

Organisation

AGASWeb - Projektpraktikum

Ina Grassmann, Nicole Rohrmeier, Bozena Zdunczyk

07.Mai 2003

Sommersemester 2003



Institut für Computer Visualistik
Universität Koblenz-Landau
Universitätsstraße 1, 56070 Koblenz
grassmann, niki, bozena@uni-koblenz.de
<http://www.uni-koblenz.de/~grassmann, niki, bozena>

Contents

CVS - Concurrent Version System

Allgemeines

CVS ist ein System zur Versionskontrolle, das für Softwareprojekte entwickelt wurde. Es dient bei folgenden Problemen als Werkzeug:

- mehrere Benutzer arbeiten gleichzeitig an einem Projekt und bearbeiten die gleiche Datei, dabei treten Konflikte auf.
- in einem Programm tritt ein Bug auf – seit wann gibt es den Bug, wie ist er entstanden?

Generell soll ein Versionsmanagementsystem folgendes tun:

- Änderungen in Quelltexten werden ersichtlich gemacht.
- alle Änderungen in Quelltexten werden dokumentiert.
- alle älteren Versionen sind wiederherstellbar.
- mehrere Benutzer können gleichzeitig an einem Projekt arbeiten.

CVS verwendet ein zentrales Repository. Dieses befindet sich in einem Verzeichnis auf einem Rechner. Darin sind enthalten:

- für jedes Projekt ein Verzeichnis
- ein Verzeichnis für allgemeine Konfigurationsparameter (z.B. Benutzerverwaltung)

CVS ist eine Client-Server-Anwendung. Das bedeutet:

- Das Repository wird von einem CVS-Server verwaltet.
- Die Benutzer verwenden einen CVS-Client, der wiederum über den CVS-Server auf das Repository zugreift.

Das Repository enthält ein Verzeichnis für jedes Projekt. Jede Datei wird im jeweiligen Verzeichnis mit ihrem Namen abgespeichert, allerdings mit erweiterten Informationen:

- einer allgemeine Beschreibung der Datei
- Unterschieden zu vorherigen Versionen, damit sind alle älteren Versionen wiederherstellbar
- dem Zeitpunkt jeder Änderung

- Kommentare des Entwicklers zu jeder Änderung
- dem Namen des ändernden Entwicklers
- einer Versionsnummer

Als erstes braucht man den Pfad zum CVS Repository, zu dem man sich verbinden will, dieser heißt **CVSROOT**. Es gibt verschiedene Formate für den CVSROOT-String, je nachdem ob das Repository lokal gespeichert oder übers Internet erreichbar ist.

lokales CVSROOT: `CVSROOT=/home/cvsroot`

auf einem Server: `CVSROOT=:pserver:cvs@bla.blub.de:/home/cvsroot`

Mit einem **checkout** Kommando besorgt man sich eine lokale Kopie des Projektes. Dies veranlasst den CVS-Client dazu, ein Modul (z.B. `/docs`) auszuchecken. Nach dem checkout befindet sich im lokalen Arbeitsverzeichnis ein `/docs`-Verzeichnis, das den aktuellen Quellcode enthält. In jedem Verzeichnis befindet sich außerdem ein CVS-Verzeichnis, in dem Informationen über den eigenen Account automatisch gespeichert werden. Einfach ignorieren.

Diesen Quellcode kann man jetzt ändern, kompilieren, installieren, untersuchen usw.. Diese Änderungen betreffen das entfernte Repository erstmal nicht. Gelegentlich müssen die ausgecheckten Dateien mit dem CVS-Repository synchronisiert werden, um von anderen Entwicklern veränderte Dateien in das lokale Verzeichnis neu einzubinden. Dies geschieht mit einem **update** Kommando. Außerdem muss es vor jedem einchecken veränderter Dateien ausgeführt werden, um Konflikte auszuschließen. Besteht kein Konflikt, kann man diese Datei dem Repository wieder hinzufügen. Dies geschieht mit einem **checkin** Kommando.

Projekt

In der Uni ist das der CVS-Server

```
:pserver:<ACCOUNT>@cvs.uni-koblenz.de
```

und für das Projekt gibt es ein Repository unter

```
CVSROOT = :pserver:  
<ACCOUNT>@cvs.uni-koblenz.de:/lab/as/REPOSITORIES/webSS03
```

Für unser Projekt verwenden wir das benutzerfreundlichere **gCVS**, das ihr unter Linux auf den Uni-Rechnern findet. Wie man von einem eigenen Rechner darauf zugreifen kann, und eine Doku findet ihr unter www.uni-koblenz.de/las/lehre/ss03/website.html

References