CVS Quick Reference

(orientiert an Steven Pemberton's Introduction)

Terminologie

CVS ist ein Version Control System, mit dem mehrere Personen gleichzeitig auf mehreren Dateien (oder auch Dateigruppen) arbeiten können. Es basiert intern auf RCS und diversen Diff–Tools. Im Gegensatz zu RCS benutzt CVS *kein* Locking von Dateien! Mit CVS lassen sich ganze Verzeichnisbäume leichter verwalten als mit RCS.

Das **Repository** ist der zentrale Aufbewahrungsort für die von CVS verwalteten Dateien. Sie werden dort in der jeweils letzten Version aufbewahrt, basierend auf der Grundversion und allen Veränderungen, die seitdem gemacht wurden. Zusammengehörige Dateien nennt man auch **Module**, im einfachsten Fall ist ein Modul ein Unterverzeichnis im Repository.

Jede von CVS verwaltete Version einer Datei erhält eine **Revision Number**. Es damit möglich, alle bisherigen Versionen einer Datei zu restaurieren. Es lassen sich Abzweigungen (**Branches**) vom eigentlichen Versionsstamm einfügen, so daß ein Versionsbaum entsteht. Die so entstandenen Zweige lassen sich auch wieder zu einem Versionsstamm zusammenfügen. Mehrere Dateien (verschiedener Versionen) können mit einem **Tag** versehen und somit zusammengehörig markiert werden (z.B. für ein Release eines Softwareprodukts). Genauere Erläuterungen zu diesen Themen sind im CVS–Manual zu finden.

Das **Working Directory** ist der „Arbeitsplatz“ des Benutzers. Hierher werden Kopien der Dateien (oder der Module) aus dem Repository kopiert (**Personal Copy**) und können bearbeitet werden. Das Arbeiten direkt im Repository ist *tabu*!

Es ist mit einem **Update** möglich, die Dateien des Working Directory immer auf dem neuesten Stand des Repository zu halten. Sollten Veränderungen an den Dateien des Working Directory vorgenommen und gleichzeitig von anderen Benutzern neue Versionen in das Repository geschrieben worden sein, versucht CVS beide Änderungen in der Personal Copy mit einem **Merge** automatisch zusammenzuführen. Treten dabei Konflikte auf, muß der Benutzer diese manuell auflösen. Dabei werden diese Konflikte in den Dateien visuell verdeutlicht.

Um Veränderungen der Dateien des Working Directory in das Repository zu übernehmen, wird vom Benutzer ein **Commit** ausgeführt. Arbeitete der Benutzer nicht auf der letzten Version der Datei, muß zunächst ein Update durchgeführt werden (mit den oben beschriebenen Konsequenzen). Die Versionen der Dateien im Repository werden anschließend inkrementiert.

Anwendungsbeispiele

Die Parameter `[Datei]` bedeutet, daß die Angabe einer oder mehrerer Dateien erforderlich ist. Wird er weggelassen, wirkt sich das Kommando auf alle Dateien — auch in Unterverzeichnissen — aus. Nahezu alle CVS–Befehle wirken sich auch auf Unterverzeichnisse aus!

Einfügen bestehender Quelldateien in das Repository

Der aktuelle Linux–Kernel soll in einem Unterverzeichnis des Repository abgelegt werden.

```
# cd /usr/src/linux-2.0.23
# cvs import linux-2.0.23 LINUX INITIAL
```

Das aktuelle Verzeichnis und alle Unterverzeichnisse werden im Repository in das Unterverzeichnis `linux-2.0.23` geladen. Das sog. Vendor–Tag und Release–Tag (die letzten beiden Parameter) sind erforderlich, Bedeutung siehe CVS–Manual.

Erstellen einer Kopie des Repository im Working Directory

Der importierte Linux–Kernel soll nun in das Benutzerverzeichnis kopiert werden. Der Kernel befindet sich im Verzeichnis `linux-2.0.32` im Repository.

```
# cd $HOME/src
# cvs checkout linux-2.0.32
```

Es wird ein Verzeichnis `linux-2.0.32` angelegt, darin befinden sich die Kernel–Sourcen.

Erneuern der Kopie im Working Directory

```
# cvs update [Datei]
```

Alle Änderungen, die seit dem Checkout der Dateien im Repository vorgenommen wurden, werden in das Working Directory übernommen (Merge). Es wird dabei angezeigt, welche Dateien betroffen sind. Konflikte mit etwaigen lokalen Änderungen werden visuell in den Dateien des Working Directory angezeigt und müssen manuell aufgelöst werden.

Veränderungen in das Repository zurückschreiben

```
# cvs commit [Datei]
```

Alle Änderungen werden in das Repository übernommen. Der Benutzer wird dazu aufgefordert, seine Änderungen in einem Editor zu kommentieren, diese Informationen werden ebenfalls abgespeichert und können später ausgelesen werden.

Hinzufügen und Löschen von Dateien im Working Directory

Um eine neue Datei in das Repository einzufügen oder daraus zu löschen, werden die folgenden Kommandos benutzt.

```
# echo LIVE > <Datei>
# cvs add <Datei>
```

und

```
# rm <Datei>
# cvs remove <Datei>
```

Um die Veränderungen tatsächlich auf das Repository wirken zu lassen, muß schließlich ein Commit ausgeführt werden. Gelöschte Dateien lassen sich für alte Versionen wiederherstellen!

Aufräumen des Working Directory

```
# cvs release -d <Working_Dir>
```

Anstatt das Working Directory mit `rm -r <Working_Dir>` zu löschen, sollte man Release benutzen, um sicherzustellen, daß alle Modifikationen eingecheckt wurden.

Weitere Möglichkeiten und Tips

Kommentare zu den Versionen ansehen:

```
# cvs log [Datei]
```

Status von Dateien anzeigen lassen (Versionen, Tags, etc.):

```
# cvs status [Datei]
```

Unterschiede zwischen Version im Working Directory und Repository anzeigen:

```
# cvs diff [Datei]
```

Datei umbenennen mit Rücksicht auf ältere Versionen:

```
# mv <Datei> <Datei_neu>
# cvs add <Datei_neu>
# cvs remove <Datei>
```

Änderungen an Datei im Working Directory verwerfen:

```
# rm <Datei>
# cvs update <Datei>
```

Ausführliche Informationen

CVS–Manual (PostScript) unter `/usr/local/doc/cvs/cvs.ps`
Manual Pages mit `man cvs`, Info Pages mit `info cvs`