

# How They Work

- Central database of source code, documentation, build tools
- Each file stored only once - all other versions are diffs of that one copy
- To Make a Change
  - Check out the latest version of a file
  - Make the changes
  - Update the database

# What should be in the database

- Source Code
- Documentation
- Build Tools
  - Often need old versions of the tools to build old versions of the software
  - Ensures software is rebuilt exactly as the customer received it
- Test Suites
- Anything else you might want later

# Version Control

- Companies ship several products from the same source base (ie Win NT and Windows 2000 versions of MS Office)
- When tracking down bugs you want to examine the code as it was when the product shipped

# Code Sharing

- Multiple people can work on the same source base without colliding
  - (1) Locks individual files so only one person at a time can modify it \*OR\*
  - (2) Allows multiple people to modify a source file and the system will automatically merge the changes (usually)

# Locking

- Only one person can work on a file at once
- Works fairly well if developers work on different areas of the project and don't conflict often
- Problem 1: People forget to unlock files when they are done
- Problem 2: People work around locking by editing a private copy and checking in when the file is finally unlocked - easy to goof and lose changes

# Merging

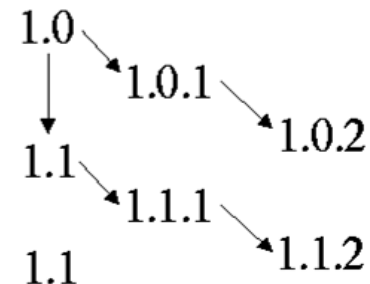
- Several people can work on a file at once
- Before committing changes, each user merges their copy with the latest copy in the database
  - This is normally done automatically by the system and usually works, but you should not blindly accept the result of the merge

# Labelling

- Label all the files in the source base that make up a product at each milestone
- Just before and just after a major change (eg. changing several interfaces)
- When a new version ships

# Version Trees

- Each file in the database has a version tree
- Can branch off the version tree to allow separate development paths
- Typically a main path (trunk) for the next major version and branches off of shipped versions for maintenance





# Branching

- When a new version ships, typically create a branch in the version tree for maintenance
- Double update: fix a defect in the latest version and then merge the changes (often by hand) into the maintenance version
- Also create personal versions so you can make a change against a stable source base and then merge in the latest version later