Kurzeinführung in CVS

Dipl.-Inf. Ursula Vollmer

30. November 2004

Inhaltsverzeichnis

Kapitel 1

Und los geht's

1.1 ... womit eigentlich?

CVS (Concurrent Versions System) dient der Versionsverwaltung (vor allem) von Quellcode-Dateien.
Das System ermöglicht es, unterschiedlichen Benutzern (auch parallel) auf die selben Daten zuzugreifen und diese zu bearbeiten.
Das vorliegende Dokument soll lediglich als kurze Einführung dienen. Weitere Informationen sind der Manual-Seite (**man cvs**), der offiziellen CVS-Homepage (**https://i““““““o“3““““3““““f“d“5o““f“2“““z.oszar.com/**) bzw. der entsprechenden Fachliteratur zu entnehmen.

1.2 Erst mal ...

Da eine Installation und allgemeine Konfiguration von CVS am IMS bereits vorhanden ist, werden diese Aspekte hier nicht weiter behandelt.
Allerdings sind für die Arbeit mit CVS trotzdem noch ein paar Vorbereitungen zu treffen, die in diesem Kapitel kurz erklärt werden.

1.2.1 ... Umgebungsvariablen setzen ...

Es besteht die Möglichkeit, die Daten vor unbeabsichtigten Änderungen zu schützen. Dazu muß die Umgebungsvariable **CVSREAD** gesetzt werden. Dies bewirkt, daß Dateien grundsätzlich erst mal schreibgeschützt vorliegen und erst nach einem **edit**-Kommando bearbeitet werden können.
Die Daten, die von CVS verwaltet werden sollen, werden in einem oder mehreren Archiv(en), **Repository** genannt, gehalten. Auf das Erstellen eines solchen Repository soll an dieser Stelle nicht weiter eingegangen werden, da am IMS bereits ein allgemein verwendetes Repository (**/mount/cvs-repository**) existiert.
Um auf dieses Repository zugreifen zu können, müssen jedoch bestimmte Umgebungsvariablen gesetzt werden. Hierbei sind Unterschiede zwischen lokalem Zugriff und Remote-Zugriff zu beachten.

1.2.1.1 ... bei lokalem Zugriff

Ist das Verzeichnis, welches das Repository enthält, von dem lokalen Rechner aus zugänglich, dann muß die Umgebungsvariable **CVSROOT** auf den entsprechenden Pfad, also **/mount/cvs-repository**, gesetzt sein. Dafür muß im Fall von **csh** und **tcsh** in der Datei **.cshrc** die Zeile

setenv CVSROOT /mount/cvs-repository

stehen. Für **sh** und **bash** müssen in **.profile** bzw. **.bash** die folgenden Zeilen hinzugefügt werden:

CVSROOT=/mount/cvs-repository
export CVSROOT

1.2.1.2 ... bei Remote-Zugriff

Hat der Benutzer einen Account auf der Server-Maschine, dann kann er auch über eine Remote-Shell auf die Daten zugreifen. In diesem Fall muß die Umgebungsvariable **CVSROOT** folgendermaßen gesetzt sein:

CVSROOT=:ext:*Login-Name*@*Server-Name*:/mount/cvs-repository[1]

Normalerweise verwendet CVS **rsh**, um die Verbindung herzustellen. Soll statt dessen (z.B. aus Sicherheitsgründen) **ssh** verwendet werden, muß zusätzlich noch die Umgebungsvariable **CVS_RSH** neu gesetzt werden:

CVS_RSH=ssh

1.2.2 ... Daten hinzufügen

Die zu verwaltenden (Projekt-)Daten müssen zu einem CVS-Repository hinzugefügt werden.
Dazu werden die Daten in eine sinnvolle Struktur gebracht und mit dem Befehl

cvs import *Verzeichnisname Hersteller Version*

hinzugefügt.
Der Hersteller bezieht sich dabei auf den Hersteller / Vertreiber der Software, die hinzugefügt werden soll. Damit soll Fremd-Software leicht identifizierbar gemacht werden.
Die Verzeichnisstruktur, die hinzugefügt wurde, kann jetzt lokal gelöscht werden. Um mit den Daten arbeiten zu können, muß eine lokale Kopie davon hergestellt werden (siehe nächsten Abschnitt).

1.2.3 ... lokale Kopie der Daten erstellen

Um mit Daten, die von CVS verwaltet werden, arbeiten zu können, muß eine Kopie der benötigten Daten auf dem lokalen Rechner erstellt werden. Dazu wird in dem lokalen Verzeichnis, welches die Datenstruktur enthalten soll, der folgende Befehl ausgeführt:

cvs checkout *Verzeichnisname*

[1] In dem vorliegenden Dokument sind kursiv gedruckte Texte als Platzhalter zu verstehen, die entsprechend zu ersetzen sind. Der kursiv gesetzte Text soll verdeutlichen, was an dieser Stelle eingesetzt werden muß.

Kapitel 2

Wie kann ich mit CVS ...

2.1 ... Moment - erst mal ein Überblick ...

Die folgende Tabelle enthält eine alphabetische Auflistung der am häufigsten verwendeten Befehle mit einer kurzen Funktionsbeschreibung.

add	Hinzufügen neuer Dateien bzw. Verzeichnisstrukturen
checkout	Erstellen einer lokalen Kopie von Dateien bzw. Verzeichnisstrukturen
commit	Übergabe von geänderten Dateien an das Repository
edit	Vorbereiten einer Datei zum Bearbeiten
remove	Löschen von Dateien aus dem Repository (wird erst nach **commit** wirksam)
tag	Zuweisung eines Tag zu Dateien
unedit	Abbruch der Bearbeitung einer Datei; Wiederherstellen des Schreibschutzes
update	Aktualisierung der lokalen Kopie der Datei bzw. Verzeichnisstruktur

Die praktische Anwendung der Befehle sieht folgendermaßen aus:

cvs *Befehlsname [Optionen] Datei(en)/Verzeichnis(se)*

Jedem Befehl wird also das Schlüsselwort **cvs** vorgestellt.

2.2 ... täglich ...

Die Befehle, die im Umgang mit CVS eigentlich fast ständig benötigt werden, sollen an dieser Stelle kurz erläutert werden. Eine kurze Beschreibung der Optionen, die im Zusammenhang mit einem Befehl häufig verwendet werden, steht dabei ebenfalls zur Verfügung.

2.2.1 ... eine lokale Kopie erstellen?

Um mit den Daten aus CVS arbeiten zu können, muß der Benutzer sich eine lokale Kopie dieser Daten auf seinen Rechner holen. Dazu dient das Kommando **checkout**. Die Syntax für dieses Kommando sieht folgendermaßen aus:

cvs checkout *[Optionen] Verzeichnisname*

Mittels der Option **-d** kann ein Verzeichnis angegeben werden, in welches die Daten kopiert werden sollen. Wird diese Option nicht verwendet, werden die Daten direkt in das aktuelle Verzeichnis kopiert.
Falls die Umgebungsvariable **CVSREAD** gesetzt ist, liegen die kopierten Daten in einer schreibgeschützten Version vor.

2.2.2 ... meine lokalen Daten aktualisieren?

Um sicherzustellen, daß Änderungen immer an der neuesten Version der entsprechenden Datei vorgenommen werden, sollte vor jeder Bearbeitung die lokale, evtl. schreibgeschützte Kopie mit dem Kommando **update** aktualisiert werden. Der Befehl hat die folgende syntaktische Struktur:

cvs update *[Optionen] Datei-/Verzeichnisname*

Die Option **-C** ersetzt auch bereits veränderte Dateien, die sonst beim Update ignoriert werden, durch neue Kopien. Allerdings wird zuvor eine Sicherheitskopie der veränderten Dateien erzeugt.
Um auch neue Unterverzeichnisse zu erhalten, sollte die Option **-d** verwendet werden.
Leere Verzeichnisse werden mittels **-P** entfernt.

2.2.3 ... Dateien bearbeiten?

Nach dem Kopieren sind die Dateien, sofern die Umgebungsvariable **CVSREAD** gesetzt ist, erst mal schreibgeschützt. Um eine Datei bearbeiten zu können, muß man diese (bzw. das gesamte Verzeichnis) mit dem Befehl **edit** zum Bearbeiten vorbereiten.
Damit wird der Schreibschutz von der Datei / den Dateien entfernt. Außerdem werden ggf. andere Anwender benachrichtigt und es wird ein temporärer Beobachtungs-Zustand eingerichtet, der bei **unedit** und **commit** automatisch wieder aufgehoben wird.
Soll die Änderung einer Datei verworfen werden, so kann dafür das Kommando **unedit** verwendet werden. Damit wird die Bearbeitung der Datei abgebrochen und der Schreibschutz wieder hergestellt. Ggf. werden andere Anwender benachrichtigt.
Die Syntax für die Kommandos **edit** und **unedit** sieht wie folgt aus:

cvs [un]edit *[Optionen] Datei-/Verzeichnisname*

2.2.4 ... Dateien hinzufügen?

Um neue Dateien oder Verzeichnisse zur CVS-Verwaltung hinzuzufügen wird das Kommando **add** mit der folgenden Syntax verwendet:

cvs add *[Optionen] Datei-/Verzeichnisname*

Soll eine Binärdatei hinzugefügt werden, so sollte die Option **-kb** verwendet werden.
Mit der Option **-m** kann eine Log-Information übergeben werden.
Um die Dateien endgültig hinzuzufügen, muß erst noch der Befehl **commit**

ausgeführt werden.
Sollen Verzeichnisse direkt unter dem Repository hinzugefügt werden, muß der Befehl **import** verwendet werden. Er hat die folgende Syntax:

cvs import *[Optionen] Verzeichnisname Hersteller Version*

Der Hersteller bezieht sich dabei auf den Hersteller / Vertreiber der Software, die hinzugefügt werden soll. Damit soll Fremd-Software leicht identifizierbar gemacht werden.
Auch hier kann mit der Option **-m** eine Log-Information übergeben werden.

2.2.5 ... geänderte Daten veröffentlichen?

Wurde eine geänderte Datei ausreichend überprüft und soll in CVS als neue Version hinzugefügt werden, so geschieht dies mit dem Befehl **commit**, der die folgende Syntax aufweist:

cvs commit *[Optionen] Datei-/Verzeichnisname*

Die Option **-m** dient der Übergabe einer Log-Information.
Mit **-F** kann erzwungen werden, daß auch unveränderte Dateien als neue Version hinzugefügt werden.

2.2.5.1 Da war wohl jemand schneller ...

Da es mit CVS möglich ist, daß mehrere Personen parallel an einer Datei arbeiten, kann es beim **commit** zu einem Versionenkonflikt kommen.
Um diesen aufzulösen, muß zuerst die lokale Kopie mit dem Kommando **update** aktualisiert werden.
Können die beiden Versionen miteinander verbunden werden, so wird das Update ohne Fehlermeldung beendet und die aktualisierte Version kann durch einen erneuten Aufruf von **commit** publiziert werden.
Konnte der Konflikt nicht automatisch aufgelöst werden, wird die alte Datei automatisch unter **.#*Dateiname.x.y*** abgespeichert, wobei ***x.y*** für die aktuelle Versionsnummer steht. Hier wird der Konflikt markiert:

‹‹‹‹‹‹‹*Dateiname*
Quellcode aus lokaler Kopie
=======
Quellcode aus CVS Distribution
›››››››

Er wird aufgelöst, indem die Datei an der markierten Stelle korrigiert wird. Hierbei müssen die Kontrollmarkierungen gelöscht werden. Danach kann die Datei mit einem erneuten Aufruf von **commit** endgültig publiziert werden.

2.2.5.2 Spuren hinterlassen ...

Soll eine neu publizierte Version z.B. mit der Versionsnummer annotiert werden, so kann mit dem Befehl **tag** ein entsprechendes Tag hinzugefügt werden. Die Syntax des Befehls lautet wie folgt:

cvs tag *[Optionen] Tag Datei-/Verzeichnisname*

Soll ein Tag wieder entfernt werden, so muß die Option **-d** angegeben werden.

Arbeiten mehrere Benutzer an den selben Daten, empfiehlt es sich, zum markieren von Versionen eine Konvention festzulegen, z.B.: **version_xx.xx.xxxx** oder **version_x.x.x**, wobei für **x** jeweils eine Zahl eingesetzt wird. Damit kann beim Suchen das korrekte Tag verwendet werden und es müssen nicht erst noch unterschiedliche Schreibweisen durchprobiert werden.

2.2.6 ... Daten löschen?

Zum Löschen einer Datei muß zuerst die lokale Kopie der Datei manuell entfernt werden. Dann muß CVS mitgeteilt werden, daß die Datei aus dem Repository gelöscht werden soll. Dafür verwendet man das Kommando **remove** mit der Syntax:

cvs remove *[Optionen] Dateiname*

Um die Dateien endgültig zu löschen, muß erst noch der Befehl **commit** ausgeführt werden.

2.3 ... hin und wieder ...

Im folgenden soll auf Möglichkeiten hingewiesen werden, die CVS bietet und die hin und wieder benötigt werden. Eine kurze Beschreibung soll dem Benutzer helfen, zumindest die richtigen Befehle für die gewünschte Funktionalität zu finden. Gegebenenfalls müssen ausführliche Informationen dann noch an anderer Stelle nachgelesen werden.

2.3.1 ... ältere Versionen wiederherstellen?

Um Kopien von älteren Versionen zu erhalten, muß der Befehl **checkout** mit der Option **-D *Datum***[1] oder **-r *Revision/Tag*** ausgeführt werden.
-D *Datum* liefert die neueste Revision, die älter als das angegebene Datum ist.
Mit **-r *Revision/Tag*** wird die Version zurückgeliefert, die mit der angegeben Revisions-Nummer bzw. dem angegebenen Tag annotiert ist.

2.3.2 ... Verzweigungen erstellen?

CVS bietet die Möglichkeit, mehrere Zweige einer Datei nebeneinander zu bearbeiten. Dies bietet die Möglichkeit, z.B. Fehler in einer älteren Version zu beseitigen, wenn diese aus Gründen der Funktionalität nicht durch die aktuelle Version ersetzt werden kann.
Zum Erstellen einer Verzweigung dient das Kommando **tag** mit der Option **-b**.

[1] Zur Angabe des Datums sind folgende Formate möglich:

- ***YYYY-MM-DD***
- ***YYYY-MM-DD HH:MM***
- ***DD Monat YYYY***
- ***Monat/Tag/Jahr***
- oder relative Angaben (auf Englisch), wie: **10 days ago**, **4 years ago**, ...

2.3.3 ... Verzweigungen wieder zusammenführen?

Nach der Beseitigung von Fehlern in einer älteren Version, sollen diese eventuell auch in der aktuellen Version, die parallel dazu weiterentwickelt wurde, beseitigt werden. Dazu werden die beiden Verzweigungen miteinander verbunden.
Um Verzweigungen zusammenzuführen wird der Befehl **update** unter Verwendung der Option(en) **-j** ***Revision1*** **-j** ***Revision2*** ausgeführt.

Kapitel 3

Was will mir CVS hier wohl sagen?

In diesem Kapitel sollen die Status-Meldungen von CVS kurz erläutert werden.

- **Locally Modified**
  Die lokale Datei unterscheidet sich von der Version im Repository.

- **Locally Added**
  Die Datei existiert nur im lokalen Arbeitsverzeichnis, nicht im Repository. Sie wurde zum Hinzufügen vorgemerkt und wird beim nächsten Aufruf von **commit** ins Repository eingefügt.

- **Locally Removed**
  Die Datei wurde zum Löschen vorgemerkt, existiert aber noch immer im Repository. Beim nächsten Aufruf von **commit** wird der Löschvorgang ausgeführt.

- **Needs Update**
  Im Repository existiert eine neuere Version der Datei (z.B. durch veröffentlichte Änderungen eines anderen Benutzers). Durch einen Aufruf von **update** wird die lokale Datei entsprechend aktualisiert.

- **Needs Patch**
  Im Repository existiert eine neuere Version der Datei, ein Aufruf von **update** liefert jedoch nur eine Korrektur, anstelle der gesamten Datei.

- **Needs Merge**
  Ein Abgleich der lokalen Datei mit der Version aus dem Repository ist notwendig (z.B. weil lokal Änderungen vorgenommen wurden, während ein anderer Benutzer seine Änderungen veröffentlichte). Der Aufruf von **update** bewirkt das Einfügen der Änderungen aus dem Repository in die lokale Datei. Im Fall eines Konflikts, wenn z.B. in beiden Datei-Versionen die selben Zeilen geändert wurden, wird der Status der Datei auf **Conflict** gesetzt.

- **Up to Date**
  Die Datei ist mit der Version im Repository identisch.

- **Conflict**
  Die Datei enthält Konflikt-Markierungen.
- **Not In CVS**
  Die Datei ist in CVS nicht registriert. Sie kann mit dem Befehl **add** zum Repository hinzugefügt und damit anderen Benutzern zugänglich gemacht werden.
- **Updated**
  Die Datei wurde aus dem Repository aktualisiert.
- **Patched**
  Der CVS-Server hat eine Korrektur für diese Datei übermittelt, die erfolgreich angewendet wurde. War die Korrektur nicht erfolgreich, wird der Status der Datei auf **Conflict** gesetzt.

Nach der Ausführung bestimmter Aktionen liefert CVS den Status der bearbeiteten Datei(en) zurück. Die hierbei verwendeten Abkürzungen werden in der nachfolgenden Tabelle kurz erläutert:

?	Die Datei ist CVS unbekannt, d.h. sie ist weder im Repository vorhanden, noch in der Verwaltungsdatei **cvsignore** eingetragen.
A	Die Datei wurde mit dem Befehl **cvs add** hinzugefügt, das Kommando **cvs commit** wurde jedoch noch nicht ausgeführt.
C	Bei dem Kommando **cvs commit** trat ein Konflikt auf, da die Datei im Repository nach dem letzten Abgleich mit der lokalen Datei (von einem anderen Benutzer) verändert wurde.
I	Die Datei wird von CVS ignoriert.
L	Bei der Datei handelt es sich um einen symbolischen Link. Sie wird von dem Befehl **cvs import** ignoriert.
M	Die Datei wurde lokal verändert. Sie unterscheidet sich von der Version im Repository.
N	Die Datei wurde neu zum Repository hinzugefügt.
P	Der CVS-Server hat eine Korrektur für diese Datei übermittelt, die erfolgreich angewendet wurde.
R	Die Datei wurde mit dem Befehl **cvs remove** entfernt, das Kommando **cvs commit** wurde jedoch noch nicht ausgeführt.
U	Die Datei wurde aus dem Repository aktualisiert.

Kapitel 4

Cervisia ...

4.1 ... nein, kein (lauwarmes) Bier!!!

Cervisia ist ein Open-source CVS-GUI-Frontend für KDE. Es bietet die Möglichkeit, die folgenden Arbeiten über eine graphische Oberfläche auszuführen:

- Abfragen des Status eines Verzeichnisses oder einzelner Dateien
- Aktualisieren des Status eines Verzeichnisses oder einzelner Dateien
- Filtern der angezeigten Dateien ihrem Status entsprechend
- Hinzufügen von Dateien
- Entfernen von Dateien
- Veröffentlichen von Dateien
- Hinzufügen von Beobachtungen
- Entfernen von Beobachtungen
- Einleiten des Bearbeitungsvorgangs für eine Datei
- Abbrechen des Bearbeitungsvorgangs für eine Datei
- Sperren einer Datei
- Aufheben einer Sperrung einer Datei
- Erstellen einer lokalen Arbeitskopie
- Importieren von Modulen in CVS
- Graphisches **diff** gegen das Repository und zwischen unterschiedlichen Revisionen
- Ansicht der Log-Informationen als Baum oder Liste
- Auflösen von Konflikten in einer Datei
- Markieren von Dateien mit Tags

- Erstellen von Verzweigungen
- Aktualisieren abhängig von Tag, Verzweigung oder Datum
- Editor zum Loggen von Änderungen bei Ausführen eines **commit**

Auf den Linux-Rechnern des IMS sind unterschiedliche Versionen von Cervisia installiert. Diese werden in den folgenden Abschnitten kurz beschrieben.

4.2 ... Version 1.4.1 ...

Die Version 1.4.1 von Cervisia läuft auf KDE 2.1.2, z.B. **felsenpinguin**.
Die Möglichkeiten, die Cervisia in der Version 1.4.1 bietet, sollen durch die Beschreibung der Menüeinträge in den folgenden Sektionen kurz erläutert werden.

4.2.1 ... File-Menü

Open Sandbox...	Öffnet einen Arbeitsordner im Hauptfenster
Open recent...	Bietet eine Liste der zuletzt genutzten Arbeitsordner und öffnet den gewählten.
Insert ChangeLog entry...	Öffnet den Editor zum Hinzufügen eines neuen Log-Eintrags. Dieser wird mit dem aktuellen Datum versehen.
Update	Aktualisiert die ausgewählten Dateien und ändert den Status und die Änderungs-Nummern in der Auflistung entsprechend.
Status	Bestimmt den aktuellen Status der ausgewählten Dateien und aktualisiert die Status-Anzeige entsprechend.
Open...	Öffnet die ausgewählte Datei im Editor.
Resolve...	Öffnet einen Dialog zum Auflösen von Konflikten in der ausgewählten Datei.
Commit...	Veröffentlicht die ausgewählten Dateien.
Add to Repository...	Fügt die ausgewählten Dateien zum Repository hinzu.
Add binary...	Fügt die ausgewählten Dateien als Binärdateien zum Repository hinzu.
Remove from Repository...	Entfernt die ausgewählten Dateien aus dem Repository.
Revert	Verwirft lokal durchgeführte Änderungen und ersetzt die Dateien durch die entsprechende Version aus dem Repository.
Quit	Beendet Cervisia

4.2.2 ... View-Menü

Browse Log...	Zeigt den Änderungs-Baum der ausgewählten Datei an.
Annotate...	Zeigt eine kommentierte Ansicht der ausgewählten Datei an, die zu jeder Zeile den Autor anzeigt, der die letzte Änderung vorgenommen hat.
Difference to repository...	Zeigt die Unterschiede zwischen der ausgewählten Datei und der zuletzt aktualisierten Version an.
Last Change...	Zeigt die Unterschiede zwischen der zuletzt aktualisierten Version der ausgewählten Datei und der davorliegenden Änderung an.
History...	Zeigt die durch den Server aufgezeichnete Historie von CVS an.
Unfold file tree	Öffnet alle Zweige des Datei-Baums.
Fold file tree	Schließt den Datei-Baum.

4.2.3 ... Advanced-Menü

Tag/Branch...	Markiert die ausgewählten Dateien mit einem Tag oder erzeugt eine Verzweigung.
Delete Tag...	Entfernt ein Tag von den ausgewählten Dateien.
Update to Tag/Date...	Holt die Version(en) der ausgewählten Dateien, die ein gegebenes Tag oder Datum haben.
Update to HEAD...	Holt die HEAD-Version der ausgewählten Dateien.
Merge...	Gleicht die ausgewählten Dateien mit einer angegebenen Verzweigung oder den Änderungen zweier Tags ab.
Add watch...	Fügt eine Überwachung der ausgewählten Dateien hinzu.
Remove watch...	Entfernt eine Überwachung der ausgewählten Dateien.
Show watchers...	Listet die Überwacher der ausgewählten Dateien auf.
Edit files	Entfernt den Schreibschutz von der ausgewählten Datei und erlaubt eine Bearbeitung.
Unedit files	Macht Änderungen an der ausgewählten Datei rückgängig und ersetzt die Datei durch die Version aus dem Repository.
Show editors	Listet die Bearbeiter der ausgewählten Dateien auf.
Lock files	Sperrt die ausgewählten Dateien für die Bearbeitung durch andere Benutzer.
Unlock files	Hebt die Sperrung der ausgewählten Dateien auf.
Create patch against repository	Erzeugt eine Korrektur aus den lokalen Änderungen.

4.2.4 ... Repository-Menü

Checkout...	Öffnet einen Dialog, über den eine Arbeitskopie eines Moduls aus dem Repository erstellt werden kann.
Import...	Öffnet einen Dialog, über den ein Modul ins Repository importiert werden kann.
Repositories...	Konfiguriert eine Liste häufig genutzter Repositories.

4.2.5 ... Settings-Menü

Hide all files	Zeigt keine Dateien im Baum an.
Hide up-to-date files	Zeigt keine aktuellen Dateien im Baum an.
Hide removed files	Zeigt keine gelöschten Dateien im Baum an.
Create directories update	Bestimmt, ob bei einer Aktualisierung neue Verzeichnisse aus dem Repository kopiert werden sollen, oder nicht.
Prune empty directories on update	Bestimmt, ob bei einer Aktualisierung leere Verzeichnisse gelöscht werden sollen, oder nicht.
Update recursively	Bestimmt, ob Aktualisierungen rekursiv ausgeführt werden.
Commit and remove recursively	Bestimmt, ob Veröffentlichungen und Löschungen rekursiv ausgeführt werden.
Do cvs edit automatically when necessary	Bestimmt, ob der Befehl **edit** beim Editieren einer Datei automatisch ausgeführt werden soll.
Configure Cervisia...	Öffnet einen Dialog zum Konfigurieren von Cervisia.
Configure keybindings...	Öffnet einen Dialog zum Konfigurieren der Tastenkombinationen.

4.2.6 ... Help-Menü

Contents	Startet die KDE-Hilfe, um die komplette Cervisia-Dokumentation anzuzeigen.
CVS Info	Startet die KDE-Hilfe, um die CVS-Dokumentation anzuzeigen.
About Cervisia	Zeigt Version und Lizenz von Cervisia an.

4.3 ... Version 2.1.2 ...

Auf KDE 3.2.2-8.FC.2 Red Hat (z.B. **pinguin**) läuft die Version 2.1.2 von Cervisia.
Cervisia bietet in der Version 2.1.2 eine deutschsprachige Oberfläche an. Die Bearbeitungsmöglichkeiten, die zur Verfügung stehen, sollen durch die Beschreibung der Menüeinträge in den folgenden Sektionen kurz erläutert werden.

4.3.1 ... Datei-Menü

CVS-Arbeitsordner öffnen...	Öffnet einen Arbeitsordner im Hauptfenster
Zuletzt geöffnete Arbeitsordner	Bietet eine Liste der zuletzt genutzten Arbeitsordner und öffnet den gewählten.
ChangeLog-Eintrag hinzufügen	Öffnet den Editor zum Hinzufügen eines neuen Log-Eintrags. Dieser wird mit dem aktuellen Datum versehen.
Aktualisieren	Aktualisiert die ausgewählten Dateien und ändert den Status und die Änderungs-Nummern in der Auflistung entsprechend.
Status	Bestimmt den aktuellen Status der ausgewählten Dateien und aktualisiert die Status-Anzeige entsprechend.
Bearbeiten	Öffnet die ausgewählte Datei im Editor.
Konflikte auflösen...	Öffnet einen Dialog zum Auflösen von Konflikten in der ausgewählten Datei.
Einspielen...	Veröffentlicht die ausgewählten Dateien.
Zum CVS-Archiv hinzufügen...	Fügt die ausgewählten Dateien zum Repository hinzu.
Binärdateien hinzufügen...	Fügt die ausgewählten Dateien als Binärdateien zum Repository hinzu.
Aus CVS-Archiv entfernen...	Entfernt die ausgewählten Dateien aus dem Repository.
Zurücknehmen	Verwirft lokal durchgeführte Änderungen und ersetzt die Dateien durch die entsprechende Version aus dem Repository.
Beenden	Beendet Cervisia

4.3.2 ... Ansicht-Menü

Anhalten	Bricht laufende Unterprozesse ab.
Log durchsuchen	Zeigt den Änderungs-Baum der ausgewählten Datei an.
Anmerkung...	Zeigt eine kommentierte Ansicht der ausgewählten Datei an, die zu jeder Zeile den Autor anzeigt, der die letzte Änderung vorgenommen hat.
Abweichungen vom Archiv...	Zeigt die Unterschiede zwischen der ausgewählten Datei und der zuletzt aktualisierten Version an.
Abweichungen vom Archiv (HEAD)...	Zeigt die Unterschiede zwischen der ausgewählten Datei und der HEAD-Version aus dem Repository an.
Letzte Änderung...	Zeigt die Unterschiede zwischen der zuletzt aktualisierten Version der ausgewählten Datei und der davorliegenden Änderung an.
Versionsgeschichte...	Zeigt die durch den Server aufgezeichnete Historie von CVS an.
Nur Ordner anzeigen	Zeigt nur Verzeichnisse im Baum an.
Dateien mit Status "Aktuell" ausblenden	Zeigt keine aktuellen Dateien im Baum an.
Dateien mit Status "Entfernt" ausblenden	Zeigt keine gelöschten Dateien im Baum an.
Dateien ausblenden, die nicht zum Archiv gehören	Zeigt nur Dateien an, die im CVS-Repository existieren.
Leere Ordner ausblenden	Zeigt keine leeren Verzeichnisse an.
Dateibaum ausklappen	Öffnet alle Zweige des Datei-Baums.
Dateibaum einklappen	Schließt den Datei-Baum.

4.3.3 ... Erweitert-Menü

Tag setzen/verzweigen...	Markiert die ausgewählten Dateien mit einem Tag oder erzeugt eine Verzweigung.
Tag löschen...	Entfernt ein Tag von den ausgewählten Dateien.
Tag/Datum aktualisieren...	Holt die Version(en) der ausgewählten Dateien, die ein gegebenes Tag oder Datum haben.
Auf HEAD aktualisieren	Holt die HEAD-Version der ausgewählten Dateien.
Zusammenführen...	Gleicht die ausgewählten Dateien mit einer angegebenen Verzweigung oder den Änderungen zweier Tags ab.
Status "Überwachen" setzen...	Fügt eine Überwachung der ausgewählten Dateien hinzu.
Status "Überwachen" entfernen...	Entfernt eine Überwachung der ausgewählten Dateien.
Überwacher anzeigen	Listet die Überwacher der ausgewählten Dateien auf.
Dateien bearbeiten	Entfernt den Schreibschutz von der ausgewählten Datei und erlaubt eine Bearbeitung.
Status "In Bearbeitung" zurücknehmen	Macht Änderungen an der ausgewählten Datei rückgängig und ersetzt die Datei durch die Version aus dem Repository.
Bearbeiter anzeigen	Listet die Bearbeiter der ausgewählten Dateien auf.
Datei sperren	Sperrt die ausgewählten Dateien für die Bearbeitung durch andere Benutzer.
Dateien freigeben	Hebt die Sperrung der ausgewählten Dateien auf.
Patch gegen das Archiv erstellen	Erzeugt eine Korrektur aus den lokalen Änderungen.

4.3.4 ... Archiv-Menü

Herunterladen...	Öffnet einen Dialog, über den eine Arbeitskopie eines Moduls aus dem Repository erstellt werden kann.
Importieren...	Öffnet einen Dialog, über den ein Modul ins Repository importiert werden kann.
Archive...	Konfiguriert eine Liste häufig genutzter Repositories.

4.3.5 ... Einstellungen-Menü

Werkzeugleiste anzeigen	Zeigt die Werkzeugleiste an.
Beim Aktualisieren fehlende Ordner anlegen	Bestimmt, ob bei einer Aktualisierung neue Verzeichnisse aus dem Repository kopiert werden sollen, oder nicht.
Leere Ordner beim Aktualisieren entfernen	Bestimmt, ob bei einer Aktualisierung leere Verzeichnisse gelöscht werden sollen, oder nicht.
Unterordner beim Aktualisieren einbeziehen	Bestimmt, ob Aktualisierungen rekursiv ausgeführt werden.
Unterordner beim Einspielen und Entfernen einbeziehen	Bestimmt, ob Veröffentlichungen und Löschungen rekursiv ausgeführt werden.
Bei Bedarf automatisch Status "In Bearbeitung" setzen	Bestimmt, ob der Befehl **edit** beim Editieren einer Datei automatisch ausgeführt werden soll.
Kurzbefehle festlegen...	Öffnet einen Dialog zum Konfigurieren der Tastenkombinationen.
Werkzeugleisten einrichten...	Öffnet einen Dialog zum Konfigurieren der Werkzeugleisten.
Cervisia einrichten...	Öffnet einen Dialog zum Konfigurieren von Cervisia.

4.3.6 ... Hilfe-Menü

Handbuch zu Cervisia	Startet die KDE-Hilfe, um die komplette Cervisia -Dokumentation anzuzeigen.
CVS-Handbuch	Startet die KDE-Hilfe, um die CVS-Dokumentation anzuzeigen.
Über Cervisia	Zeigt Version und Lizenz von Cervisia an.