# INF 111 / CSE 121:
## Software Tools and Methods

**Lecture Notes for Fall Quarter, 2007**
**Michele Rousseau**
**Set 14**

---

## Announcements

- **Still – no lab today!**
- **Quiz #3- Next Friday 11/9**
  - All readings assigned since the last quiz
  - Plus the readings not covered on the last quiz
    - Ch 2 from "The Mythical Man-Month"
    - Van Vliet Ch. 4 will not be included on this quiz
  - Lectures from 10/31 – 11/7
- **Assignment #2 has been posted**
  - TA will cover it in lab
- **Read: Van Vliet Ch. 12**
  - Other info on UML that might be useful:
    http://atlas.kennesaw.edu/~dbraun/csis4650/A&D/UML_tutorial/
  - Argo UML Info:
    http://argouml.tigris.org/

---

## Previously in INF 111…

- **Configuration Management**

## Today's Lecture

- **Review Quiz #2**
- **CM Continued**

## The Change Process

*Customer, Developer*

Complete CRF

*CCB*

Investigate CR
*Is it Valid?*
*Is it Cost-effective?*

*Rejected* → Notify CR Owner

*Deferred* → Request more info From CR Owner

*Approved*

*CM Team*

Prepare & Schedule Changes

*Developer*

Make Changes

*QA*

Test Changes — *Fail* — *Pass* → New Version
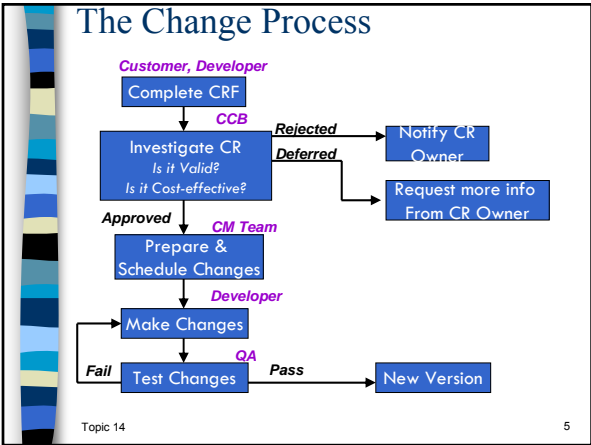
## Derivation history

- **A record of changes**
  - To a document **or**
  - code
- **Records:**
  - The change made
  - Rationale for the change
  - Who made the change
  - When it was implemented.
- **May be a comment in the code**
- **Tools can process this automatically**

## Component header information

```
// BANKSEC project (IST 6087)
// BANKSEC-TOOLS/AUTH/RBAC/USER_ROLE
//
// Object: currentRole
// Author: P. Anteater
// Creation date: 10th November 2005
//
// © Lancaster University 2002
//
// Modification history
// Vers,    Modifier    Date        Change        Reason
// 1.0   J. Cash      1/12/2006    Add header    Submitted to CM
// 1.1   E. Costello  9/4/2007     New field     Change req. R07/02
```

Topic 14                                                                7

## Version and release management

- **Determine an identification scheme to distinguish versions.**
- **Plan when a new system version will be produced.**
- **Ensure that version management procedures and tools are properly applied.**
- **Plan and distribute new system releases.**

Topic 14                                                                8

## Version identification

- **Versions should be identified in an unambiguous way**
- **There are three basic techniques for component identification**
  - Version numbering;
  - Attribute-based identification;
  - Change-oriented identification.

Topic 14                                                                9

## Version numbering

- **Simple naming scheme uses a linear derivation**
  - V1, V1.1, V1.2, V2.1, V2.2 etc.
- **Derivation structure is a tree or a network**
  - rather than a sequence
- **CONS: Names are not meaningful**
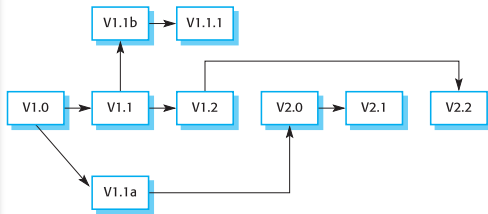- **A hierarchical naming scheme leads to fewer errors in version identification.**

## Version derivation structure

## Attribute-based identification

- **Use a combination of attributes to identify the version**
  - Examples of attributes are Date, Creator, Programming Language, Customer, Status etc.
- **More flexible than an explicit naming scheme**
- **Problem: it is difficult to keep the names unique**
  - the set of attributes have to be chosen such that the versions can be uniquely identified.
- **In practice, a version also needs an associated name for easy reference.**

## Attribute-based queries

- **Pros: Can support queries so that you can find 'the most recent version in Java' etc.**
- **The query selects a version depending on attribute values**
  - AC3D (language =Java, platform = XP, date = Jan 2003).

Topic 14                                     13

## Change-oriented identification

- **Integrates versions + changes made**
- **Used for systems rather than components.**
- **Change set**
  - describes changes made to implement the implementation
  - Then – change sets are applied in sequence
- **in principle, a version of the system that incorporates an arbitrary set of changes may be created.**

Topic 14                                     14

## Release management

- **Versions can stay internal → releases are external**
- **Releases must be…**
  - Determined by Configuration Management Team
  - Must be Validated
  - Documentation must be updated
  - → This can be expensive

  *Serious faults can force a release*

Topic 14                                     15

## New Releases

- **The more you change ➜ The more new faults introduced**
  - System reliability may be impaired

| Enhanced Release | Repair Release | Repair Release | Enhanced Release | Repair Release |

## Change Types

- **Corrective – fix faults**
- **Perfective – improve non-functional behavior**
- **Adaptive – Change functionality**

- **Don't want to mix corrective with perfective or adaptive**
  - Fix faults first!
  - Then change behavior
  - ➜ Too expensive to check if faults still apply

## System releases

- **Not just a set of executable programs.**
- **May also include:**
  - Configuration files defining how the release is configured for a particular installation;
  - Data files needed for system operation;
  - An installation program or shell script to install the system on target hardware;
  - Electronic and paper documentation;
  - Packaging and associated publicity.
- **Systems are now normally released on optical disks (CD or DVD) or as downloadable installation files from the web.**

## Release problems

- **Customer may not want a new release of the system**
  - They may be happy with their current system as the new version may provide unwanted functionality.
- **Should not assume that all previous releases have been accepted.**
  - ➔ **All files required for a release should be re-created**

## CASE tools for configuration management

- **CM processes are often standardised**
  - procedures are pre-defined
- **Lots of Docs and Data to be managed**
- **Tools make it possible**
- **Can be…**
  - Individual tools
  - Workbenches
  - Environments

## CM workbenches

- **Open workbenches**
  - Tools for each stage in the CM process are integrated
    - Includes organizational procedures
- **Integrated workbenches**
  - Provide whole-process, integrated support
    - More tightly integrated tools ➔ easier to use.
    - Less flexible in the tools used
      - Have to use tools built in

## Change management tools

○ **Change management is a procedural process**
  - it can be modelled & integrated with a version management system.
○ **Change management tools**
  - Form editor
    - supports processing the CRFs
  - Workflow system
    - define who does what
    - automates information transfer;
  - Change database
    - manages change proposals
    - linked to a VM system
  - Change reporting system
    - generates management reports (CR status)

## Version management tools

○ **Version and release identification**
  - assigns identifiers automatically for each new version
○ **Storage management.**
  - Stores the differences between versions (the delta)
    - rather than all the version code.
○ **Change history recording**
  - Record reasons for version creation.
○ **Independent development**
  - Only one version at a time may be checked out for change.
  - Parallel working on different versions.
○ **Project support**
  - Manages groups of files associated with a project
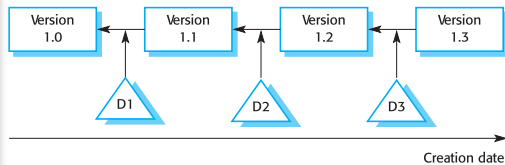    - rather than just single files.

## Delta-based versioning

## For more information

- **Read up on ch. 19 on SCCS**