



IBM Software Group

# 什么是好的软件配置管理？

Rational software

IBM 软件部 李纪华

IBM developerWorks **Live!**  
The Technical Conference with More  
**2005 开发者大会**

**ON DEMAND BUSINESS**

© 2005 IBM Corporation

## 软件开发管理的最佳实践经验

- 迭代化开发
- 管理需求
- 使用组件架构
- 可视化建模（UML）
- 持续质量验证
- 管理变更



## 软件开发管理的最佳实践经验

- 迭代化开发
- 管理需求
- 使用组件架构
- 可视化建模（UML）
- 持续质量验证
- **管理变更**



# 为什么我们需要好的软件配置管理？

好的软件配置管理可以产生积极的投资回报

- 更快的开发意味着产品更快的进入市场
- 更好的质量意味着减少在错误修复上的工作量，应该让工作集中在功能增强以及新需求上
- 更高的可靠性会转化为更多的正常运行时间并提高生产率



## 配置管理基础

明确什么是你想要的

- 不理解什么是你所要的注定要失败
  - ▶ 什么是软件配置管理 (SCM)?
    - “软件配置管理[SCM] 是一种包含**工具**和**过程**的软件工程工艺，开发机构可以使用它来管理其软件资产的变化” – Brian White
    - SCM 就是你如何控制软件项目的演进。



“如果你不知道你要去哪儿，你就不知道什么时候到那儿。”

## 软件配置管理基础

### 工具强化流程

- 一些现状（误区）
  - ▶ 我们实施了一个（好的）配置管理流程，因为我们使用了象 **XXX**或其他一些类似工具。
  - ▶ 我只知道用**XXX**工具，不过我真的不是很清楚该工具如何给我们带来好处。
  - ▶ 我们使用工具**Y**和**Z**通过了**CMMI 2级**评估。



## 软件配置管理基础

什么是你所需要的?

- 模糊的定义会导致失败
- 方法-明确方向、现状和实现步骤
  - ▶ 管理
    - 资产和过程的保护
    - 知道你要去哪儿?
    - 知道你在哪儿?
    - 知道什么时候到那儿?
  - ▶ 开发
    - 知道你应该做什么?
    - 是否能够工作在自己的任务上?
    - 是否可以一直跟踪你所做的事情?
    - 是否可以将稳定的工作成果交付给项目?



## “好的软件配置管理”意味着什么？

1. 安全性
2. 稳定性
3. 控制能力
4. 再现性
5. 审计能力
6. 可跟踪性
7. 可扩展性



## “好的软件配置管理”意味着什么？

1. 安全性
2. 稳定性
3. 控制能力
4. 再现性
5. 审计能力
6. 可跟踪性
7. 可扩展性

# 安全性

您遇到下面一些情况吗？

- 版本库容易损坏
- 用户可以直接对**SCM** 版本库进行危险操作
- 开发或相关人员可以绕开审计流程来进行变更
- 没有或只有有限的手段来保证版本库的安全，缺乏对未经授权的访问或修改的控制

## 好的软件配置管理的7个属性

### 1. 安全性

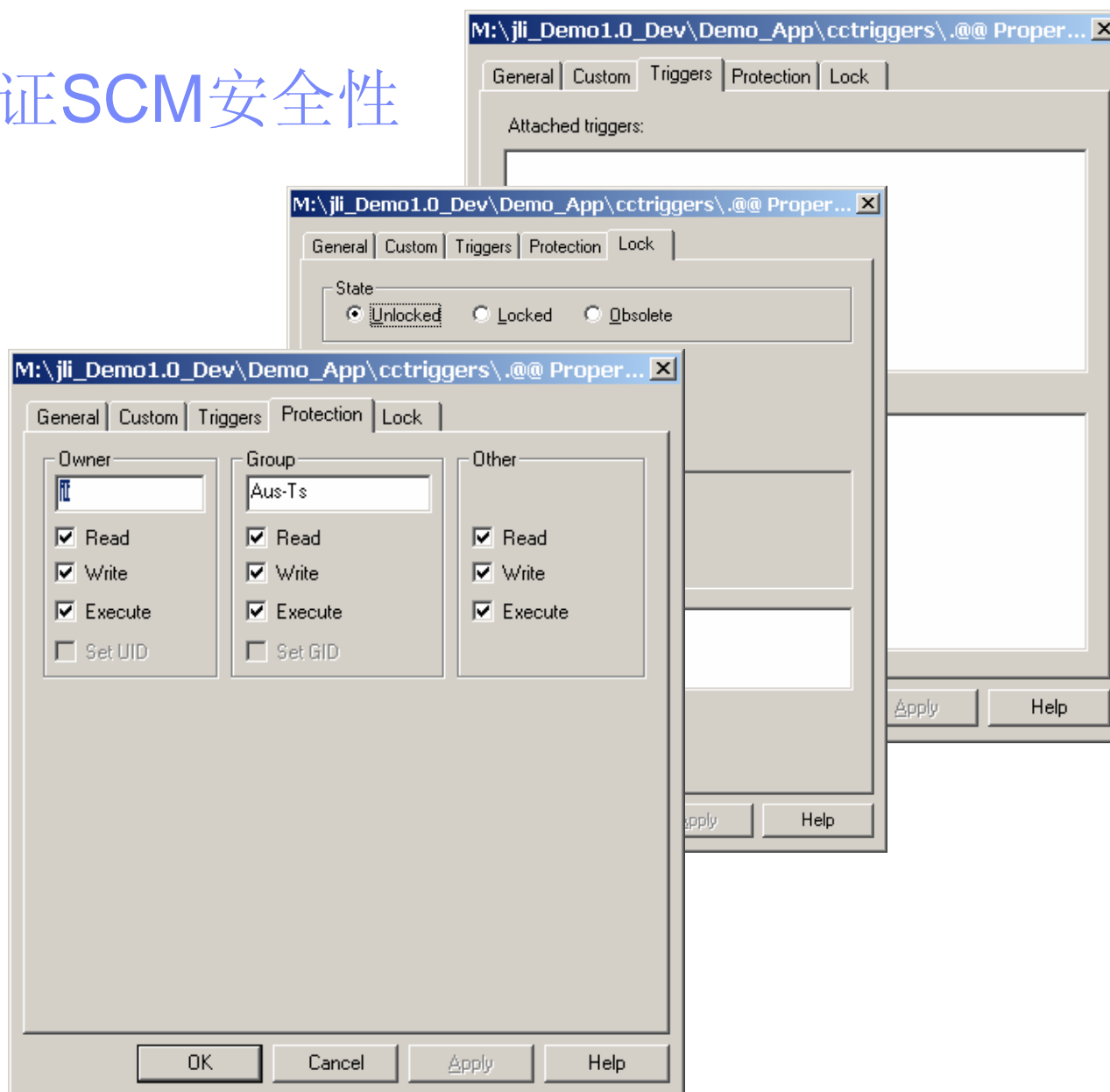
- 系统必须提供对灾难的基本保护
- 用户不应该也不能毁坏系统
  - ▶ 合理的步骤用来做合理的事
- 系统必须提供保障措施来应对未经授权的访问和/或对系统受控数据的操作



业务价值： 保护关键的资产

## ClearCase如何保证SCM安全性

- 基于实时数据库的专有版本库存储
- 基于文件系统的备份和恢复
- 利用MultiSite应对单点意外情况
- 基于版本库、目录/文件、版本控制对象等的多级权限控制
- 利用触发器对各类操作施加控制
- 支持Windows 及 Unix 互操作机制



## “好的软件配置管理”意味着什么？

1. 安全性
2. 稳定性
3. 控制能力
4. 再现性
5. 审计能力
6. 可跟踪性
7. 可扩展性

# 稳定性

您遇到下面一些情况吗？

- 开发人员不愿意或害怕检入 (**check in**) 他们的变更，因为害怕影响其他开发人员的工作或者造成项目不稳定
- 开发人员同样害怕其他人检入他们的代码
- 开发人员害怕更新他们的工作空间因为他们需要将其他人的变化同他们自己的代码立即进行集成
- 开发人员害怕更新他们的工作空间因为他们对他们将要得到的代码是否已经通过测试没有信心
- 昨天晚上开发人员离开时曾经在他们工作空间中运行良好的代码是否能在他们今天上午返回时仍然运行良好，开发人员对此没有信心

## 好的软件配置管理的7个属性

### 2. 稳定性

- 系统必须能提供稳定的开发人员/用户工作区域
  - ▶ 开发人员必须能在星期三晚上离开时确保在星期四上午（或者其他时间）回来时他们环境下的数据保持原样，除非他们同意
- 系统必须为每个用户提供不稳定因素引入时机的控制，即当用户需要时才引入
- 系统必须为每个用户提供不稳定因素引入内容的控制，即只引入用户需要的部分



“在软件项目中没有人喜欢令人惊讶的东西”

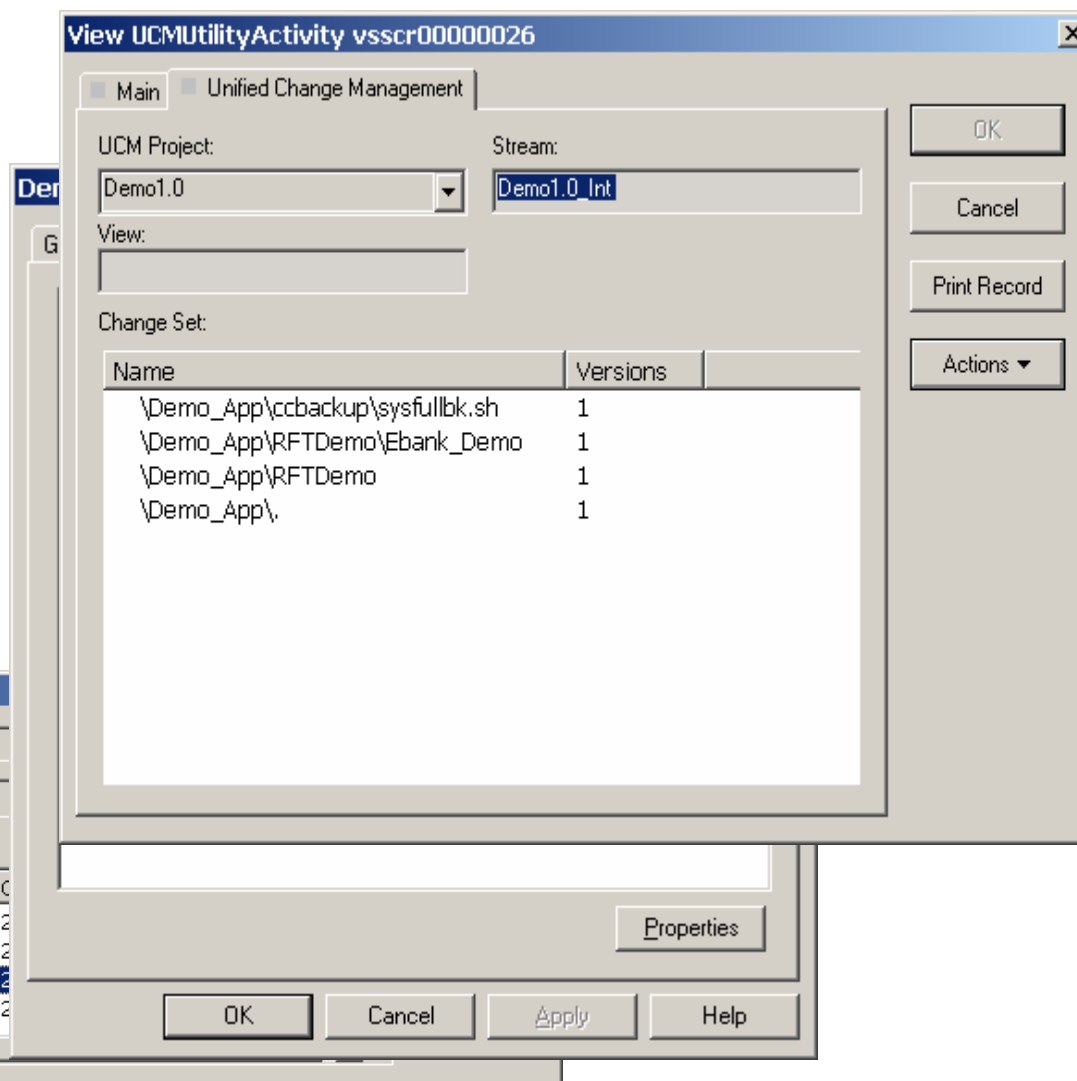
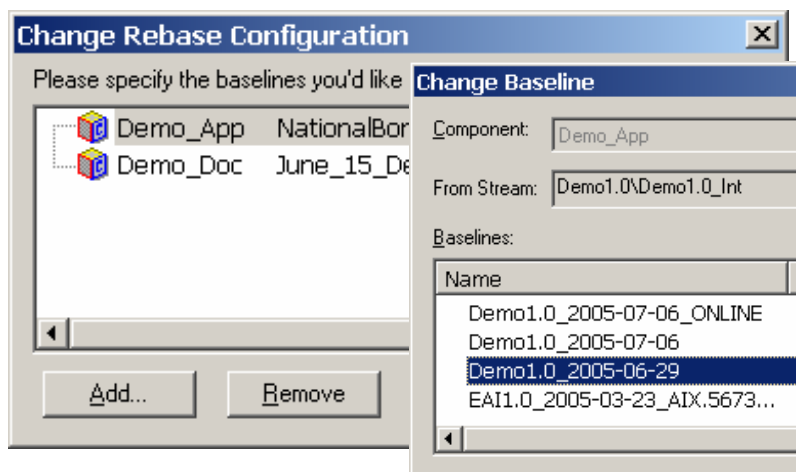
业务价值：

开发人员和团队工作效率显著增长

对项目提交成果更好的可预见性

## ClearCase如何保证SCM稳定性

- 视图（**view**）机制保证开发人员工作空间的稳定性
- 开发人员可以决定何时将集成视图（空间）下的内容引入到自己的开发视图（空间）
- 开发人员可以决定将集成视图哪一条基线的内容引入自己的开发视图中





## “好的软件配置管理”意味着什么？

1. 安全性
2. 稳定性
3. 控制能力
4. 再现性
5. 审计能力
6. 可跟踪性
7. 可扩展性

# 控制能力

您遇到下面一些情况吗？

- 开发人员无法同其他人在同一时间修改同一文件
- 开发人员无法管理其他人同一时间修改同一文件后所引入的冲突
- 开发人员无法得知所有他们当前进行的工作活动
- 开发人员无法只把稳定的或完成的工作活动交付给项目
- 开发人员记不起他们修复某个缺陷时改过的所有文件了

## 好的软件配置管理的7个属性

### 3. 控制能力

- 系统必须能提供某种机制来加强项目工作流程中适当的控制措施
- 系统必须提供某种机制来允许对内置控制机制进行扩展和定制
- 系统必须能对不同的项目不同的工作流程提供不同的控制
- 系统必须从受控流程中提供有用信息

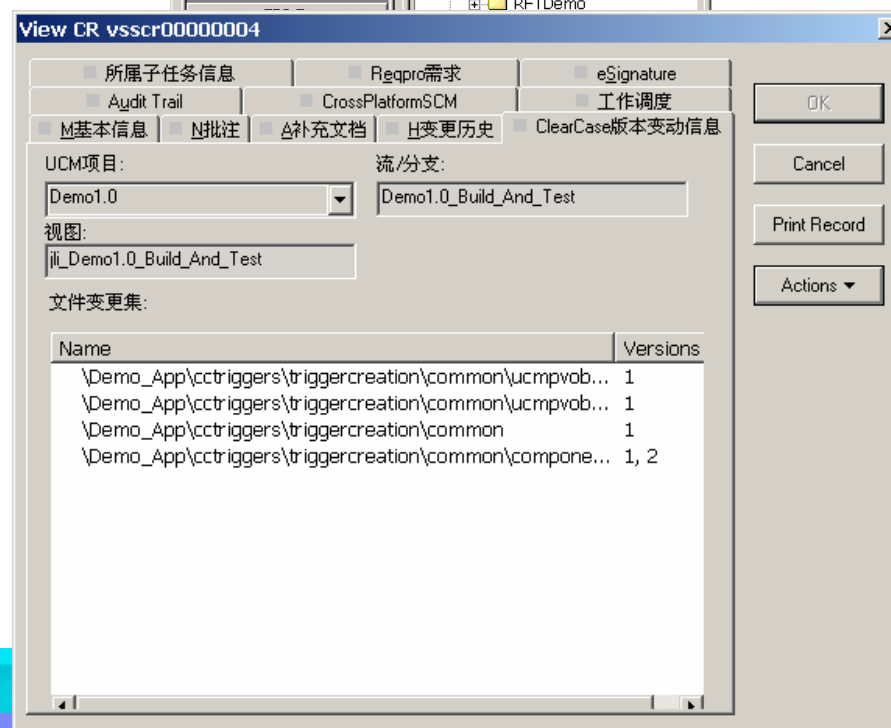
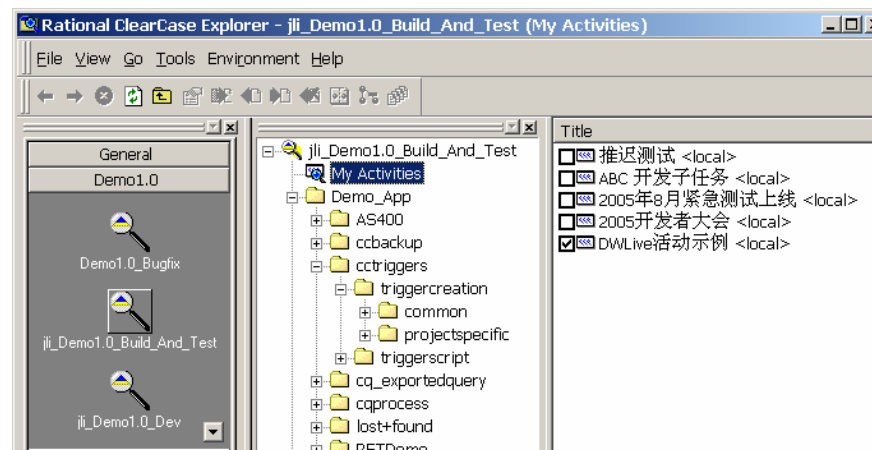


#### 业务价值:

受控的开发使得自动项目跟踪成为可能，从而获取期望结果

## ClearCase如何保证SCM控制能力

- 利用统一变更管理（UCM）加强SCM流程控制
- UCM提供了丰富的控制策略，适用于不同用户要求
- 通过“我的工作活动”提供自己的待做工作清单
- 从开发空间到集成空间活动级交付
- 活动变更集保证工件版本与相应活动的关联性



## “好的软件配置管理”意味着什么？

1. 安全性
2. 稳定性
3. 控制能力
4. 再现性
5. 审计能力
6. 可跟踪性
7. 可扩展性

## 再现性

您遇到下面一些情况吗？ **无法回到过去**

- 无法对源代码进行重新整理，使它们重新有意义
- 不能修复前一发布版本中的一个缺陷，并且让该版本与其他发布版本隔离
- 使用“备份到磁带”的方法来确保你可以重现前一发布版本
- 不能再次生成一个没有“备份到磁带”上的发布版本
- 不知道两个可执行文件之间到底什么区别

## 好的软件配置管理的7个属性

### 4. 再现性

- 系统必须能重构特定“重要”项目点的文件、目录以及命名空间的配置
- 系统必须能重构过去某个特定稳定点的文件、目录以及命名空间的配置
- 系统必须允许基于上面的配置可以继续进行开发



“如果你不能重构昨天曾构造过的但还没有测试的系统……”

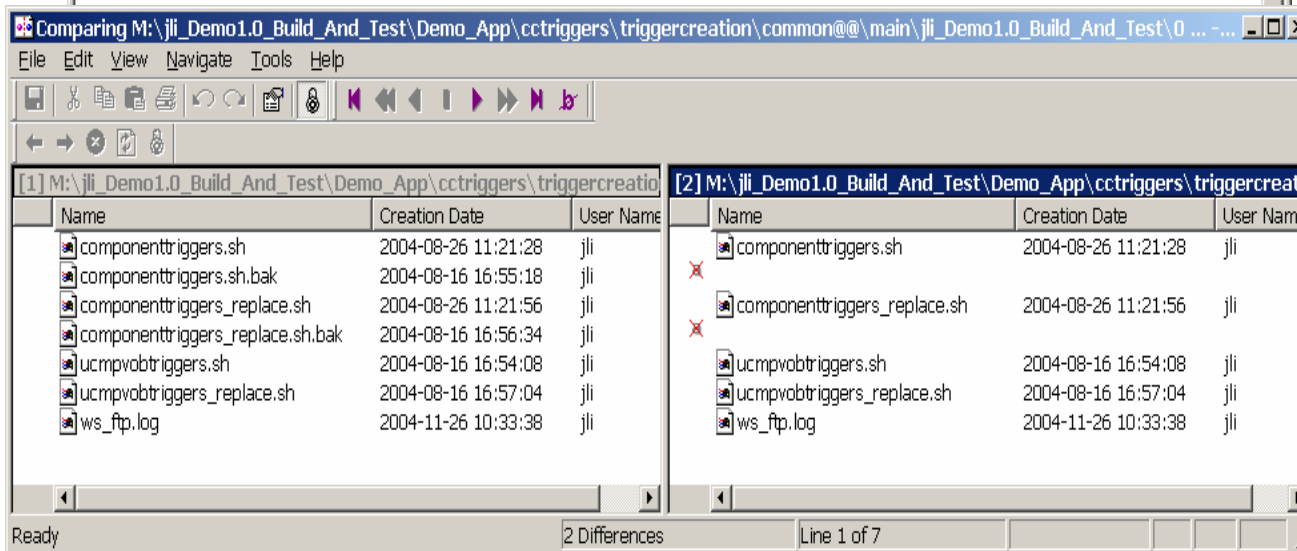
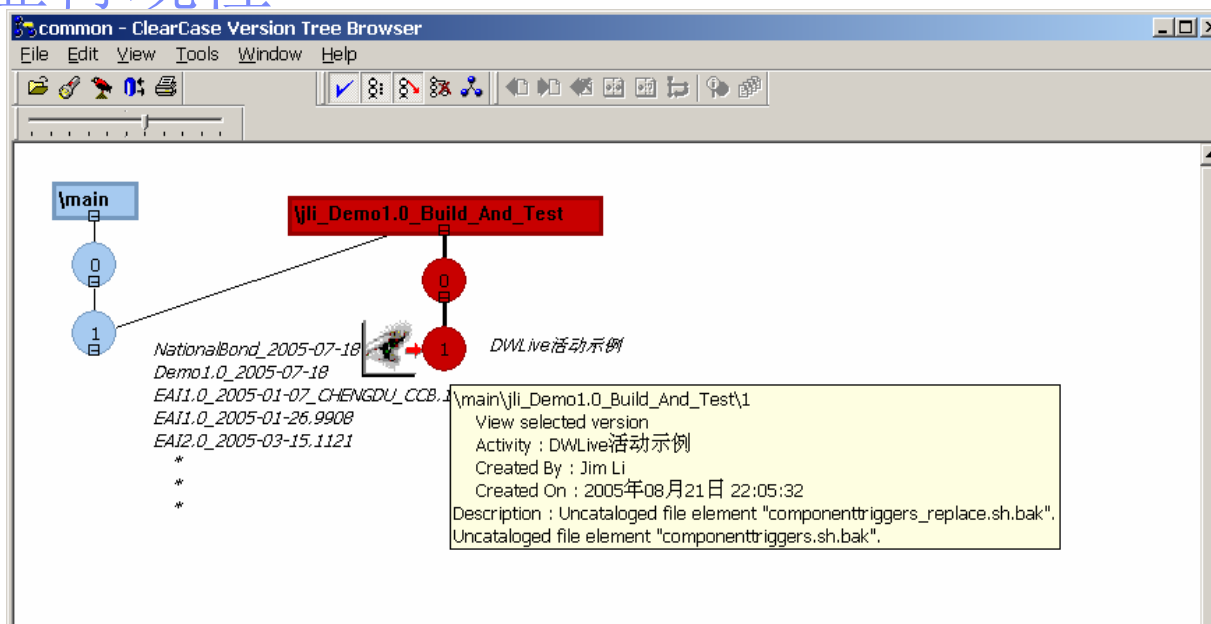
业务价值:

更好的质量

减少支持成本

# ClearCase如何保证再现性

- 目录版本控制全面记录目录命名空间变化，包括文件改名、移动、增加以及删除等
- 利用基线重构任何历史版本，包括特定目录版本下的特定文件
- 通过分支/流机制以某条历史基线为基础继续进行开发





## ClearCase如何保证再现性（续）

- 强大的构造管理机制全面记录二进制文件与参与构造的目录以及文件版本之间的关系，以及构造程序、参数等，全面保证构造的可再现性

```

C:\WINNT\system32\cmd.exe
M:\jli_mainview\Demo_App\build>ct lsdo
22-八月.10:33 "hello.exe@22-八月.10:33.1312"
22-八月.10:33 "hello.obj@22-八月.10:33.1311"

M:\jli_mainview\Demo_App\build>ct catcr -l "hello.obj@22-八月.10:33.1311"
Derived object: \Demo_App\build\hello.obj@22-八月.10:33.1311
Target hello.obj built by jli.Aus-Ts
Host "ibm-384d5f4760d" running NT 5.0 (i586)
Reference Time 22-八月-05.10:33:09, this audit started 22-八月-05.10:33:10
View was ibm-384d5f4760d:c:\ClearCase_Storage\views\IBM-384D5F4760D\jli\jli_main
view.vws [uuid d36aa030.42e54267.b1d0.78:75:4a:8c:e7:e1]
Initial working directory was M:\jli_mainview\Demo_App\build
-----
MUPS objects:
-----
directory version      \Demo_App\.\@main\4                <22-
八月-05.10:32:53>
directory version      \Demo_App\build@main\1             <22-
八月-05.10:33:09>
version                \Demo_App\build\hello.c@main\1     <22-
八月-05.10:32:56> [in makefile, primary dep]
version                \Demo_App\build\hello.h@main\1     <22-
八月-05.10:32:55>
derived object          \Demo_App\build\hello.obj@22-八月.10:33.1311 [new derive
d object]
-----
Build Script:
-----
      cl /c hello.c
-----
M:\jli_mainview\Demo_App\build>
    
```

参与编译时的目录版本

参与编译时的文件版本

编译脚本

## “好的软件配置管理”意味着什么？

1. 安全性
2. 稳定性
3. 控制能力
4. 再现性
5. 审计能力
6. 可跟踪性
7. 可扩展性

# 审计能力和可跟踪性

您遇到下面一些情况吗？

- 无法确定当前软件发布版本中的确切内容
- 不能确定不同发布版本之间的不同
- 不能确定谁做了什么
- 不能确定为什么一个特定活动要放在某个发布版本中
- 你有**ISO**、**CMM**等审计/跟踪需求



## 好的软件配置管理的7个属性

### 5. 审计能力

- 系统必须提供相应机制应对下面的问题：
  - ▶ 谁做了这一工作，什么时间做的？
  - ▶ 这次构造中包含了哪些缺陷修复/特性/功能增强？
  - ▶ 与上次构造相比，这次构造多了哪些缺陷修复/特性/功能增强？
  - ▶ 某个缺陷修复/特性/功能增强是否包含进了本次发布？
  - ▶ 文件foo.c的版本5在这次发布中吗？



业务价值：

一致性

成本

进入市场时间

## 好的软件配置管理的7个属性

### 6. 可跟踪性

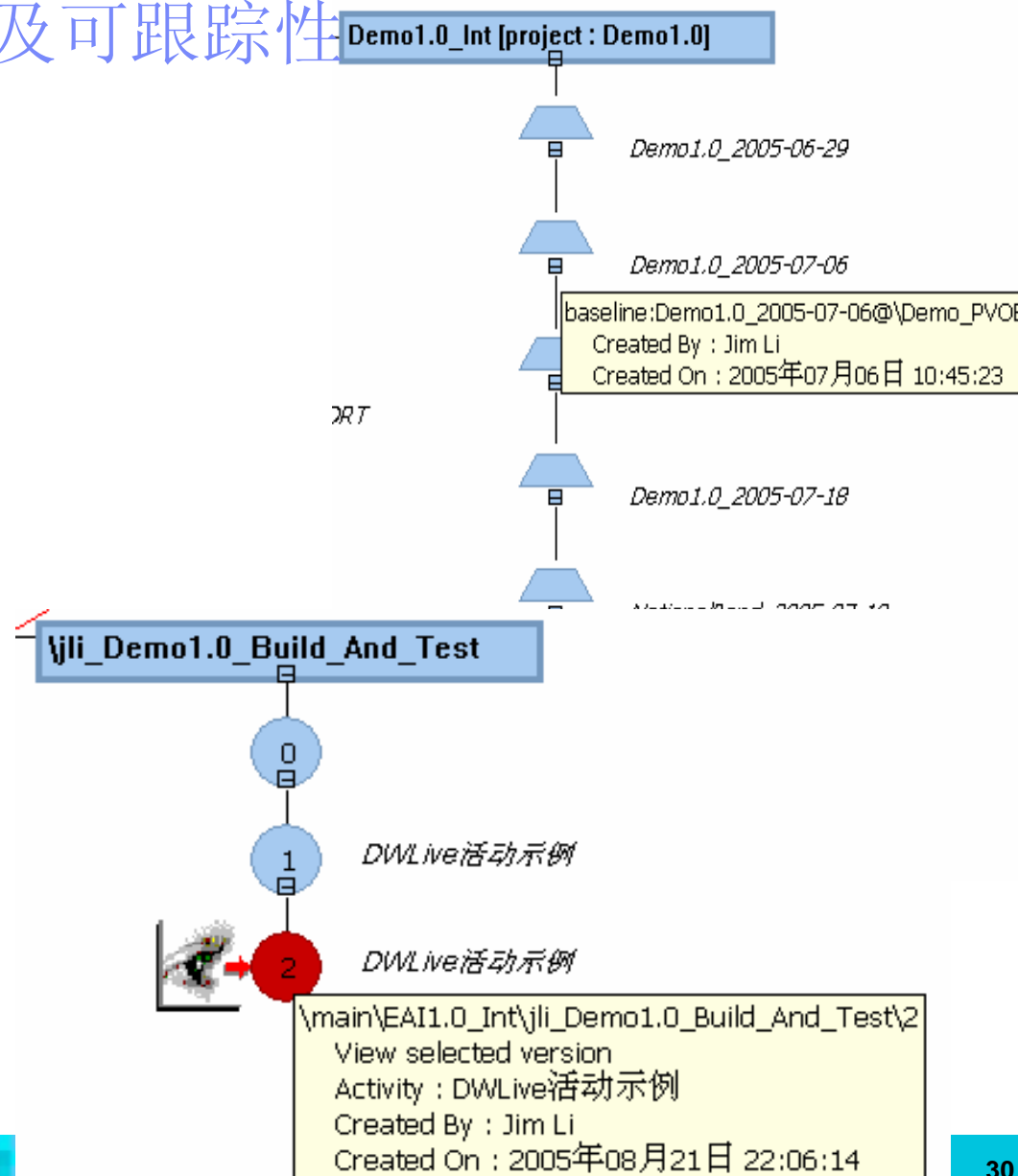
- 系统必须提供机制来识别任意机器上的软件版本
- 系统必须提供机制来确定一个发布版本的成因是什么？
  - ▶ 变更请求
  - ▶ 需求



业务价值：  
客户提交件校验

## ClearCase如何保证审计及可跟踪性

- 历史记录全面记录了谁做了这一工作，什么时间做的？在哪台机器上做的？
- 基线中包含了参与基线构成的活动，包括缺陷修复/特性/功能增强等等
- 不同基线可以同时在活动级以及版本级进行比较
- 活动变更集提供了活动与相关文件目录版本之间的关联
- 利用文件版本树提供了文件版本的可视化跟踪机制
- 通过组件基线树提供了组件基线的可视化跟踪机制
- 与ClearQuest的集成可以提供变更请求（缺陷、需求变更、增强请求等）生命周期以及相关工件变化的更为完整的跟踪过程



## “好的软件配置管理”意味着什么？

1. 安全性
2. 稳定性
3. 控制能力
4. 再现性
5. 审计能力
6. 可跟踪性
7. 可扩展性



# 可扩展性

您遇到下面一些情况吗？

- 您的项目当前规模较小，但预计会快速增长
- 您的团队过去是进行多个小项目开发，现在需要进行大项目开发协作
- 在项目的不同阶段投入不同，产出成果不同
- 您的项目参加人员分布在多个地点





## 好的软件配置管理的7个属性

### 7. 可扩展性

- 系统必须提供对不同类型和规模项目的支持，以及项目规模发生变化的支持
  - ▶ 规模可以用不同方式来度量：例如，参与者的数量，复杂程度，所管理的资产的数量和类型
- 系统必须提供项目之间的代码共享机制
- 系统必须提供地理分布团队的支持机制

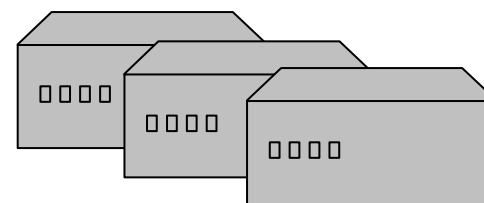


#### 业务价值：

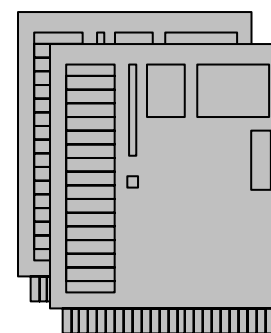
不会因项目规模或配置的变化而使用不同的工具  
整个企业范围的标准解决方案  
代码重用可以提高开发人员效率

## ClearCase如何保证可扩展性

- 可动态扩充的工作空间View服务器，在扩充时做到与客户端无关
- 可动态扩充的版本库VOB服务器，在扩充时做到与客户端无关
- ClearCase LT, ClearCase以及ClearCase MultiSite支持不同规模团队，并可以进行迁移
- 通过存储池机制，版本库VOB的存储可以动态扩充，并且可以跨不同的存储设备，从而支持超大型版本库（百G以上）
- ClearCase MultiSite可以提供地理分布团队的支持
- 支持不同平台之间的版本库迁移



跨存储设备



跨VOB 服务器



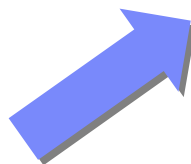
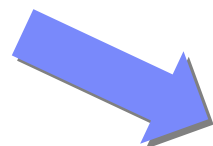
跨View 服务器



ClearCase MultiSite

## 流程是好的软件配置管理系统成功实施的关键

1. 安全性
2. 稳定性
3. 控制能力
4. 再现性
5. 审计能力
6. 可跟踪性
7. 可扩展性



要成功实施SCM，流程  
是关键!!!



IBM Software Group

## 配置管理案例分析

# 如何制定有效的配置管理流程

Rational software

IBM developerWorks **Live!**  
The Technical Conference with More  
**2005 开发者大会**

**ON DEMAND BUSINESS**

© 2005 IBM Corporation

## 客户需求：严格的质量管理流程

开发环境



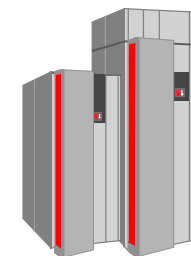
测试平台



准生产环境

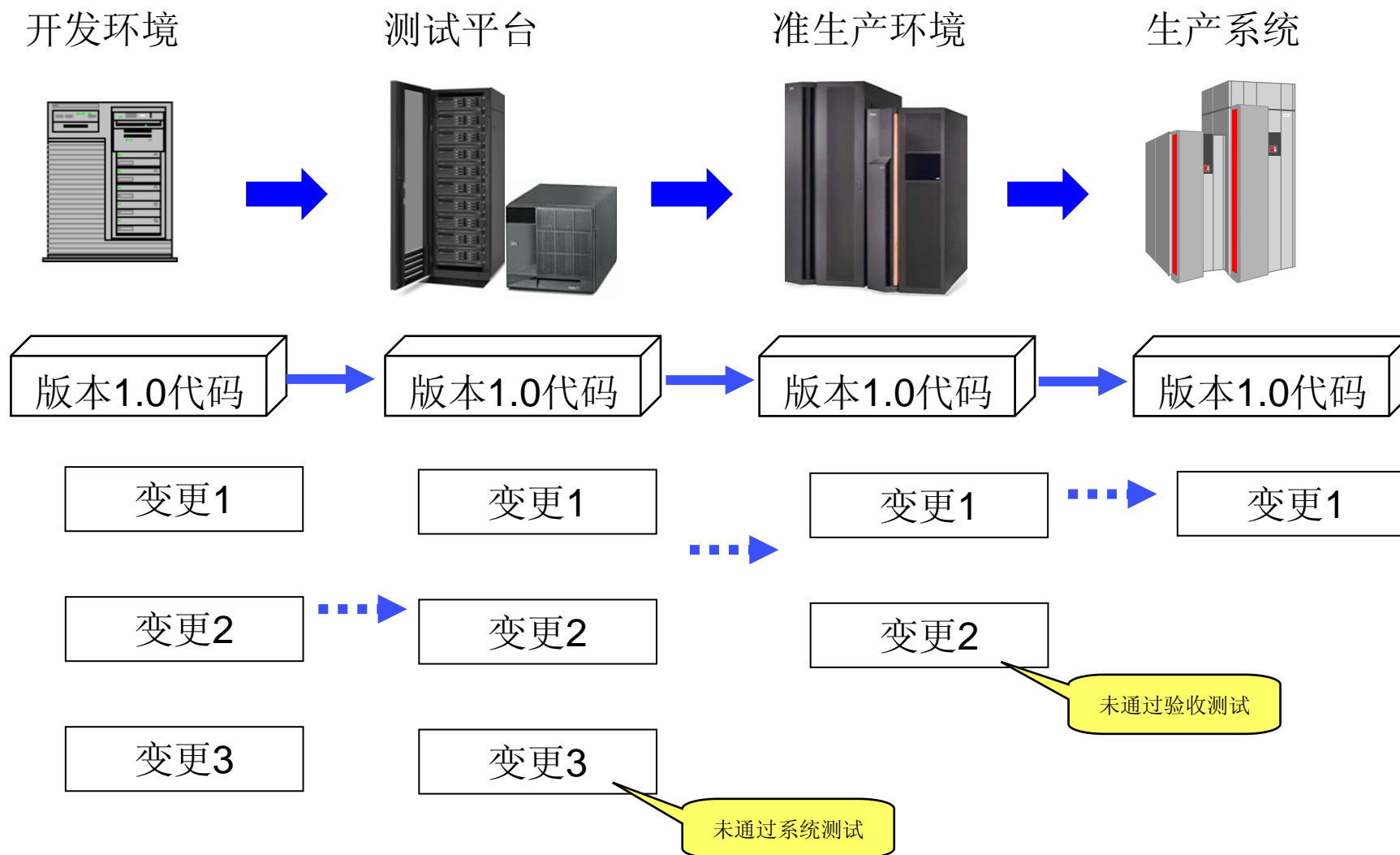


生产系统



- 对产品质量的要求很高，产品必须具有高可靠性
- 有严格的开发测试流程，每一个投产的软件产品在正式使用之前，都需要经过测试环境、准生产环境的严格测试

# 配置管理流程



# 质量陷阱一

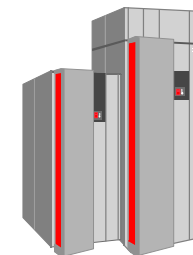
# 在生产系统上运行的是未经测试的代码?!

开发环境

测试平台

准生产环境

生产系统



版本1.0代码

版本1.0代码

版本1.0代码

版本1.0代码

变更1  
f1(v2), f2(v3)

变更1  
f1(v2), f2(v3)

变更1  
f1(v2), f2(v3)

变更1  
f1(v2), f2(v3)

变更2  
f2(v4), f3(v5)

变更2  
f2(v4), f3(v5)

变更2  
f2(v4), f3(v5)

变更3

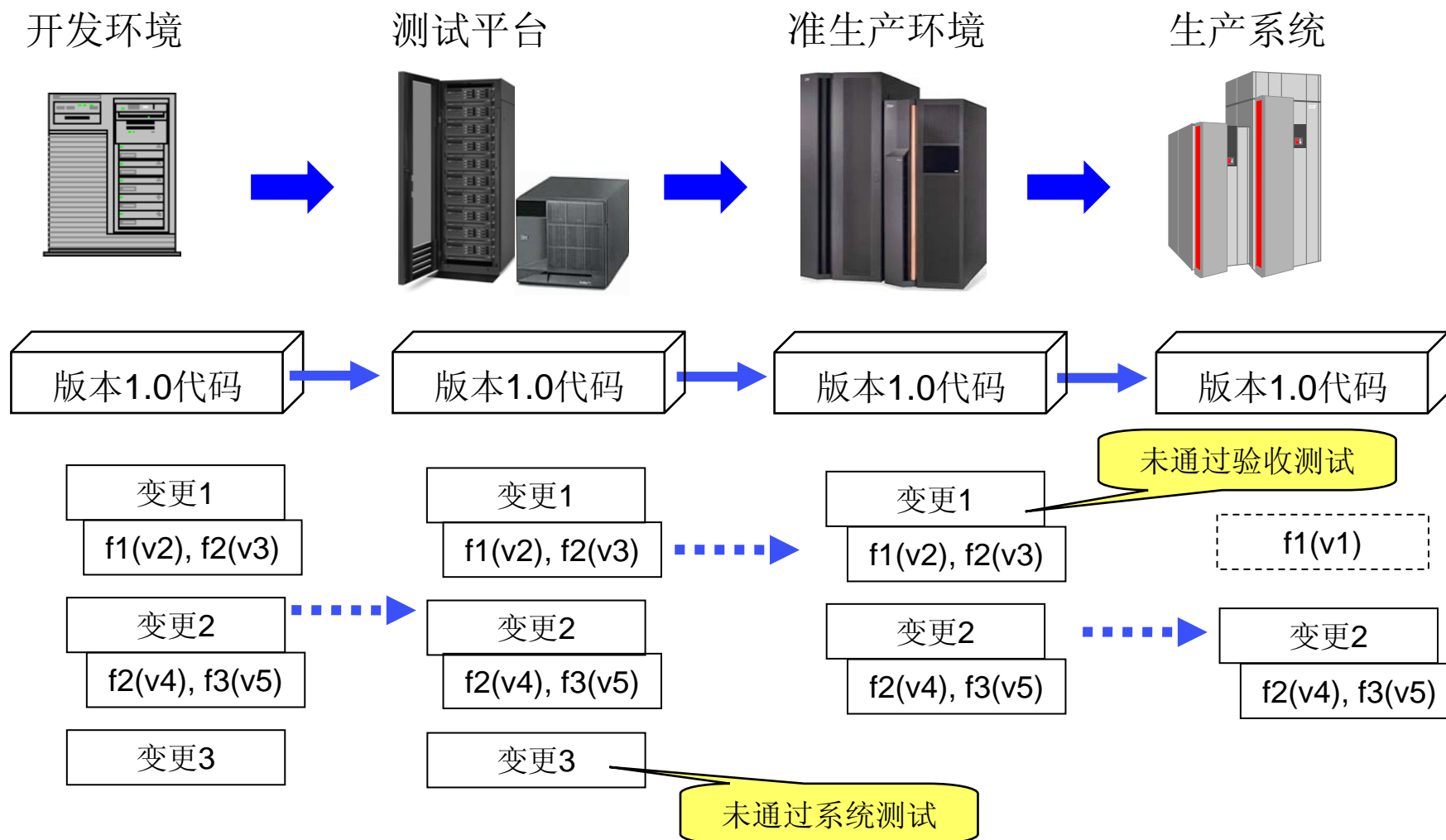
变更3

未通过系统测试

未通过验收测试

## 质量陷阱一

# 在生产系统上运行的是未经测试的代码组合?!



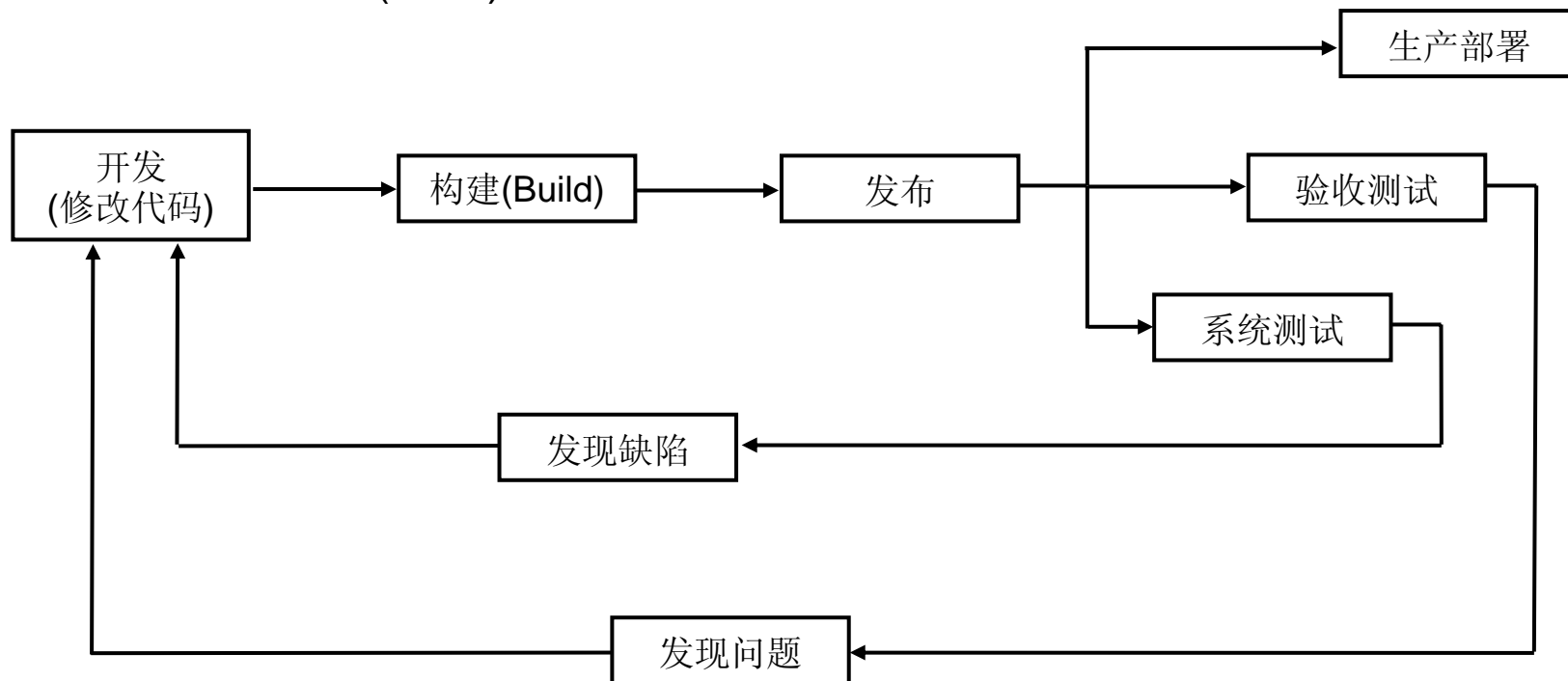


## 问题分析

- 造成问题的主要原因是变更之间的关联性，两个不同的变更修改了同一批文件
- 其他.....?

## 改进建议：闭环配置管理流程

- 建立闭环的质量保证流程
- 所有的变更都必须全部实现后一起发布
- 发布版本构建(build)而不是源代码

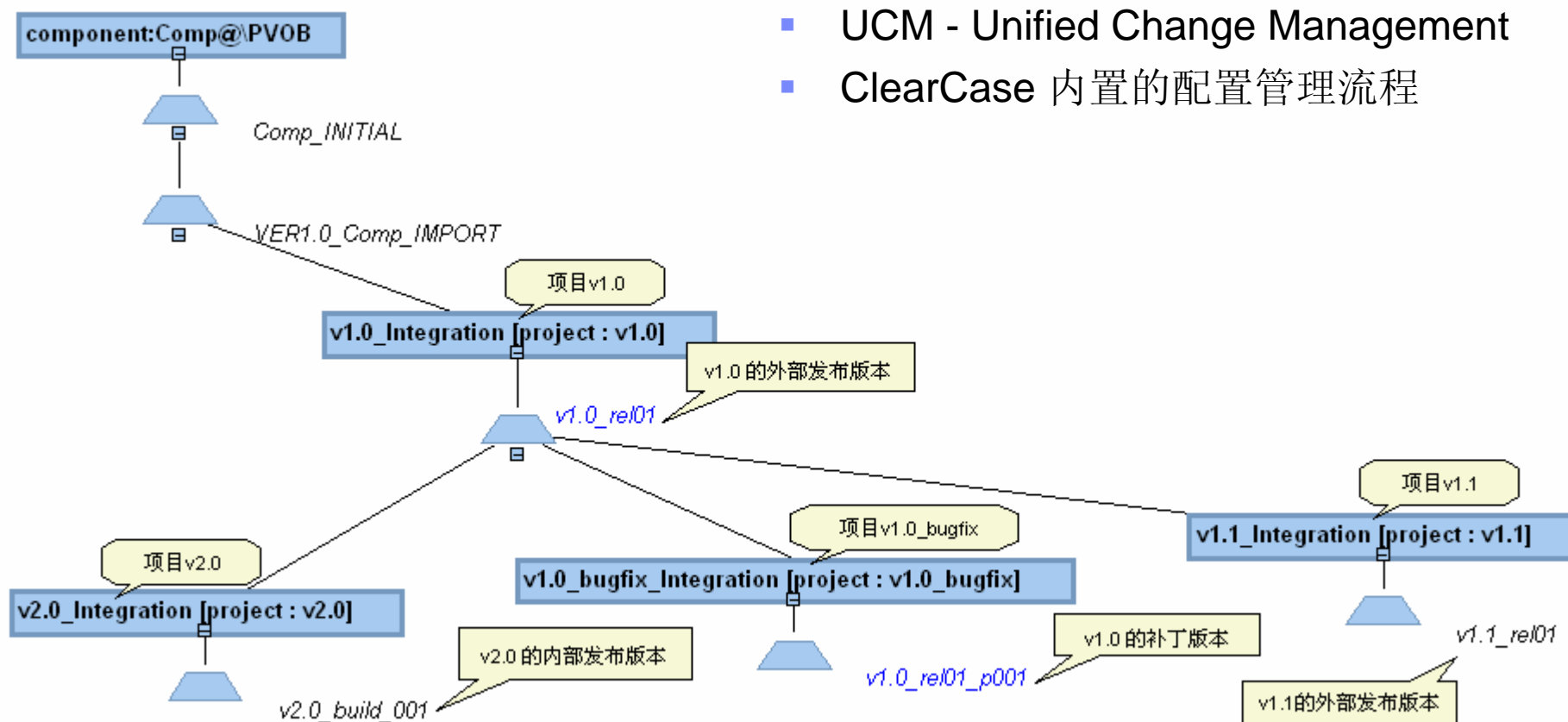


## 改进建议：区别对待缺陷和新增功能

- 把缺陷的修复工作和新增功能的开发工作区分开来：
  - ▶ v1.0中的缺陷修复
  - ▶ v1.0的新增功能版本v1.1
  - ▶ 下一个版本v2.0
- 缺陷修复和新增功能开发相互独立，保证紧急的缺陷修复不会受到新增功能的影响。

# ClearCase UCM 实现

- UCM - Unified Change Management
- ClearCase 内置的配置管理流程



# ClearCase Explorer 方便开发人员选择工作版本

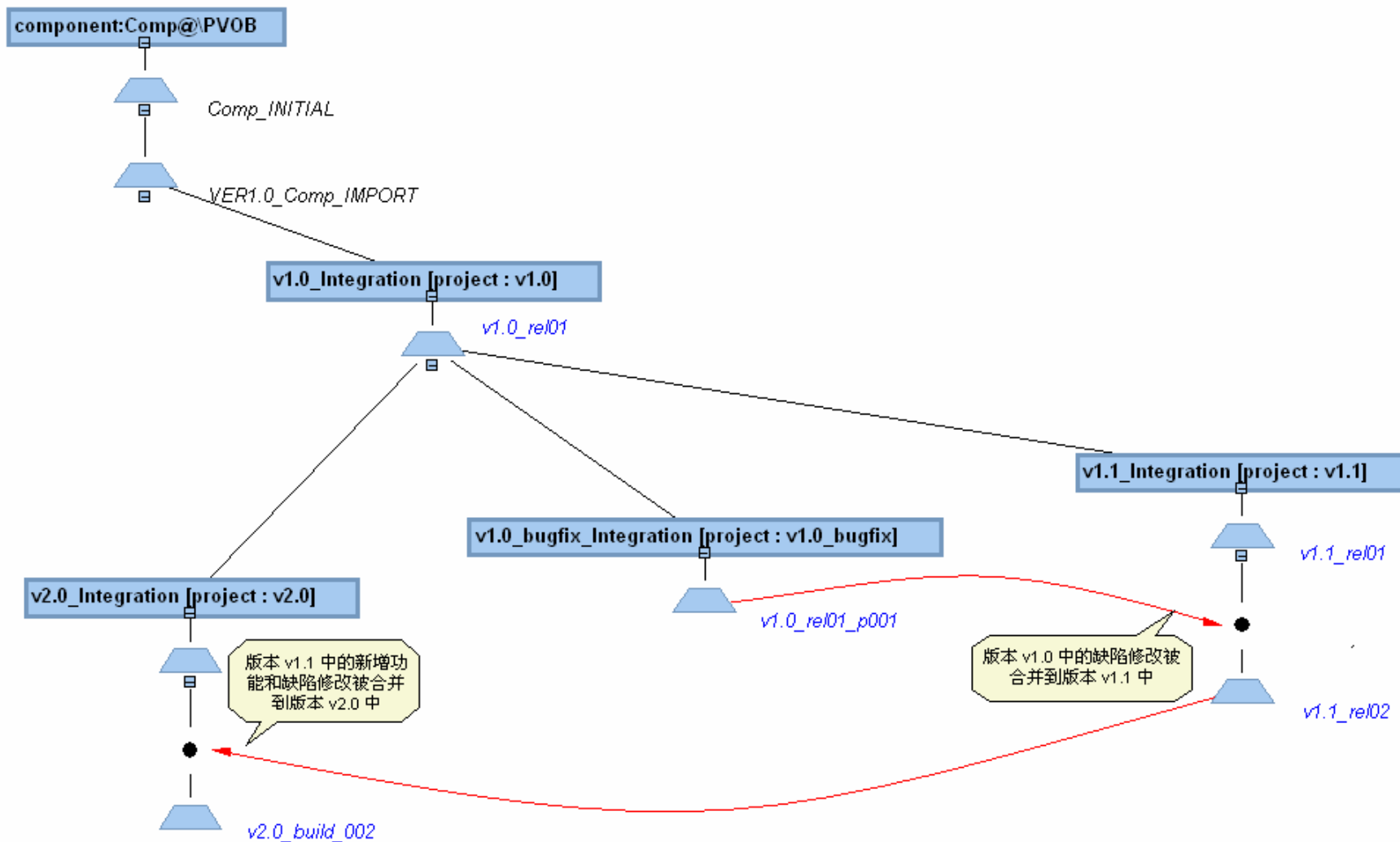
The screenshot displays the Rational ClearCase Explorer interface for a project named 'DemoUser\_v1.0\_bugfix'. The interface is divided into several panes:

- Left Pane (Views):** Shows a list of views including 'v1.0', 'v1.0\_bugfix', 'DemoUser\_v1.0\_bugfix', and 'DemoUser\_v1.0\_bugfix\_int'. Below this list are 'Views' and 'Toolbox' buttons.
- Tree View:** Shows the project structure with folders like 'My Activities', 'Comp', and 'lost+found'.
- Table View:** A table listing files and folders with columns for Name, Size, Kind, Modified, and Version.

Name	Size	Kind	Modified	Version
lost+found	0	Direct...	8/2/2005 1:58:18 ...	\main\0
file.txt	43	File El...	8/2/2005 2:04:36 ...	\main\v1.0_Int...
- UCM Dynamic Development View:** A text area providing instructions on how to use the UCM view, including details about activities, rebasing, and refreshing.

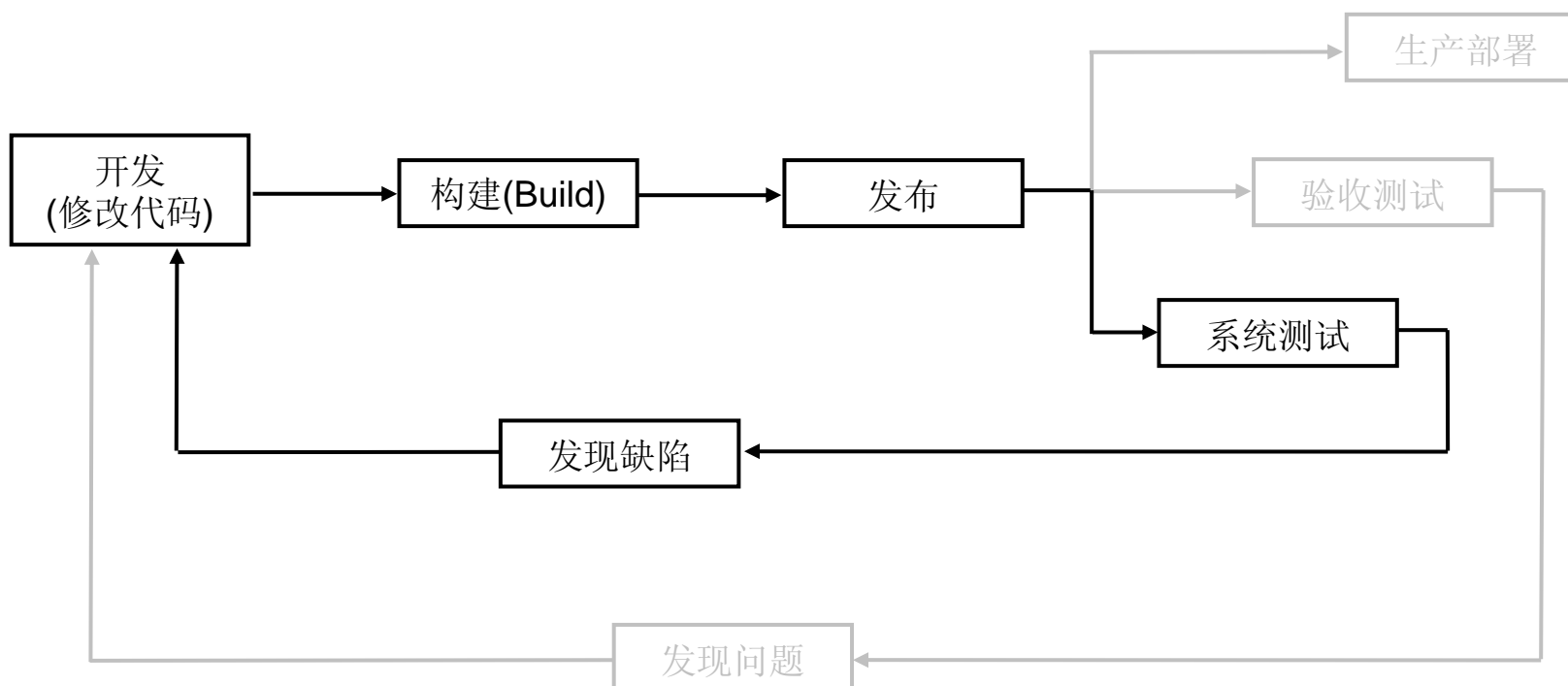
The status bar at the bottom indicates 'Ready', 'View: DemoUser\_v1.0\_bugfix', 'No Current Activity', 'Items: 2', and 'Selected: 0'.

# UCM 帮助版本之间提交变更



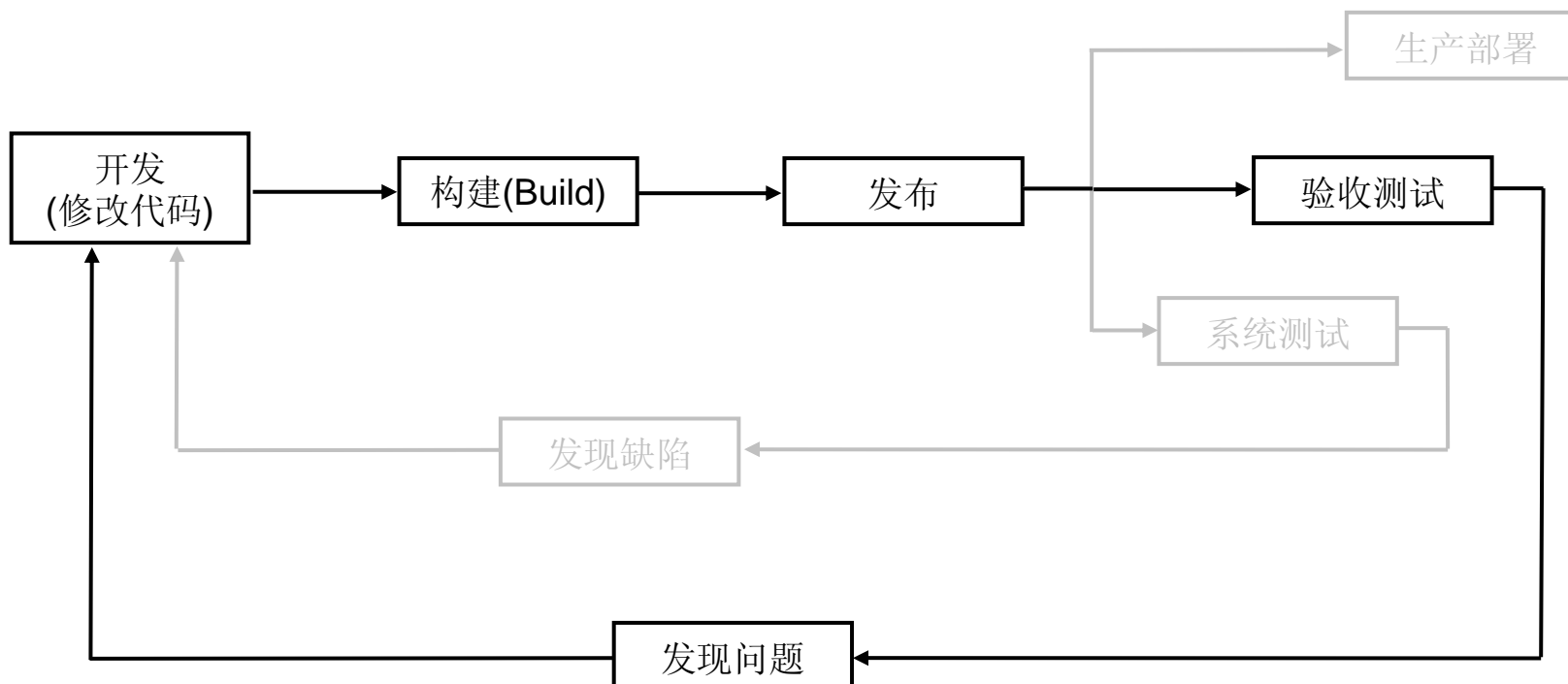
## 版本发布管理：内部发布

- 供开发团队内部测试使用，只能安装在测试平台上
- v1.0\_build\_008 表示版本v1.0开发过程中生成的第8次构建
- v1.0\_build\_008\_p001 表示上述版本的第一个补丁



## 版本发布管理：外部发布

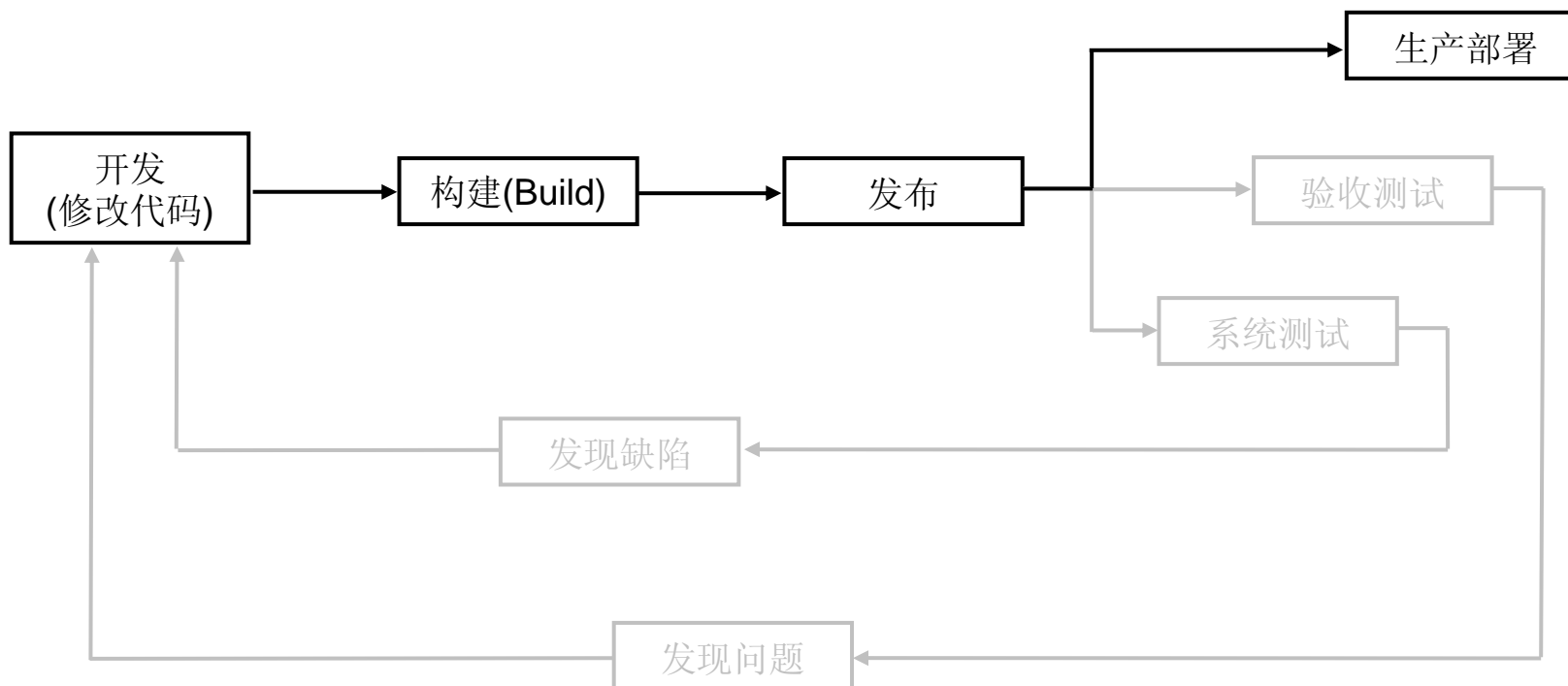
- 发布给客户的版本，可以安装到验收平台和生产系统上去
- v1.0\_rel01 表示版本v1.0的第一个外部发布版本
- v1.0\_rel01\_p001 表示针对发布版本v1.0\_rel01的第001号补丁
- v1.1\_rel02 表示版本v1.1的第二个外部发布版本





## 版本发布管理：安装部署

- 通过验收的发布版本及其所有的累积补丁全部被安装到生产系统上
- 中间可能跳过一些质量不够成熟的版本
- 只有通过用户验收测试的补丁才会被最终安装到生产系统上
- 不紧急的补丁可以累积几个以后批量安装上去



# QUESTIONS

THANK YOU