Tipps und Troubleshooting

Autor: Karl Fogel ()
Layout: Matthias Hagedorn (*matthias.hagedorn@selflinux.org*)

Der folgende Text enthält das Kapitel 8 der deutschen Übersetzung des Buches "Open Source Development with CVS", welche unter der GNU Public License veröffentlicht wurden. Die Originalversion dieses Textes ist unter http://cvsbook.redbean.com/cvsbook.html erhältlich. Wir haben an dieser Stelle das Inhaltsverzeichnis den tatsächlich enthaltenen Kapiteln angepaßt.

Das SelfLinux-Team

Inhaltsverzeichnis

1 Wenn mal was schief läuft

Ich habe schon in den vorangegangenen Kapiteln betont, dass CVS keine **Blackbox**-Software ist. In eine **Blackbox** kann man nicht hineinschauen, sie bietet keinen Zugriff auf die inneren Vorgänge, sodass man nichts bereinigen kann (aber auch nichts durcheinander bringen). Die Prämisse ist die, dass bei einer **Blackbox** normalerweise nichts bereinigt werden muss. Die Software sollte die meiste Zeit über perfekt laufen, daher brauchen sich die Anwender nicht um die inneren Vorgänge zu kümmern. Wenn eine **Blackbox** ausfällt, so geschieht das jedoch meist gründlich. Jedes Problem bedeutet den Totalausfall, da man nicht viele Möglichkeiten hat, etwas zu reparieren.

CVS ist eher eine völlig transparente Box - nur ohne die Box. Die beweglichen Teile sind der Umwelt direkt ausgesetzt - nicht etwa hermetisch abgeschlossen -, und Teile der Umwelt (unerwartete Dateizugriffs echte, abgebrochene Kommandos, konkurrierende Prozesse, was auch immer) können manchmal in den Mechanismus gelangen und sozusagen das Uhrwerk blockieren. Aber selbst wenn CVS nicht immer perfekt läuft, so fällt es auch fast nie komplett aus. Es hat den Vorteil des kontrollierten Leistungsrückgangs, und der Grad, zu dem es nicht perfekt läuft, ist für gewöhnlich proportional zur Anzahl und Schwere der in seiner Umgebung auftretenden Probleme. Wenn man genug darüber weiß, was CVS zu tun versucht - und wie es das versucht -, wird man wissen, was im Problemfall zu tun ist.

Obwohl ich hier natürlich nicht alle Probleme aufzählen kann, an die Sie geraten können, so habe ich doch versucht, die häufigsten hier abzudecken. Das Kapitel ist in zwei Teile aufgeteilt: Der erste beschreibt die Umgebungseinflüsse, auf die CVS am empfindlichsten reagiert (hauptsächlich Zugriffsrechte im Archiv und die der Verwaltung dienenden Teile der Arbeitskopie), und der zweite Teil behandelt einige der am häufigsten anzutreffenden Probleme und deren Lösung. Indem Sie sich damit befassen, wie man diese Standardsituationen meistert, werden Sie auch ein Gefühl dafür bekommen, wie man an unerwartete Probleme im CVS herangeht.

2 Die üblichen Verdächtigen

Als CVS-Administrator (oder Notarzt) merkt man schnell, dass 90 Prozent der Probleme, welche die Anwender haben, durch Inkonsistenzen in deren Arbeitskopien und die übrigen 90 Prozent durch falsche Zugriffsrechte im Archiv verursacht werden. Deshalb werde ich, bevor wir uns den einzelnen Situationen widmen, einen kurzen Überblick über den administrativen Teil der Arbeitskopien geben und einige wichtige Tatsachen über die Zugriffsrechte ansprechen

2.1 Der administrative Teil der Arbeitskopie

In Kapitel 2 haben Sie schon die Grundlagen des Aufbaus einer Arbeitskopie kennen gelernt; in diesem Abschnitt werden wir ein wenig mehr ins Detail gehen. Dabei geht es hauptsächlich um die Dateien in den administrativen Unterverzeichnissen, die den Namen `CVS/` tragen. `Entries`, `Root` und `Repository` haben Sie bereits kennen gelernt, doch in `CVS/` können sich auch noch andere, von der Situation abhängige Dateien befinden. Diese Dateien werde ich jetzt beschreiben, teils, damit sie Sie nicht überraschen, wenn Sie sie antreffen, teils, damit Sie von ihnen verursachte Probleme lösen können.

`CVS/Entries.Log`

Gelegentlich wird eine Datei namens `CVS/Entries.Log` auf mysteriöse Weise erscheinen. Der einzige Zweck dieser Datei ist es, kurzzeitig kleinere Änderungen an `CVS/Entries` zwischenzuspeichern, und zwar so lange, bis eine Operation ansteht, die so wichtig ist, dass es sich lohnt, die gesamte Datei `Entries` neu zu schreiben. CVS kann Änderungen in der Datei Entries nicht an Ort und Stelle vornehmen, sondern muss, um eine Änderung vorzunehmen, die ganze Datei einlesen und wieder zurück schreiben. Um das zu vermeiden, vermerkt CVS manchmal kleine Änderungen in `Entries.Log`, bis es `Entries` das nächste Mal neu schreiben muss.

Das Format von `Entries.Log` ist das gleiche wie das von `Entries`, bis auf einen zusätzlichen Buchstaben am Anfang jeder Zeile. **A** bedeutet, dass die Zeile der Datei `Entries` hinzuzufügen ist, und **R** heißt, dass die Zeile entfernt werden muss.

Sie können `Entries.Log` fast immer ignorieren, es kommt sehr selten vor, dass ein menschliches Wesen deren Inhalt verstehen muss. Wenn Sie jedoch `Entries` durchgehen, um ein Problem in der Arbeitskopie zu beheben, dann sollten Sie auch `Entries.Log` untersuchen.

`CVS/Entries.Backup`

Die Datei `CVS/Entries.Backup` ist der Ort, wohin CVS die neue `Entries`-Datei schreibt, bevor es sie in `Entries` umbenennt. (Ähnlich wie es erst temporäre `RCS`-Dateien in das Archiv schreibt und sie erst, wenn sie fertig gestellt sind, an den vorgesehenen Ort verschiebt und umbenennt.) Da die Datei `Entries.Backup` zu `Entries` wird, werden Sie selten eine Datei namens `Entries.Backup` zu Gesicht bekommen; wenn doch, dann bedeutet das vermutlich, dass CVS inmitten einer Operation abgebrochen wurde.

`CVS/Entries.Static`

Wenn eine Datei `CVS/Entries.Static` existiert, dann heißt das, dass nicht das ganze Verzeichnis aus dem Archiv geholt wurde. (Wenn CVS weiß, dass die Arbeitskopie in einem unvollständigen Zustand ist, wird es keine zusätzlichen Dateien in das Verzeichnis bringen.)

Die Datei `Entries.Static` ist während eines `Checkout` oder eines `Update` vorhanden und wird sofort entfernt, wenn die Operation abgeschlossen ist. Wenn Sie also eine Datei `Entries.Static` sehen, dann heißt das, dass CVS unterbrochen wurde und dass deren Vorhandensein CVS vom Anlegen neuer Dateien in der Arbeitskopie abhält. (Oftmals löst der Aufruf von `cvs update -d` das Problem und entfernt `Entries.Static`.)

Bemerkung

Das Nichtvorhandensein von `Entries.Static` bedeutet nicht automatisch, dass die Arbeitskopie alle Dateien des Projektes enthält. Jedes Mal, wenn ein neues Verzeichnis im Archiv des Projekts angelegt wird und jemand seine Arbeitskopie aktualisiert, ohne `-d` bei `update` anzugeben, wird das neue Verzeichnis nicht in der Arbeitskopie angelegt werden. Lokal betrachtet weiß CVS nichts davon, dass es ein neues Verzeichnis im Archiv gibt, und wird die Datei `Entries.Static` entfernen, wenn das `Update` gelaufen ist, selbst wenn das neue Verzeichnis nicht in der Arbeitskopie vorhanden ist.

`CVS/Tag`

Wenn die Datei `CVS/Tag` vorhanden ist, so benennt sie eine Marke, die sozusagen dem Verzeichnis zugeordnet ist. Ich sage bewusst **sozusagen**, da, wie Sie ja wissen, CVS keinerlei Revisionshistorie für Verzeichnisse vorhält und streng genommen auch keine Marken an ihnen anbringen kann. Marken werden nur an normale Dateien angebracht, oder, genauer gesagt, bestimmten Revisionen einer normalen Datei zugeordnet.

Wenn jedoch jede Datei in einem Verzeichnis einer bestimmten Marke zugeordnet ist, dann stellt sich CVS die Marke als auch an dem ganzen Verzeichnis angebracht vor. Wenn Sie zum Beispiel eine Arbeitskopie aus einem bestimmten Zweig per `Checkout` holen

```
user@linux ~/ # cvs co -r Bugfix_Branch_1
```

und dann eine Datei hinzufügen, so ist es wünschenswert, dass die Anfangsrevision der Datei auch diesem Entwicklungszweig zugeordnet ist. Aus ähnlichen Gründen muss CVS wissen, ob das Verzeichnis eine sonstige bindende Marke 2 oder einen Datumsstempel aufweist.

Die für die Marken zuständigen Dateien (`CVS/Tag`) enthalten nur eine Zeile. Das erste Zeichen dieser Zeile ist ein einbuchstabiges Kürzel, das aussagt, um was für eine Marke es sich handelt; der Rest der Zeile ist der Name der Marke. Zurzeit verwendet CVS nur diese drei einbuchstabigen Kürzel:

T-Die Marke eines Zweiges

N-eine normale Marke (also nicht die Marke eines Zweiges)

D-ein bindendes Datum. So etwas tritt auf, wenn ein Kommando wie

```
user@linux ~/ # cvs checkout -D 1999-05-15 myproj
```

oder

```
user@linux ~/ # cvs update -D 1999-05-15 myproj
```

gestartet wird.

(Wenn Sie noch andere einbuchstabige Kürzel entdecken, dann heißt das, dass CVS neue Markenarten eingeführt hat, seitdem dieses Kapitel geschrieben wurde.)

Sie sollten die Datei `Tag` nicht von Hand entfernen; verwenden Sie besser `cvs update -A`.

Echte Seltenheiten

Es gibt noch andere Dateien, die Sie gelegentlich im `CVS/`-Unterverzeichnis vorfinden können:

`CVS/Checkin.prog, CVS/Update.prog CVS/Notify, CVS/Notify.tmp CVS/Base/, CVS/Baserev, CVS/Baserev.tmp CVS/Template`

Diese Dateien verursachen normalerweise keine Probleme, daher führe ich sie hier nur der Vollständigkeit halber auf (siehe Kapitel 9 für deren vollständige Beschreibung).

Portabilität und zukünftige Erweiterungen

Jedes Mal, wenn CVS neue Features hinzugefügt werden, können neue Dateien (die ier nicht aufgeführt sind) in den administrativen Abteilungen der Arbeitskopien auftreten. Wenn es neue Dateien gibt, so werden sie wahrscheinlich im *Cederqvist* unter **Working Directory Storage** dokumentiert. Wenn Sie es vorziehen, aus dem Quelltext zu lernen, können Sie auch in `src/cvs.h` aus der Quelltextdistribution nachlesen.

Bedenken Sie, dass alle `CVS/*`-Dateien - aktuelle und zukünftige - jeweils die Zeilenendekennung verwenden, die für das lokale System der Arbeitskopie (zum Beispiel LF unter UNIX, CRLF unter Windows) angebracht ist. Das bedeutet, dass wenn Sie eine Arbeitskopie von einer Maschine auf eine andere verfrachten, CVS sie nicht mehr verarbeiten können wird. (Sie haben dann aber noch ganz andere Probleme, da die unter Revisionskontrolle stehenden Dateien selbst die für ihre neue Heimat falsche Zeilenendekennung haben werden.)

2.2 Zugriffsrechte im Archiv

CVS setzt nicht ein bestimmtes Schema bei der Vergabe der Zugriffsrechte im Archiv voraus - es kommt mit einem breiten Spektrum davon aus. Um allerdings einem verwirrenden Verhalten vorzubeugen, sollten Sie sicherstellen, dass Sie sich beim Einrichten des Archivs an die folgenden Kriterien halten:

Wenn ein Benutzer irgendeine Form des Zugriffs auf ein Unterverzeichnis im Archiv wünscht - und sei es nur lesender Zugriff - benötigt er für gewöhnlich auf Systemebene Schreibzugriff auf das Unterverzeichnis. Das ist nötig, weil CVS temporäre Lockfiles im Archiv erzeugt, um die Datenkonsistenz gewährleisten zu können. Auch reine Leseoperationen (wie `checkout` oder `update`) erzeugen Lockfiles, die anzeigen, dass die Daten, bis sie fertig sind, in ein und demselben Zustand zu bleiben haben.

Wie schon in Kapitel 4 besprochen, kann man die Notwendigkeit der Schreibberechtigung umgehen, indem man in `CVS/config` den Parameter LockDir wie folgt setzt:

CVS/config

```
LockDir=/usr/local/cvslocks
```

Natürlich müssen Sie dann dafür sorgen, dass das Verzeichnis `/usr/local/cvslocks` für alle Benutzer von CVS schreibbar ist. In jedem Fall benötigen die meisten CVS-Operationen, auch die rein lesenden, irgendwo ein beschreibbares Verzeichnis. Als Vorgabe ist dieses Verzeichnis das des Archivs; wenn Sie sehr sicherheitsbewusst sind, dann lenken Sie es woandershin um.

Stellen Sie sicher, dass die Datei `CVSROOT/history` (wenn sie existiert) world-writeable3 ist. Wenn die History-Datei existiert, dann versuchen die meisten CVS-Operationen ihr einen Eintrag hinzuzufügen, und wenn dieser Versuch fehlschlägt, schlägt die ganze Operation mit einer Fehlermeldung fehl. Leider (und unerklärlicherweise) ist die History-Datei nicht von Geburt an world-writeable, wenn Sie ein neues Archiv mit `cvs init` erzeugen. Zumindest bei der derzeit aktuellen CVS-Version sollten Sie deren Zugriffsrechte direkt nach dem Erzeugen des neuen Archivs ändern (oder die Datei einfach entfernen, wenn Sie das Speichern der Projekthistorie gänzlich unterbinden wollen).

Bemerkung

Dieses Problem könnte sich von alleine lösen - ich habe den Betreuern von CVS einen Patch zukommen lassen, durch den die History-Datei world-writeable wird, wenn ein neues Archiv angelegt wird. Wenn Sie also eine neuere Version von CVS beziehen, als gerade erhältlich ist (Stand: September 1999), könnte das Problem Sie nicht mehr betreffen.

Aus Sicherheitsgründen sollten Sie dafür sorgen, dass die meisten Benutzer von CVS auf Unix-Ebene keinen Schreibzugriff auf das `CVSROOT`-Verzeichnis haben. Wenn jemand Checkin-Zugriff auf `CVSROOT` hat, dann kann er `commitinfo`, `loginfo` oder beliebige andere triggernde Dateien so anpassen, dass beliebige Programme seiner Wahl aufgerufen werden, er könnte sogar ein neues Programm mit einem `Commit` in das System bringen, wenn das, was er haben möchte, sich noch nicht dort befindet. Sie sollten daher davon ausgehen, dass jeder, der `Commit`- Zugang zu `CVSROOT` hat, in der Lage ist, beliebige Kommandos im System abzusetzen.

3 Häufige Probleme und deren Lösung

Der Rest dieses Kapitels ist in einer Folge von Fragen und Antworten organisiert, ähnlich den FAQ (Frequently Asked Questions 4) aus dem Internet. Sie alle stammen aus wirklichen Erfahrungen mit CVS. Aber bevor wir uns die einzelnen Fälle ansehen, wollen wir uns noch einen Moment Zeit nehmen und uns die CVS Fehlerbehebung allgemein ansehen.

Der erste Schritt bei der Behebung eines CVS-Problems ist üblicherweise der, festzustellen, ob es sich um ein Problem in der Arbeitskopie oder im Archiv handelt. Die beste Technik dafür - und das wird niemanden überraschen -, ist zu überprüfen, ob das Problem auch in anderen Arbeitskopien, als in der, bei der es erstmals festgestellt wurde, auftritt. Wenn das der Fall ist, dann ist es vermutlich ein Problem im Archiv, ansonsten wohl ein lokales Problem der betroffenen Arbeitskopie.

Probleme in den Arbeitskopien treten häufiger auf. Nicht etwa, weil Arbeitskopien irgendwie weniger zuverlässig sind als das Archiv, sondern weil jedes Archiv normalerweise mehrere Arbeitskopien hat. Obwohl die meisten **Verknotungen** in einer Arbeitskopie mit genügend Geduld aufgelöst werden können, ist es häufig effizienter von der Zeit her, einfach die Arbeitskopie zu löschen und mittels `Checkout` neu anzulegen.

Wenn natürlich das Auschecken zu lange dauert oder sich noch wesentliche Änderungen, die Sie noch nicht per `Commit` in das Archiv gespeichert haben und die Sie nicht verlieren wollen, in der Arbeitskopie befinden, oder Sie einfach wissen wollen, was denn nun schief gelaufen ist, dann lohnt es sich, etwas herum zu stochern und die Ursache des Problems herauszufinden. Beim Herumstochern sind die ersten Orte, an denen man suchen sollte, die `CVS/`-Unterverzeichnisse. Prüfen Sie die Dateiinhalte und die Dateizugriffsrechte. Ganz vereinzelt kann es vorkommen, dass die Zugriffsrechte auf einmal **nur lesbar** oder sogar noch nicht einmal lesbar werden. (Ich vermute, dass das eher daran liegt, dass sich ein Benutzer bei einem Unix-Kommando vertippt hat, statt dass ein Fehler aufseiten von CVS vorliegt.)

Probleme im Archiv werden fast immer von falsche Datei- oder Verzeichniszugriffsrechten verursacht. Wenn Sie vermuten, dass ein Problem an falschen Zugriffsrechten im Archiv liegt, sollten sie erst einmal die effektive Benutzerkennung5 derjenigen Person, die das Problem hat, herausfinden. Bei allen lokalen und bei den meisten nicht lokalen Benutzer ist das entweder der normale Benutzername oder der Benutzername, den sie beim Auschecken ihrer Arbeitskopie angegeben haben. Wenn Sie die pserver-Methode zusammen mit user-aliasing verwenden (siehe Abschnitt **Der Passwortauthentisierungs-Server** in Kapitel 4), dann ist die Benutzerkennung diejenige, die in der Datei `CVSROOT/passwd` steht. Wenn Sie das nicht gleich bemerken, können Sie eine Menge Zeit damit verschwenden, den Fehler an der falschen Stelle zu suchen.

Doch jetzt, ohne weitere Umschweife ...

3.1 Alltägliche Probleme und ihre Lösung

Die folgenden Situationen habe ich erlebt, als ich anderen bei ihren Abenteuern mit CVS zur Seite gestanden habe. Dazu kommen noch einige Punkte, die keine echten Probleme sind, sondern lediglich Fragen, die mir schon einmal gestellt wurden und die ich hier beantworten möchte. Ich halte diese Liste für ziemlich umfassend, und manches, was Sie schon in früheren Kapiteln gelesen haben, kann sich hier wiederholen.

Die Situationen sind entsprechend der Häufigkeit ihres Auftretens angeordnet, die häufigsten dabei zuerst.

Ich bekomme dauernd die Meldung 'Waiting for Locks'. Was soll das?

Wenn Sie eine Meldung wie

```
cvs update: [22:58:26] waiting for qsmith's lock in /usr/local/newrepos/myproj
```

erhalten, dann bedeutet das, dass Sie versuchen, auf ein Unterverzeichnis im Archiv zuzugreifen, das momentan von irgendeinem anderen CVS-Prozess belegt wird. Da ein Prozeß in diesem Verzeichnis abläuft, befindet es sich möglicherweise nicht in einem Zustand, der es erlaubt, dass andere CVS-Prozesse darauf zugreifen.

Wenn allerdings diese Wartemeldung lange Zeit bestehen bleibt, dann heißt das, dass ein CVS-Prozess - aus welchem Grund auch immer - dabei gescheitert ist, textblocker sich aufzuräumen. So etwas kann auftreten, wenn CVS plötzlich und unerwartet stirbt, beispielsweise durch Stromausfall bei dem Rechner, der das Archiv beherbergt.

Die Lösung liegt darin, die Lock-Dateien von Hand aus dem fraglichen Verzeichnis im Archiv zu entfernen. Wechseln Sie in das entsprechende Verzeichnis im Archiv, und suchen Sie nach Dateien namens `#cvs.lock` oder solchen, die mit `#cvs.wfl` oder `#cvs.rfl` beginnen. Vergleichen Sie die Zeitstempel der Dateien mit den Startzeiten aller gerade laufenden CVS-Prozesse. Wenn ausgeschlossen ist, dass die Dateien von einem noch laufenden Prozess angelegt worden sind, dann ist es ungefährlich, sie zu löschen. Der wartende CVS-Prozess wird irgendwann merken, dass die `Lock`-Dateien verschwunden sind - das sollte ungefähr 30 Sekunden dauern -, und wird dann der angeforderten Operation gestatten, fortzufahren.

Wenn Sie an mehr Details interessiert sind, dann lesen Sie unter **Locks** im *Cederqvist* nach.

CVS behauptet, dass bei einer Datei der »Up-to-Date Check« fehlschlägt.

Was mache ich bloß?

Keine Panik - das bedeutet einfach, dass sich die Datei, seit Sie sie das letzte Mal ausgecheckt oder ein Update gemacht haben, geändert hat.

Starten Sie `cvs update` auf der Datei, um die Änderungen aus dem Archiv einzuarbeiten. Wenn die Änderungen mit Ihren lokalen Änderungen kollidieren, dann editieren Sie die Datei, um den Konflikt zu beheben. Versuchen Sie danach erneut den `commit` - er wird nicht fehlschlagen -, wenn man einmal die Möglichkeit ausschließt, dass jemand eine weitere Revision per `Commit` veröffentlicht hat, während Sie damit beschäftigt waren, die letzten Änderungen einzuarbeiten.

Ich schaffe es nicht, die pserver-Zugangsmethode ans Laufen zu kriegen.

Der häufigste, aber am wenigsten offensichtliche Grund für dieses Problem ist, dass Sie vergessen haben, das Archiv mit der Option `--allow-root` in Ihrer `inetd`-Konfigurationsdatei aufzulisten.

Erinnern Sie sich an die Beispiel-`/etc/inetd.conf`-Datei aus Kapitel 4:

/etc/inetd.conf

```
cvspserver stream tcp nowait root /usr/local/bin/cvs cvs \
--allow-root=/usr/local/newrepos pserver
```

(In der eigentlichen Datei ist das eine einzige lange Zeile, ohne den Backslash.)

Der Teil `--allow-root=/usr/local/newrepos` ist eine Sicherheitsmaßnahme, die dafür sorgt, dass niemand CVS dazu missbraucht, um `pserver`-Zugang zu Archiven zu erhalten, auf die nur lokal zugegriffen werden soll. Jedes Archiv, auf das pserver-Zugriff möglich sein soll, benötigt einen `--allow-root`-Eintrag. Sie können so viele `--allow-root`-Optionen, wie Sie für alle Archive im System benötigen, haben (oder so viele, wie Sie wollen, solange Sie nicht an die Längenbeschränkung der Argumente für inetd stoßen).

Lesen Sie in Kapitel 4 nach, wenn Sie genauere Informationen zur Einrichtung eines Passwort authentisierten Servers wünschen.

Die pserver-Zugriffsmethode funktioniert IMMER noch nicht!

O.K., wenn es nicht an einem fehlenden `--allow-root` liegt, gibt es noch andere Möglichkeiten:

Der Benutzer hat keinen Eintrag in der Datei `CVSROOT/passwd`, und die Datei `CVSROOT/config` enthält `SystemAuth=no`, sodass CVS nicht auf die System-`passwd`-Datei zurückfällt (oder sie enthält `SystemAuth=yes`, aber die System-`passwd`-Datei enthält auch keinen Eintrag für diesen Benutzer).

Der Benutzer hat zwar einen Eintrag in `CVSROOT/passwd`, aber es gibt keinen entsprechenden Benutzer im System, und `CVSROOT/passwd` bildet den Benutzer nicht auf einen im System gültigen Benutzernamen ab.

Das Passwort ist falsch. (CVS ist aber gut darin, den Benutzer darauf hinzuweisen, also ist das wahrscheinlich nicht die richtige Antwort.)

Die Passwortdateien und `/etc/inetd.conf` sind richtig eingerichtet, aber Sie haben einen Eintrag wie diesen in `/etc/services` vergessen:

etc/services

```
cvspserver 2401/tcp
```

und so lauscht `inetd` gar nicht auf dem Port und kann die Verbindung auch nicht an CVS weiter reichen.

Meine Commits geschehen stückweise, nicht atomar.

Das liegt daran, dass CVS-`Commits` stückweise geschehen und nicht atomar. :-)

Genauer gesagt laufen sie verzeichnisweise ab. Wenn Sie einen `commit` (oder ein `update` oder irgendetwas anderes) ausführen, der mehrere Verzeichnisse betrifft, dann wird CVS jedes korrespondierende Verzeichnis im Archiv nacheinander sperren, während es die Operation darin durchführt.

Bei kleinen bis mittleren Projekten ist das selten ein Problem - CVS erledigt seine Aufgaben in jedem Verzeichnis so schnell, dass es gar nicht auffällt, dass es nicht atomar geschieht. Bei großen Projekten kann es aber leider zu Szenarios wie folgendem kommen. (Stellen Sie sich vor, dass das bei einem Projekt geschieht, das mindestens zwei Unterverzeichnisse (A und B) enthält, die beide viele Dateien enthalten.):

Der Benutzer qsmith startet einen `commit`, der Dateien aus beiden Unterverzeichnissen betrifft. CVS führt den `Commit` zuerst mit den Dateien in Verzeichnis B aus (möglicherweise weil qsmith die Verzeichnisse in seiner Kommandozeile in dieser Reihenfolge angegeben hat).

Die Benutzerin jrandom startet `cvs update`. Aus welchem Grund auch immer startet der `Update`-Prozess in dem Verzeichnis A in der Arbeitskopie. (CVS gibt keine Garantien, in welcher Reihenfolge es Verzeichnisse oder Dateien bearbeitet, wenn man es nicht dazu zwingt.) Beachten Sie, dass es keinen Konflikt mit den Sperren gibt, da ja qsmith in Verzeichnis A noch gar nicht aktiv ist.

Der `commit` von qsmith wird mit B fertig und macht bei A weiter und wird auch dort fertig.

Zum Schluss wechselt der `update`-Prozess von qsmith zu Verzeichnis B und wird auch dort fertig.

Es ist klar, dass, wenn alles vorüber ist, die Arbeitskopie von jrandom die Änderungen von qsmith an Verzeichnis B enthält, nicht jedoch die in Verzeichnis A. Obwohl qsmith vorhatte, dass die Änderungen als Einheit vorgenommen werden, ist es nicht so gekommen. Jetzt ist die Arbeitskopie von jrandom in einem Zustand, den qsmith nie vorhergesehen hätte.

Die Lösung ist natürlich, dass jrandom einen weiteren cvs `update`-Lauf durchführt, der die noch unberücksichtigten Änderungen aus dem `Commit` von qsmith nachholt. Das setzt natürlich voraus, dass jrandom überhaupt die Möglichkeit hat, herauszufinden, dass sie nur einen Teil von qsmiths Änderungen mitbekommen hat.

Es gibt für dieses Dilemma keine einfache Lösung. Sie müssen einfach hoffen, dass der inkonsistente Zustand der Arbeitskopie irgendwie auffällt. (Vielleicht lässt sich die Software nicht übersetzen, oder jrandom und qsmith haben eine für beide verwirrende Unterhaltung, bis ihnen auffällt, was passiert sein muss.)

Dass CVS keine **atomaren** Transaktionen bereitstellt, wird allgemein als Manko gesehen. Der alleinige Grund dafür, dass keine Sperrmechanismen im Hauptverzeichnis des Archivs eingeführt werden, ist der, dass sich dann bei großen Projekten mit vielen Benutzern häufig Verklemmungen dieser Sperren ergeben würden. CVS hat also das kleinere Übel gewählt, das die Häufigkeit dieser Verklemmungen verringert, aber eben diese versetzten Lese- und Schreiboperationen zulässt. Eines Tages wird jemand CVS anpassen (also die Operationen des Archivs beschleunigen), sodass man nicht mehr zwischen zwei Übeln wählen muss; bis dahin müssen wir uns mit den nicht atomaren Vorgängen abfinden.

Für weitere Informationen empfehle ich die Sektion **Concurrency** im *Cederqvist*.

Warum verändert CVS dauernd die Zugriffsrechte meiner Dateien?

CVS ist allgemein nicht gut darin, die Zugriffsrechte von Dateien zu bewahren. Wenn man ein Projekt importiert und dann auscheckt, gibt es keine Garantie, dass die Zugriffsrechte der Dateien in der neuen Arbeitskopie dieselben sind wie beim Importieren des Projekts. Wahrscheinlicher ist, dass die Dateien der Arbeitskopie mit denselben Standardzugriffsrechten erzeugt werden, mit denen neue Dateien normalerweise angelegt werden.

Es gibt aber mindestens eine Ausnahme. Wenn Sie ausführbare Shell-Skripten im Projekt speichern wollen, können Sie sie in jeder Arbeitskopie ausführbar machen, indem Sie die korrespondierende Datei im Archiv ausführbar machen:

```
user@linux ~/ # ls -l /usr/local/newrepos/someproj

total 6
-r--r--r-- 1 jrandom users 630 Aug 17 01:10 README.txt,v
-r-xr-xr-x 1 jrandom users 1041 Aug 17 01:10 scrub.pl,v*
-r--r--r-- 1 jrandom users 750 Aug 17 01:10 hello.c,v
```

Beachten Sie, dass, obwohl die Datei ausführbar ist, sie immer noch read-only ist, wie es alle Dateien im Archiv sein sollten. (Wie schon gesagt macht CVS erst eine temporäre Kopie der `RCS`-Datei, führt alle Operationen auf der Kopie durch und ersetzt, wenn es fertig ist, das Original mit der Kopie.)

Wenn Sie eine ausführbare Datei importieren oder hinzufügen, dann erhält CVS die **Executable Flags** (diese machen sie ausführbar); wenn also die Rechte von Anfang an richtig gesetzt waren, haben Sie nichts zu befürchten. Wenn Sie allerdings aus Versehen die Datei hinzufügen, bevor Sie sie ausführbar gemacht haben, dann müssen Sie manuell die `RCS`-Datei im Archiv ausführbar machen.

Bemerkung

Die Zugriffsrechte im Archiv behalten immer die Oberhand. Wenn also eine Datei im Archiv nicht ausführbar ist, in der Arbeitskopie aber schon, dann wird die Datei in der Arbeitskopie nach einem `Update` nicht mehr ausführbar sein. Die Tatsache, dass sich die Zugriffsrechte der Dateien still und heimlich ändern können, kann ziemlich frustrierend sein. Wenn das geschieht, sollten Sie als Erstes das Archiv überprüfen, um zu sehen, ob Sie das Problem lösen können, indem Sie dort die Zugriffsrechte der entsprechenden `RCS`-Dateien setzen.

Ein Feature namens **PreservePermissions** wurde CVS hinzugefügt, dass manche dieser Probleme erträglicher machen könnte. Durch dessen Nutzung können aber andere unerwartete Ereignisse auftreten (weshalb ich es hier nicht uneingeschränkt empfehle). Machen Sie sich mit den Punkten **config** und **Special Files** im *Cederqvist* vertraut, bevor Sie `PreservePermissions=yes` in `CVSROOT/config` hinein schreiben.

CVS kann unter Windows meine .cvspass-Datei nicht finden. Warum?

Für `pserver`-Verbindungen sucht CVS auf der Client-Seite in Ihrem Home-Verzeichnis nach der Datei `.cvspass`. Windows-Maschinen kennen aber kein natürliches Home-Verzeichnis, weswegen CVS in der Umgebungsvariablen `%HOME%` nachschaut. Man muss aber bei der Einrichtung von HOME sehr vorsichtig sein. Folgendes funktioniert:

Datei .cvspass

```
set HOME=C:
```

Das aber nicht:

Datei .cvspass

```
set HOME=C:\
```

Der abschließende Backslash reicht aus, um CVS zu verwirren; es wird `C:\.cvspass` nicht öffnen können.

Die schnelle und dauerhafte Lösung ist also

Datei .cvspass

```
set HOME=C:
```

in Ihre `autoexec.bat` zu schreiben und Windows neu zu starten. Danach sollte CVS-`pserver` richtig funktionieren.

Meine Arbeitskopie ist auf verschiedene Zweige verteilt. Hilfe!

Sie haben festgestellt, dass verschiedene Unterverzeichnisse Ihrer Arbeitskopie irgendwie in verschiedenen Zweigen gelandet sind? Das liegt vermutlich daran, dass Sie `Updates` mit der Option `-r` durchgeführt haben, jedoch nicht vom Hauptverzeichnis Ihrer Arbeitskopie aus.

Kein Problem. Wenn Sie zur Hauptentwicklungslinie zurückkehren wollen, führen Sie einfach

```
user@linux ~/ # cvs update -r HEAD
```

oder

```
user@linux ~/ # cvs update -A
```

aus dem Hauptverzeichnis aus. Wenn Sie hingegen Ihre ganze Arbeitskopie auf eine bestimmte Entwicklungslinie setzen wollen, führen Sie Folgendes aus:

```
user@linux ~/ # cvs update -r Name_des_Zweiges
```

Es ist an sich nichts Falsches daran, ein oder zwei Unterverzeichnisse Ihrer Arbeitskopie auf einem anderen Zweig zu haben als den Rest, wenn Sie beispielsweise einige kurze Arbeiten in diesem Zweig und nur in diesen bestimmten Verzeichnissen durchführen wollen. Es ist aber generell eine gute Idee zurückzukehren, sobald Sie fertig sind - das Leben ist viel weniger verwirrend, wenn Ihre ganze Arbeitskopie derselben Entwicklungslinie folgt.

Wenn ich export -D durchführe, scheint es manchmal die neusten Änderungen zu missachten!

Das liegt an den unterschiedlich laufenden Uhren auf der lokalen Maschine und im Archiv. Sie können das Problem lösen, indem Sie eine oder beide Uhren neu stellen oder indem Sie ein anderes Datum als Argument zu `-D` angeben. Es ist durchaus akzeptabel, ein in der Zukunft liegendes Datum (wie zum Beispiel `-D tomorrow`) anzugeben, wenn das nötig ist, um den Zeitunterschied zu kompensieren.

Ich kann export -r nicht ausführen wegen irgendwelcher 'val-tags'. Was soll das?

Siehe nächste Frage.

Tag-Operationen schlagen wegen irgendwelcher 'val-tags' fehl. Was soll das?

Wenn Sie einen ähnlichen Fehler wie diesen

```
cvs [export aborted]: cannot write /usr/local/myproj/CVSROOT/val-tags: \
Operation not permitted
```

erhalten, dann bedeutet das, dass der Benutzer, unter dem CVS läuft, nicht die Berechtigung hat, in die Datei `CVSROOT/val-tags` zu schreiben. In dieser Datei werden gültige Namen von Marken gespeichert, damit CVS schnell feststellen kann, welche Marken gültig sind. Leider verändert CVS diese Datei selbst für Operationen, die auf das Archiv bezogen eigentlich nur lesend sind, wie beispielsweise das Auschecken eines Projektes.

Dieser Fehler in CVS könnte zurzeit, da Sie dies lesen, schon behoben sein. Bis es so weit ist, liegt die Lösung entweder darin, die Datei `val-tags world-writeable` zu machen oder, wenn das nicht hilft, sie zu löschen oder ihren Eigentümer auf den abzuändern, der CVS laufen lässt. (Man sollte denken, dass das Ändern der Zugriffsrechte genügen sollte, ich habe aber schon mehrere Situationen erlebt, bei denen ich auch den Eigentümer ändern musste.)

Ich habe Probleme mit bindenden Marken, ich möchte sie loswerden.

Verschiedene CVS-Operationen ziehen mit sich, dass die Arbeitskopie eine bindenden Marke, ein so genanntes **Sticky Tag**, erhält, also eine einzige Marke, der jeder Revision jeder Datei zugeordnet ist. (Im Falle einer verzweigten Version wird die bindende Marke auf jede Datei, die der Arbeitskopie hinzugefügt wird, angewendet.) Ihre Arbeitskopie erhält jedes Mal eine bindende Marke, wenn sie abhängig von einer Marke oder von einem Datum auschecken oder ein `Update` durchführen.

Zum Beispiel:

```
user@linux ~/ # cvs update -r Name_der_Marke
```

oder

```
user@linux ~/ # cvs checkout -D "1999-08-16"
```

Wenn ein Datum oder der Name einer normalen Marke (nicht die eines Zweiges) verwendet wird, dann wird die Arbeitskopie zu einem **eingefrorenen** Abbild dieses Augenblicks in der Geschichte des Projektes - so werden Sie natürlich nicht in der Lage sein, irgendwelche Änderungen daraus per `Commit` im Archiv abzulegen.

Um die bindende Marke zu entfernen, lassen Sie `update` mit der Option `-A` laufen

```
user@linux ~/ # cvs update -A
```

wodurch die bindende Marke aufgelöst wird und jede Datei auf den neusten Stand der Hauptentwicklungslinie gebracht wird.

CVS Checkout/Update schlägt mit einer Fehlermeldung fehl, die 'cannot expand modules' lautet.

Das ist ein Fall einer falschen Fehlermeldung in CVS; wahrscheinlich wird früher oder später jemand sie korrigieren, in der Zwischenzeit kann sie Sie aber erwischen.

Die Fehlermeldung sieht etwa so aus:

```
user@linux ~/ # cvs co -d bwf-misc user-space/bwf/writings/misc

cvs server: cannot find module 'user-space/bwf/writings/misc' - ignored
cvs [checkout aborted]: cannot expand modules
```

CVS scheint zu sagen, dass etwas mit der Datei `CVSROOT/modules` nicht stimmt. In Wahrheit handelt es sich aber um ein Problem mit den Zugriffsrechten im Archiv. Entweder ist das Verzeichnis, das ich auschecken möchte, nicht lesbar, oder eines seiner übergeordneten Verzeichnisse ist nicht lesbar. In diesem Fall war eines der übergeordneten Verzeichnisse schuld:

```
user@linux ~/ # ls -ld /usr/local/cvs/user-space/bwf

drwx------ 19 bwf users 1024 Aug 17 01:24 bwf/
```

Lassen Sie sich nicht von dieser offensichtlich falschen Fehlermeldung täuschen - es ist ein Problem mit den Zugriffsrechten im Archiv.

Irgendwie kann ich die Watches nicht abschalten!

Sie haben wahrscheinlich

```
user@linux ~/ # cvs watch remove
```

für alle Dateien aufgerufen, aber vergessen, ebenfalls

```
user@linux ~/ # cvs watch off
```

aufzurufen. Ein kleiner Hinweis, wie man Probleme mit `Watches` (Beobachtungslisten) diagnostiziert: Manchmal ist es immens erhellend, wenn man einfach in das Archiv geht und die `CVS/fileattr`-Dateien direkt untersucht. Siehe Kapitel 4 für weitere Informationen darüber.

Meine Binärdateien werden verunstaltet.

Haben Sie daran gedacht, `-kb` anzugeben, als Sie sie dem Archiv hinzugefügt haben? Wenn nicht, dann könnte CVS Zeilenendekonvertierungen oder `RCS`-Schlüsselwortersetzungen durchgeführt haben. Die einfachste Lösung ist normalerweise, sie als Binärdatei zu kennzeichnen:

```
user@linux ~/ # cvs admin -kb foo.gif
```

und es dann nach Korrektur der Datei dem Archiv per `Commit` mitzuteilen. CVS wird sie bei dem neuen `Commit` nicht wieder verändern und bei späteren `Commits` auch nicht, da es ja jetzt weiß, dass es sich um eine Binärdatei handelt.

CVS macht die Zeilenendekonvertierung nicht richtig.

Wenn Sie CVS auf einer Nicht-Unix-Plattform laufen haben und bei manchen Arbeitskopien nicht die gewünschte Zeilenendekennung erhalten, dann liegt das normalerweise daran, dass die Dateien aus Versehen mit der Option `-kb` hinzugefügt wurden. Im Archiv wird das, ob man es glaubt oder nicht, mit dem Kommando

```
user@linux ~/ # cvs admin -kkv DATEI
```

behoben. Die Option `-kkv` besagt, dass normale Schlüsselwortersetzung durchgeführt werden soll, und impliziert auch die normale Zeilenendekonvertierung. (CVS ist intern betrachtet ein wenig wirr, wenn es um die Unterscheidung zwischen Schlüsselwortersetzung und Zeilenendekonvertierung geht. Das zeigt sich darin, dass man mit den `-k`-Optionen nicht beide Parameter festlegen kann.)

Leider behebt das `admin`-Kommando das Problem mit der Datei nur im Archiv - Ihre Arbeitskopie denkt immer noch, dass es sich um eine Binärdatei handelt. Sie können zwar die Zeile für diese Datei in `CVS/Entries` anpassen, sodass sie `-kb` nicht mehr enthält, aber das löst das Problem nicht in den anderen Arbeitskopien, die es sonst noch gibt.

Ich muss ein Unterverzeichnis aus meinem Projekt löschen. Wie stelle ich das an?

Tja, richtig löschen können Sie das Unterverzeichnis nicht, aber Sie können alle Dateien darin entfernen (indem Sie sie erst löschen, dann `cvs remove` und `commit` ausführen). Sobald das Verzeichnis leer ist, kann jeder es automatisch aus seiner Arbeitskopie heraus schneiden, indem er die Option `-P` beim `update` mit angibt.

Kann ich .cvspass-Dateien oder Teile davon kopieren?

Ja, das ist möglich. Man kann `.cvspass`-Dateien von Rechner zu Rechner kopieren, und man kann auch einzelne Zeilen aus einer `.cvspass`-Datei zur anderen kopieren. Bei Servern mit hoher Latenz kann das schneller gehen, als sich mittels `cvs login` bei jedem Rechner, der eine Arbeitskopie beheimatet, anzumelden.

Denken Sie daran, dass es vermutlich nicht funktionieren wird, wenn Sie eine `.cvspass`-Datei zwischen zwei Rechnern mit unterschiedlicher Zeilenendekennung transferieren. (Natürlich können Sie die Konvertierung wahrscheinlich ohne große Mühe manuell durchführen.)

Ich habe gerade einige Dateien mit der falschen Log-Nachricht committed.

Sie müssen nichts im Archiv von Hand editieren, um das zu korrigieren. Lassen Sie einfach `admin` mit der Option `-m` laufen. Bedenken Sie, dass Sie kein Leerzeichen zwischen die Option `-m` und ihr Argument setzen dürfen und dass Sie die Ersatzlog-Nachricht genau wie jede normale mit Anführungszeichen kapseln müssen:

```
user@linux ~/ # cvs admin -m1.17:"I take back what I said about the customer."
hello.c
```

Ich möchte Dateien verschieben, ohne die Revisionshistorie zu verlieren.

Kopieren Sie (nicht verschieben) die RCS-Dateien an den gewünschten neuen Platz im Projekt. Sie müssen zusätzlich an der alten Stelle verbleiben.

Führen Sie dann aus einer Arbeitskopie Folgendes aus:

```
user@linux ~/ # rm alte_Datei_1 alte_Datei_2 ...
user@linux ~/ # cvs remove alte_Datei_1 alte_Datei_2 ...
user@linux ~/ # cvs commit -m "removed from here" alte_Datei_1 alte_Datei_2
...
```

Wenn jetzt jemand ein `Update` durchführt, dann wird CVS die alten Dateien richtigerweise entfernen und die neuen Dateien in die Arbeitskopie hineinbringen, so als ob sie ganz normal dem Archiv hinzugefügt worden wären (abgesehen davon, dass Sie für angeblich neue Dateien ungewöhnlich hohe Revisionsnummern haben).

Wie bekomme ich eine Liste aller Marken in einem Projekt?

Dafür gibt es in CVS momentan keine bequeme Möglichkeit. Alle Benutzer bekommen diesen Mangel schmerzlich zu spüren, und ich glaube, dass daran gearbeitet wird, diese Funktion verfügbar zu machen. Wenn Sie dies lesen, gibt es vielleicht schon ein Kommando `cvs tags` oder so ähnlich.

Bis es soweit ist, können Sie die Liste auch über Umwege erhalten. Führen Sie `cvs log -h` aus, und lesen Sie den Teil der Ausgabe, welcher der Kopfzeile symbolic names: folgt. Oder, falls Sie sich auf dem Rechner, auf dem das Archiv liegt, befinden, können Sie sich einfach den Anfang einiger der RCS-Dateien direkt im Archiv ansehen. Alle Marken (solche, die sich auf Entwicklungszweige beziehen, wie auch solche, die das nicht tun) sind im Feld symbols aufgeführt:

```
user@linux ~/ # head /usr/local/newrepos/hello.c,v

head 2.0;
access;
symbols
Release_1_0:1.22
Exotic_Greetings-2:1.21
merged-Exotic_Greetings-1:1.21
Exotic_Greetings-1:1.21
merged-Exotic_Greetings:1.21
Exotic_Greetings-branch:1.21.0.2
Root-of-Exotic_Greetings:1.21
start:1.1.1.1
jrandom:1.1.1;
locks; strict;
comment @ * @;
```

Wie bekomme ich eine Liste aller Projekte im Archiv?

Wie auch das Auflisten aller Marken, so ist diese Funktion in der aktuellen Version von CVS nicht implementiert, aber es ist sehr wahrscheinlich, dass es bald implementiert wird. Ich glaube, dass das Kommando `cvs list` oder kurz `cvs ls` genannt werden wird und dass es sowohl die Datei modules parsen wird, als auch die Unterverzeichnisse im Archiv auflisten wird.

In der Zwischenzeit fahren Sie wahrscheinlich am besten damit, sich die Datei `CVSROOT/modules` (entweder direkt oder mit `checkout -c`) anzusehen. Wenn allerdings noch niemand ein Modul für ein bestimmtes Projekt angelegt hat, wird es da auch nicht aufgeführt werden.

Manche Kommandos schlagen fehl, wenn sie remote ausgeführt werden, nicht aber lokal. Wie kann ich das

Problem eingrenzen und beheben?

Es gibt hin und wieder Probleme mit der Kommunikation zwischen Client und Server. Wenn das der Fall ist, so handelt es sich um einen Fehler in CVS, aber wie soll man so etwas aufspüren?

CVS ermöglicht es, das Protokoll zwischen Client und Server mitzulesen. Bevor Sie das Kommando auf dem lokalen Rechner (mit der Arbeitskopie) ausführen, setzen Sie die Umgebungsvariable `CVS_CLIENT_LOG`. Die Bourne-Shell-Syntax dafür lautet so:

```
user@linux ~/ # CVS_CLIENT_LOG=clog; export CVS_CLIENT_LOG
```

Sobald diese Variable gesetzt wurde, zeichnet CVS die gesamte Kommunikation zwischen Client und Server in zwei Dateien auf, deren Namen auf dem Namen in der Variablen basieren:

```
user@linux ~/ # ls

CVS/ README.txt a-subdir/ b-subdir/ foo.gif hello.c
cvs update
? clog.in
? clog.out
cvs server: Updating .
cvs server: Updating a-subdir
cvs server: Updating a-subdir/subsubdir
cvs server: Updating b-subdir

user@linux ~/ # ls

CVS/ a-subdir/ clog.in foo.gif
README.txt b-subdir/ clog.out hello.c

user@linux ~/ #
```

Die Datei `clog.in` enthält alles, was der Client zum Server geschickt hat, und `clog.out` enthält all das, was der Server zurück zum Client gesendet hat. Hier der Inhalt von `clog.out`, damit Sie eine Vorstellung von dem Protokoll erhalten:

clog.out

```
Valid-requests Root Valid-responses valid-requests Repository \
Directory Max-dotdot Static-directory Sticky Checkin-prog Update-prog
\ Entry Kopt Checkin-time Modified Is-modified UseUnchanged Unchanged
\ Notify Questionable Case Argument Argumentx Global_option
Gzip-stream \ wrapper-sendme-rcsOptions Set expand-modules ci co
update diff log add \ remove update-patches gzip-file-contents status
rdiff tag rtag import \ admin export history release watch-on
watch-off watch-add watch-remove \ watchers editors init annotate noop
ok M ? clog.in M ? clog.out E cvs server: Updating .
E cvs server: Updating a-subdir
E cvs server: Updating a-subdir/subsubdir
E cvs server: Updating b-subdir
ok
```

Die Datei `clog.in` ist sogar noch komplizierter aufgebaut, da Revisionsnummern und andere dateibezogene Informationen zum Server geschickt werden müssen.

Ich habe hier nicht genug Platz, um das Client/Server-Protokoll zu dokumentieren, aber Sie können, wenn Sie eine

vollständige Beschreibung wünschen, die Info-Seiten namens **cvsclient** lesen, die bei CVS mitgeliefert werden. Man kann wahrscheinlich eine Menge über das Protokoll herausfinden, wenn man es einfach roh liest. Wenngleich Sie wohl kaum anfangen werden, die Client/Server-Kommunikation zu untersuchen, bevor Sie nicht alle anderen möglichen Ursachen für ein Problem ausgeschlossen haben, ist es doch ein unschätzbar wertvolles Hilfsmittel, um herauszufinden, was wirklich zwischen den beiden vorgeht.

Ich konnte mein Problem nicht in diesem Kapitel beschrieben finden.

Mailen Sie eine vollständige und genaue Beschreibung Ihres Problems an info-cvs@gnu.org, das ist die Diskussions-Mailingliste von CVS. Ihre Mitglieder sind über viele Zeitzonen verteilt, und ich habe normalerweise innerhalb von ein bis zwei Stunden eine Antwort erhalten. Treten Sie der Liste bei, indem Sie eine E-Mail an info-cvs-request@gnu.org schicken, dann können auch Sie dabei helfen, Fragen zu beantworten.

Ich glaube, ich habe einen Fehler in CVS gefunden. Was mache ich?

CVS ist weit davon entfernt, perfekt zu sein - wenn Sie die Dokumentation schon gelesen und auch die Frage auf der Mailingliste erörtert haben und immer noch glauben, dass Sie einen Fehler vor sich haben, dann ist das vermutlich auch der Fall.

Schicken Sie eine so komplette Beschreibung wie möglich an bug-cvs@gnu.org. (Sie können auch dieser Liste beitreten, verwenden Sie einfach stattdessen bug-cvs-request.) Stellen Sie sicher, dass Sie Ihre Versionsnummer von CVS beifügen (sowohl die Client- als auch die Serverversion, falls von Bedeutung) sowie ein Rezept, wie der Fehler reproduziert werden kann.

Wenn Sie einen Patch geschrieben haben, der den Fehler behebt, dann fügen Sie ihn bei und erwähnen das auch im Betreff der E-Mail. Die Betreuer werden Ihnen sehr dankbar sein.

(Weitere Details über diese Vorgänge sind unter BUGS im Cederqvist und in der Datei `HACKING` in der Quelltextdistribution von CVS umrissen.)

Ich habe ein neues Feature für CVS implementiert, wem schicke ich das?

Gleiche Vorgehensweise wie bei einem Fehler: Schicken Sie den Patch an bugs-cvs@gnu.org. Lesen Sie aber zuerst die Datei `HACKING`.

3.2 Die Dinge ändern sich

Die Techniken der Fehlerbeseitigung und die bekannten Fehler, die in diesem Kapitel beschrieben worden sind, waren (ungefähr) zu Zeiten von CVS Version 1.10.7 zutreffend. In der Welt von CVS bewegt sich aber alles sehr schnell. Als ich die letzten paar Kapitel geschrieben habe, ist der inoffizielle Mantel der CVS-Betreuung von Cyclic Software auf SourceGear, Inc. (www.sourcegear.com) übergegangen, denn diese haben Cyclic aufgekauft. SourceGear hat öffentlich verkündet, dass sie beabsichtigen, eine aktive Rolle bei der CVS-Betreuung zu übernehmen, und hat die Zustimmung von Cyclic erhalten, was (mehr oder weniger) ausreicht, um sie zu den derzeitigen »Hauptbetreuern« von CVS zu machen. (Die Adresse www.cyclic.com bleibt aber erhalten, sodass alle URLs, die vorher in diesem Buch angegeben wurden, ihre Gültigkeit behalten.)

SourceGear ist momentan damit beschäftigt, die verschiedenen Patches, die noch herumfliegen, zu bereinigen und zu organisieren, mit der Absicht, viele davon in CVS zu integrieren. Manche dieser Patches werden wahrscheinlich Fehler beheben, die ich schon aufgelistet habe, und manche werden den Anwendern von CVS neue Mittel zur Fehlerbehebung bieten.

Die beste Möglichkeit, um up-to-date mit dem, was da vorgeht, zu bleiben, ist die Datei NEWS in Ihrer CVS-Distribution zu lesen, die Mailinglisten zu beobachten und Änderungen im Cederqvist und in der Online-Version einiger der Kapitel aus diesem Buch zu verfolgen.

1. Anm. d. Übers.: Schwarze (geschlossene) Kiste

2. Anm. d. Übers.: »Nonbranch Sticky Tag«
3. Anm. d. Übers.: Von jederman beschreibbar
4. Anm. d. Übers.: Häufig gestellte Fragen
5. Anm. d. Übers.: User ID
6. Anm. d. Übers.: Im Deutschen auch Rückstrich genannt: »\«.
7. Anm. d. Übers.: Batch-Dateien, um ein anderes »deutsches« Wort zu bemühen. Stapelverarbeitungsdateien, wenn's denn unbedingt sein muss.