<u>Everything on Version management</u>

The configuration database contains information about system versions/configurations.

A system version is any instance of the system that differs in some way from others; this includes bug fixes, added features, etc.

A system release is a version of the system that is distributed to clients.

There are more versions than releases.

Configuration management database identifies particular version of components in some way; doesn't contain components themselves.

The actual components are managed by a version management or version control system (**VCS**). This can be integrated to database or separate system.

VCS must refer to version; the most common way being hierarchical numbering.

<u>Version numbering</u>

A component/system's version has format X.Y.Z

X	Major number Incremented when release contains significant changes
Y	Minor number Increased when minor features are introduced
Z	Patch number Increased due to small bug fixes

This approach is OK for sequential development.

In practice, a more elaborate scheme is needed; branching.

Branching: A duplication of an object under revision control so that modifications can happen in parallel along both branches.

A parent branch might have child branches; development may continue along the main trunk as well.

Branches, often merged back into parent or main trunk.

Forking: Branch not intended to be merged
Forking allows experimentation with changes or developing a variant system without affecting the original project.

Version management

VCS tools control a repository where the different versions of components are held. The versions in the repo are immutable.

The VCS can report complete change histories for components and report the differences between versions.

VCS tools apply access control mechanisms:

A programmer checks a component version out of the repo, works on it and then commits the update

The original version is not discarded

A component might be locked whilst checked-out; prevents accidental second checkout followed by parallel updates

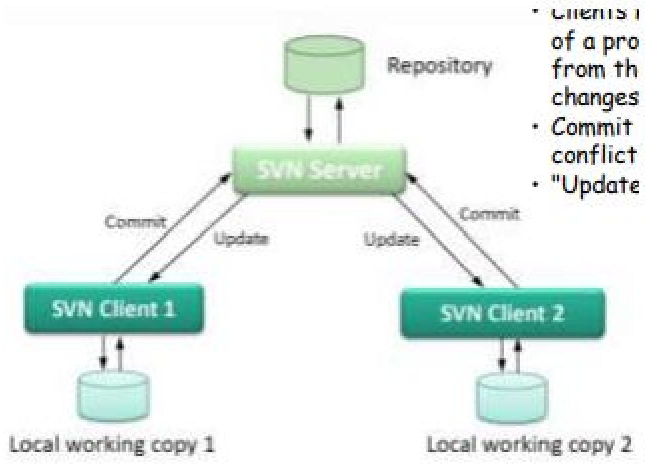
Multiple checkouts might be allowed with merging and conflict resolution occuring at check-in.

Can also request a copy of a version without checking-out. These copies may not be checked in again.

Typically this is administered as a distributed system:

Repository on a central server

Programmers at their workstations



Server organizes central repo; clients keep a local copy of a project and can fetch updates from repo and commit changes to it.

Commits may merge and trigger conflict resolution; Update = Checkout.

CVS Examples:

1. Apache Subversion
2. GitHub
3. Subclipse