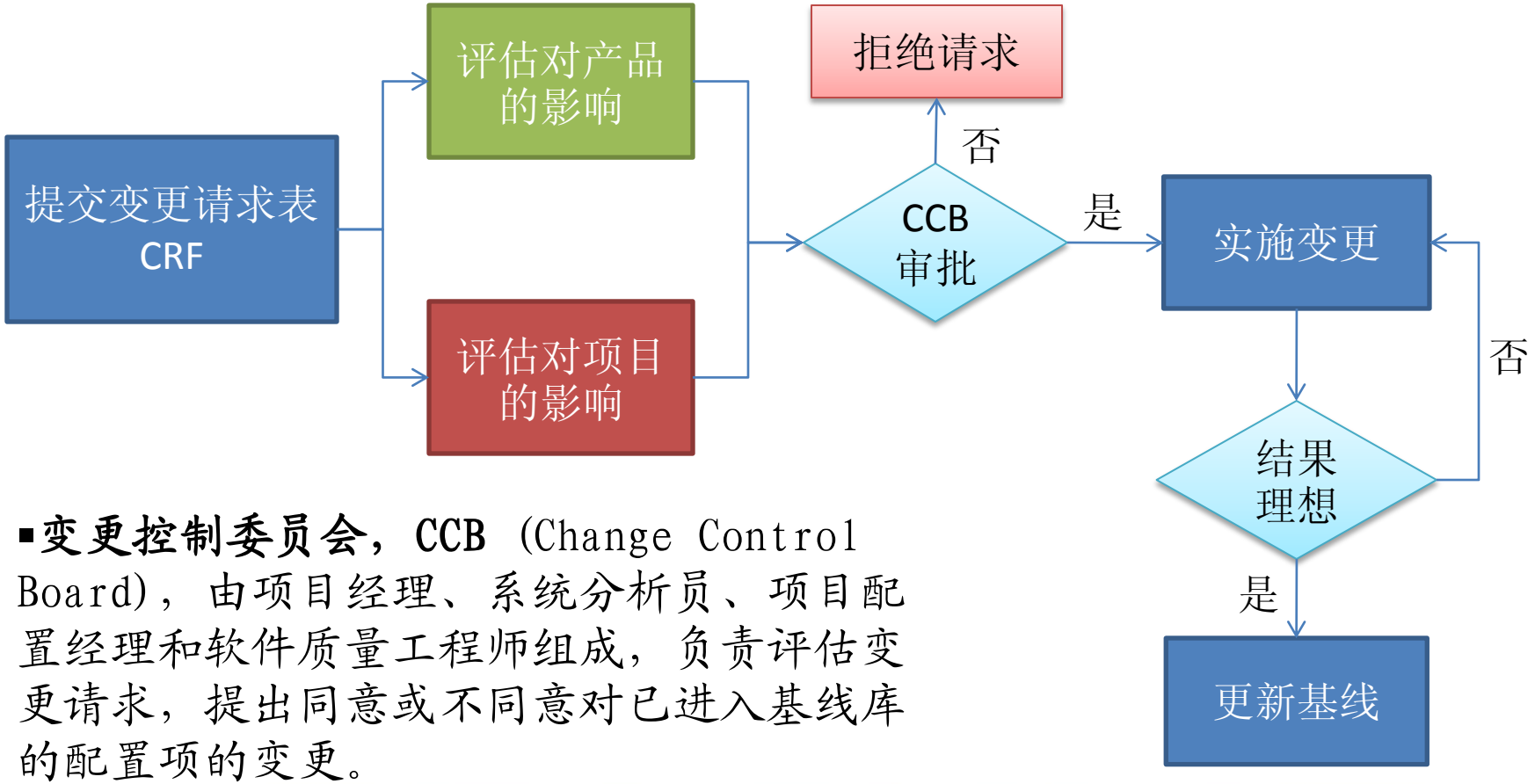
软件开发各阶段的基线


软件系统
测试计划、过程、数据
测 试
源代码
程序编写
设计规格说明
软件设计
软件需求规格说明
需求分析
项目开发计划
计 划

二、变更控制

- 通过建立产品基线，控制软件产品的发布和在整个软件生命周期中对软件产品的修改。
- 目的是建立确保软件产品质量的机制。它回答：受控产品怎样变更？谁控制变更？何时接受，恢复，验证变更？
- 变更控制包括建立控制点和建立报告与审查制度

变更控制过程



- **变更控制委员会，CCB** (Change Control Board)，由项目经理、系统分析员、项目配置经理和软件质量工程师组成，负责评估变更请求，提出同意或不同意对已进入基线库的配置项的变更。

变更请求的主要内容

- 变更描述
- 对变更的审批
- 有关变更实施的一些信息

表2 变更请求表CRF

项目名		变更请求标识	

变更请求：　　变更请求人 _____　日期 _____

变更理由 _____

变更描述 _____

影响范围 _____

变更优先性考虑 _____

评估变更工作量 _____

分析与评估　　分析人 _____　日期 _____

分析与评估意见 _____

审批　　CCB 负责人 _____　日期 _____

CCB 审查意见 _____

变更实施　　实施负责人 _____　日期 _____

变更实施情况 _____

质量保证审查　　QA 负责人 _____　日期 _____

审查意见 _____

配置管理审查　　CM 负责人 _____　日期 _____

审查意见 _____

三、配置状态报告

配置状态报告（CSA, Configuration Status Accounting ）

是一种配置管理活动，对开发过程作出系统的记录。它报告已批准的基线和过程的当前状态，和已提出并批准的请求变更的状态。

-**任务**：有效的记录和报告管理配置所需要的信息。

-**目的**：及时、准确的给出软件配置项的当前状况，供相关人员了解，以加强配置管理工作。

- *发生了什么（What）？*
- *为什么要发生（Why）？*
- *谁做的（Who）？*
- *什么时候发生的（When）？*
- *在哪儿改变的（Where）？*

配置状态报告

典型的工作产品包括

- – 配置项修订记录
- – 变更日志
- – 变更请求
- – 配置项状态
- – 基线间差异

配置状态报告示例

定期提交的配置状态报告的内容示例

- —各份变更请求概要：变更请求号、日期、申请人、状态、估计工作量、实际工作量、发行版本、变更结束日期
- —基线库状态：库标识、至某日预计库内配置项数、实际配置项数
- —发行信息：发行版本、计划发行时间、实际发行 时间、说明
- —备份信息：备份日期、介质、备份存放位置
- —配置管理工具状态
- —配置管理培训状态

配置项状态报告示例

配置项状态报告									
项目名称	项目编号		项目负责人			CM工程师		报告日期	
配置项名称	版本	配置项状态	计划入库时间	作者	入库时间	出库时间	所属基线及版本	配置项存放路径	变更次数
		开发中							
		受控库							
		基线化							

四、配置审计

配置审计是对配置管理的独立的查检过程，确认受控软件配置项满足需求并就绪。

验证包括:

- 软件产品的构造是否符合需求、标准、或合同的要求;
- 配置标识的准则是否得到了遵循;
- 变更控制规程是否已遵循，变更记录是否可供使用
- 是否保持了软件产品的可追溯性

配置审计的必要性

为什么要实施配置审计

确保软件配置管理的有效性，不允许出现任何混乱现象。

例如:

- 防止出现向用户提交了不适合的产品，如交付了用户手册不适当的版本;
- 发现不完善的实现，如开发出不符合初始规格说明或未按变更请求实施变更;
- 找出各配置项间不匹配或不相容的现象;
- 确认配置项已在所要求质量控制审查之后作为基线入库保存;
- 确认记录和文档保持着可追溯性。

配置审计实施

1、实施配置审计的时机

——软件产品交付或是软件产品正式发行前

——软件开发的阶段工作结束之后

——在维护工作中，定期的进行

2、实施配置审计的责任人

参与实施配置审计的人员包括：项目组人员和非项目组人员，例如其他项目的配置管理人员、软件组织的内部审核员以及软件组织的软件配置管理人员。

配置审计实施

3、配置审计工作的开展

（1）由项目经理决定何时进行配置审计工作

（2）质量保证组或软件组的配置管理组指定该项目的配置审计人员

（3）项目经理和配置审计员决定审核范围。

（4）配置审计员准备配置审计检查单

（5）配置审计员安排时间审核文档和记录，审核活动可能涉及到：

项目范围	配置项的检入（check-in）及检出（check_out）
评审记录	配置项的变更历史
测试记录	文件的命名
变更请求	版本的编号

（6）配置审计员在审核中发现不符合现象，并作记录。

（7）由项目经理负责消除不符合现象。

（8）配置审计员验证所有发现的不符合现象确已得到解决。

五、软件配置管理工具

软件配置管理工具的主要功能

- 版本控制
- 变更管理
- 配置审核
- 状态统计（查询和报告）
- 问题跟踪（跟踪缺陷和变更）
- 访问控制和安全控制

常用的配置管理工具

- ClearCase & ClearQuest
- CVS
- Subversion（SVN）
- PVCS
- Harvest
- Visual SourceSafe（VSS）

培训小结

- 理解软件配置管理的作用和有关软件配置管理的相关概念（配置项、基线、CCB）
- 识别纳入基线的配置项
- 了解配置项变更控制过程
- 了解配置审计和配置状态报告的作用

谢　谢！