

Teil der Diplomarbeit 'Open Source Webshop - Phpeppershop'

Studenten: José Fontanil und Reto Glanzmann

Dozent: Patrick Feisthammel

Erfahrungsbericht:

SourceForge einrichten



Bericht

Dieser Bericht ist ein Auszug aus der kompletten Diplomarbeitsdokumentation. Sie enthält Erfahrungen die wir gerne weitergeben möchten.

Inhaltsverzeichnis

SourceForge einrichten.....	3
Einleitung.....	3
SourceForge einrichten.....	3
Eigener Account.....	3
Ein neues Projekt lancieren.....	3
Das neue Projekt einrichten.....	3
Übersicht des Projekts.....	4
Trove Categorisation.....	4
Public Information.....	4
Entwickler einrichten.....	5
Webseiten für das Projekt aufsetzen.....	5
Mailinglisten anlegen.....	5
News-Foren.....	6
Dokumentation.....	6
(Bug-, Support-, Patch-, Feature-Request-) Tracker.....	6
CVS einrichten.....	7
Was ist CVS?.....	7
CVS Entwicklungszyklus.....	7
CVS für sein Projekt auf SourceForge einrichten.....	7
Den Quellcode ins Repository importieren (Import).....	8
Arbeitskopie aus dem Repository auschecken (Checkout).....	8
Upgedatete Datei zurück ins Repository schreiben (Update, Commit).....	9
Eine neue Datei ins Repository einfügen (Add, Commit).....	9
Webzugriff auf das Repository (Diff, ...).....	9
Eine Datei oder ein Verzeichnis entfernen.....	10
Eine Datei verschieben.....	10
CVS Id-Tag.....	10
Weitere Services von SourceForge.....	10
Aufgaben (Task Management).....	10
Surveys.....	11
Compile Farm.....	11
MySQL Datenbank.....	11
Virtual Hosts Service.....	11
Fragen zu SourceForge.....	11

SourceForge einrichten

Einleitung

In den folgenden Kapiteln beschreiben wir chronologisch unsere Erfahrungen, wie wir unser Projekt auf SourceForge eingerichtet haben, welche Hürden wir nehmen mussten und auf was man achten muss, wenn man selber ein Projekt auf SourceForge einrichten will.

SourceForge bietet Open Source Projekten eine kostenlose, sehr umfangreiche und international ausgerichtete Entwicklungsumgebung an. Geboten wird unter anderem ein CVS Repository, Mailinglisten, eine Homepage, Newsgroups und vieles mehr.

SourceForge einrichten

Eigener Account

Um ein Open Source Projekt überhaupt erst lancieren zu können, muss man sich zuerst unter <http://www.sourceforge.net/> anmelden. Dies macht man am einfachsten mit einem Mausklick links oben auf 'Neuer Benutzer (SSL)'. Die folgenden einzugebenden Felder sind selbst-erklärend. Wenn man dann sein Login hat und sich einloggt, findet man auf der linken oberen Ecke eine Option 'Neues Projekt'. Auf diese Weise kann man sein neues Projekt einrichten.

Ein neues Projekt lancieren

Nun muss man dem SourceForge Terms of Service Agreement zustimmen. Vor allem Punkt 6 und 7 sind interessant. Unter anderem wird dort beschrieben, dass SourceForge keinerlei Haftung für Datenverlust übernimmt. Es ist also trotz allem ratsam immer ein aktuelles Backup zuhause zu haben. Der Punkt 7 der TOS widmet sich der Account- respektive Project Termination. Hierbei sei nur erwähnt, dass man grundsätzlich nicht von SourceForge verbannt wird, wenn man sich halbwegs ordnungsgemäss verhält. Wir wollen hier nicht alle Paragraphen besprechen, aber Punkt 5 finden wir erwähnenswert: Man stimmt mit dem Akzeptieren dieses Vertrags zu, dass man mit den Ressourcen von SourceForge keinerlei kommerzielle Ziele verfolgt.

Nun muss man seinem Projekt zuerst einmal einen Namen geben und es auf Englisch beschreiben. Diese Beschreibung ist Grundlage für die Zulassung des Projekts durch ein Beurteilungsteam von SourceForge. Wenn es unter einer Open Source Lizenz steht und frei verfügbar ist, sollte das aber ohne Probleme klappen. Weiter muss noch eine Lizenz gewählt werden (GPL oder LGPL dürften wohl die am meisten verwendeten Lizenzen sein) und es muss eine kleine Beschreibung angegeben werden. Die Public Description kann nur 255 Zeichen lang sein und wird bei Suchresultaten zu Projekten mit angezeigt. Zu guter letzt muss man noch einen Unix-Namen des Projekts angeben (die beschriebenen Restriktionen beachten).

Jetzt braucht man ein bisschen Geduld. Unser Projekt wurde innerhalb von 12 Stunden akzeptiert und von SourceForge eingerichtet, das kann aber zuweilen auch bis zu 24 Std. dauern. Wenn man sich nun wieder einloggt, findet man unten rechts sein Projekt.

Das neue Projekt einrichten

Nach einem Klick auf das Projekt befindet man sich auf der Projekt Übersichtsseite. Von hier aus komplettiert man zuerst einmal die Projekt-Konfiguration. Man hat eine reiche Auswahl an angebotenen Dienstleistungen, welche alle konfiguriert und eingerichtet werden wollen.

Übersicht des Projekts

Summary

[Überblick](#) | [Admin](#) | [Homepage](#) | [Forums](#) | [Tracker](#) | [Bugs](#) | [Support](#) | [Patches](#) | [Mailinglisten](#) | [Aufgaben](#) | [Dokumentation](#) | [Neuigkeiten](#) | [CVS](#) | [Filesgroup](#)
[short_forum](#) | [Foren](#)

Anfangen sollte man beim 'Admin'-Teil. Im Misc. 'Project Information'-Fenster sieht man in welchem Verzeichnis sich was befindet. Bei uns sah die Verzeichnisstruktur wie folgt aus:

```

Homepage Link:                phpeppershop.sourceforge.net
Group shell (SSH) server:     phpeppershop.sourceforge.net
Group directory on shell server: /home/groups/p/ph/phpeppershop
Project WWW directory:       /home/groups/p/ph/phpeppershop/htdocs
    
```

Man hat also sicher schon mal eine Homepage für das Projekt (100MB Webspace, CGI und PHP-Scripts sind auch erlaubt), ein CVS Repository (dazu später mehr) und Zugriff auf einen Mailinglisten-Server. Natürlich darf auch eine Shell nicht fehlen. Mit Telnet kommt man aber nicht weit, SourceForge erlaubt klugerweise nur SSH (SSH1 und SSH2).

Wenn man sich das erste mal auf die Shell des SourceForge Servers einloggt, (ssh projektname.sourceforge.net) wird einem der Fingerprint des jeweiligen Servers angezeigt. Man sollte sich kurz die Zeit nehmen und diesen mit dem jeweiligen Fingerprint hier vergleichen. Das beste Sicherheitstool nützt nichts, wenn man es nicht richtig anwendet:

Host	Protocol	Key Fingerprint
shell.sourceforge.net	SSH1	33:50:33:91:b4:ec:7a:b6:fa:39:99:b0:ff:65:63:a7
shell.sourceforge.net	SSH2	4c:68:03:d4:5c:58:a6:1d:9d:17:13:24:14:48:ba:99
cvs.sourceforge.net	SSH1	ff:f1:42:ca:ac:ae:a7:c8:e8:52:45:65:a1:a0:0a:b6
cvs.sourceforge.net	SSH2	02:ab:7c:aa:49:ed:0b:a8:50:13:10:c2:3e:92:0f:42

Bevor man weiterfährt, sollte man sich gleich einmal einloggen und zwar auf dem 'Projekt Server': projektname.sourceforge.net und ganz wichtig auch auf dem CVS-Server cvs.projektnamen.sourceforge.net. Man wird das CVS-System *nicht* benutzen können, wenn man sich nicht vorher einmal per SSH darauf eingeloggt hat (erst da wird das benötigte Home-Verzeichnis erstellt).

Trove Categorisation

Man kann sich jetzt der Trove Categorisation seines Projekts widmen (auf [Edit] klicken). Hier kann man sein Projekt beschreiben und kategorisieren, sodass Leute, die nach Projekten mit gewissen Eigenschaften suchen, auch fündig werden. Wenn ein Projekt nicht eindeutig zugeordnet werden kann, darf man mehrere Einteilungen benutzen. SourceForge erlaubt bis zu drei Einteilungen gleichzeitig.

Public Information

Im Admin-Bereich klicken wir nun auf 'Edit Public Info'.

[Admin](#) | [User Permissions](#) | [Edit Public Info](#) | [Project History](#) | [VHOSTS](#) | [Edit/Release Files](#) | [Post Jobs](#) | [Edit Jobs](#) | [Edit Multimedia Data](#) | [Database Admin](#)

Hier sieht man die öffentlich zugänglichen Dienste für dieses Projekt und die Beschreibung dazu. Hier muss natürlich ein jeder selber wählen, was für sein Projekt alles benötigt wird. Mailinglisten, Foren und CVS wird wohl jeder wählen, Surveys wahrscheinlich eher weniger. Es ist erstaunlich, was SourceForge einem hier alles anbietet, und zwar gratis und franko. Bleibt nur die Frage, ob das von VA Linux gesponserte System auch für immer bestehen bleibt (216 Mio. \$ Verlust bei 16 Mio. \$ Umsatz im letzten Quartal...) Unserer Meinung nach

wäre SourceForge eine Einrichtung, die vom Staat getragen werden müsste.

Entwickler einrichten

Als nächstes werden alle am Projekt beteiligten Mitarbeiter eingefügt und ihren Usern die entsprechenden Rechte geben:

1. Im Group Members Fenster den entsprechenden Usernamen eingeben und 'Add User' wählen.
2. In den 'User Permissions' (oben) kann man dem jeweiligen User Rechte und Ämter geben.

Webseiten für das Projekt aufsetzen

Jedes Projekt braucht natürlich seine Homepage. Hier wollen wir nicht darauf eingehen wie man seine Page gestalten sollte, sondern nur, wohin man sie ablegen muss und einige Punkte, welche man nicht vergessen sollte.

In der Projektübersicht (im Admin Modus) sieht man den Project WWW Directory Path. Dorthin muss man seine Seiten ablegen.

Am besten loggt man sich in SourceForge per SSH ein:

```
ssh -l Username Projektname.sourceforge.net
```

Danach wechselt man in das Project WWW Directory. Dort kann man dann mit dem Secure Copy Befehl von SSH die Daten hochladen. Beispiel:

```
scp Datei Username@shell.sourceforge.net:/home/groups/Projektname/htdocs
```

```
scp test.pl fontajos@shell.sourceforge.net:/home/groups/p/ph/phpeppershop/htdocs
```

Das sollte auf einer Projekt-Webpage alles enthalten sein:

- Einen Link auf die SourceForge Projektseite (z.B. <http://sourceforge.net/projects/phpeppershop/>)
- Links zu den Mailinglisten (die gleich angelegt werden)
- SourceForge Icon
- Einen Link zum Webinterface des CVS-Servers:
<http://cvs.sourceforge.net/cgi-bin/cvsweb.cgi/?cvsroot=Projektname>)
- Projektspezifische Daten (z.B. ein Howto für CVS oder ähnlich)

Wer direkt Projektdaten auf der Homepage darstellen will, kann dies auch tun. Genauerer findet man hier: https://sourceforge.net/docman/display_doc.php?docid=1502&group_id=1

Mailinglisten anlegen

Mailinglisten und Foren sind die Kommunikationsmittel der verteilten Entwickler. Es wird empfohlen mindestens folgende drei Standard-Listen für sein Projekt anzulegen:

- announce Hier werden neue Versionen angekündigt
- devel Eine Mailingliste für Entwickler
- users Dasselbe für die Benutzer

Mailinglisten anlegen und verwalten kann man über das 'Mailinglisten' Menü in der Projektübersicht. Wenn man auf 'Admin' klickt, kann man neue Mailinglisten anlegen oder bestehende administrieren (ist sie öffentlich, Beschreibung des Zwecks, ...).

Man hat noch die Wahl, seine Mailinglisten im 'GNU Mailman' zu administrieren. Dort kann

man sich mit dem per E-Mail erhaltenen Passwort einloggen und seine Mailingliste ganz detailliert und komfortabel konfigurieren.

News-Foren

News-Foren zum Projekt lassen sich per Klick auf 'Forums' in der Projektübersicht administrieren. In den Admin Teil kommt man über 'Admin'. Dort kann man die Foren konfigurieren. Drei News-Foren waren für uns schon automatisch eingerichtet worden.

- Developers
- Help
- Open Discussion

Es wurde noch ein weiteres, spezielles Forum angelegt: **Neuigkeiten**. Der Zweck dieses Forums ist analog zur announce-Mailingliste.

Dokumentation

Wie koordiniert man die Dokumentation der einzelnen Funktionen und Module, welche von verschiedenen Entwicklern von irgendwo auf der Welt programmiert wurden? SourceForge bietet hier mit der Dokumentationsfunktionalität eine gute Lösung. Jeder Entwickler kann seine Dokumentation in verschiedenen Gruppen organisiert (z.B. Deutsch, Englisch,...) posten und jeder hat danach Einsicht, so muss nicht alles im Code sein.

Eine kleine aber wichtige Anmerkung dazu: Die Dateien die upgeloaded werden, müssen zuerst von einem Administrator begutachtet und für gut befunden werden, erst dann sind sie für alle ersichtlich. Auch muss man aufpassen, wenn man Plain-Text Berichte uploaden will. Alle Umbrüche sind dann weg, weil das Dokument wie eine HTML-Seite betrachtet wird (Stichwort
-Tag). Auf diese Weise hat man aber gute Gestaltungsmöglichkeiten.

(Bug-, Support-, Patch-, Feature-Request-) Tracker

Der Tracker ist ein Support-Management Tool. Man kann mit ihm 'Tickets' erstellen und an die Entwickler des Projekts senden. Für den betroffenen Entwickler, erscheint dieses mit einer Priorität versehen und nach Themen sortiert auf seiner personalisierten Einstiegsseite.

Um den Tracker zu konfigurieren kann man als Projekt-Administrator von der Projekt Übersichtsseite aus auf 'Tracker' klicken und dort die entsprechende Sparte wählen. Wenn man jetzt auf 'Admin' klickt, kann man den Tracker konfigurieren. Es ist wichtig, dass man das am Anfang macht, da die Leute ihre Meldungen sonst nicht sinnvoll einordnen können. Ausserdem geht unkonfiguriert jedes Ticket an alle Admins, was meistens wohl auch nicht erwünscht ist.

ACHTUNG: Einmal eingerichtete Kategorien und Gruppen können *nicht* mehr entfernt werden.

Die einzelnen Tracker-Listen können über 'update preferences' noch zusätzlich konfiguriert werden. Unter anderem kann man dort auch festlegen, ob diese Liste überhaupt zur Verfügung stehen soll.

So, jetzt kommt nur noch CVS.

CVS einrichten

Was ist CVS?

Wenn man mit CVS (Current Versions System: <http://www.cvshome.org/>) noch keine Erfahrung hat, braucht man ein bisschen Eingewöhnungszeit. Diese sollte man aber tunlichst auf sich nehmen. Die Vorteile einer CVS-gestützten Software-Entwicklung gegenüber einer ohne CVS, sind vergleichbar mit einem Zirkus-Seiltänzer ... mit und ohne Netz! Wer einmal mit CVS gearbeitet hat, wird es garantiert nicht mehr missen wollen.

Was macht CVS eigentlich? CVS führt Buch über die Veränderungen, die am Sourcecode vorgenommen wurden, zusammen mit einer Beschreibung dessen, was verändert wurde. Man kann sich zu jeder Zeit von jedem Ort die aktuelle und jede vorangegangene Version jeder Datei im Repository ansehen. Das CVS-Repository ist eine Datenbank, in welcher alle Daten abgelegt werden. Durch CVS können mehrerer Entwickler zur gleichen Zeit an der gleichen Datei arbeiten und ihre Änderungen zusammenführen. Wenn zwei Änderungen sich gegenseitig beeinflussen, werden diese Entwickler darauf hingewiesen und es wird sichergestellt, dass der Konflikt behoben wird, bevor die neue Version der Datei im Repository aufgenommen wird.

CVS Entwicklungszyklus

Im Wesentlichen besteht der Entwicklungszyklus aus folgenden drei Schritten:

1. Man importiert den initialen Sourcecode ins Repository. Anschliessend checkt jeder Entwickler eine Arbeitskopie aus dem Entwicklungsbaum aus.
2. Nun arbeitet jeder Entwickler in seiner gewohnten Umgebung am Programm. Sobald ein neues Feature fertig ist, aktualisiert man zunächst die lokale Kopie, um auf dem laufenden zu sein und schreibt die veränderten Dateien zurück in den Entwicklungsbaum. Siehe dazu auch 'Update-Policy' weiter unten.
3. Sollten beim Zurückschreiben der Veränderungen irgendwelche Konflikte auftreten, wird man darauf hingewiesen und muss anschliessend manuell entsprechende Änderungen vornehmen. In den Dateien werden die veränderten Bereiche mit ">>>>" hervorgehoben und beide Versionen angezeigt. Hier löscht man entweder einfach die alte Version, oder man passt den Bereich an, damit er wie gewünscht arbeitet. Wenn alle Konflikte beseitigt sind, geht man wieder zurück zu Punkt 2 und setzt seine Arbeit fort.

CVS für sein Projekt auf SourceForge einrichten

Um mit CVS zu arbeiten und es für sein Projekt auf SourceForge einzurichten, benutzt man verschiedene Kommandozeilen Befehle. Wenn es jemand vorzieht unter Windows mit CVS zu arbeiten, so kann er z.B. WinCVS verwenden. WinCVS bietet noch nicht die komplette Funktionalität des UNIX Kommandozeilen Pendant, es kann aber für den täglichen Gebrauch gut verwendet werden. Weitere Angaben zu WinCVS und SourceForge findet man unter: http://sourceforge.net/docman/display_doc.php?docid=766&group_id=1. Für die weitere Beschreibung gehen wir auf die UNIX Kommandozeilen Implementation von CVS ein.

Bevor man nun seinen bestehenden Code ins CVS Repository von SourceForge importiert, noch ein wichtiger Hinweis: Die Verzeichnisstruktur, welche man dem CVS System übergibt, ist 'für die Ewigkeit' gedacht. Man kann nicht einfach wieder ein Verzeichnis löschen, nachdem man es importiert hat! Wir mussten z.B. den Support von SourceForge darum bitten unser CVS-Repository entsprechend zu ändern, nachdem uns ein Faux-pas passiert ist. Man sollte sich deshalb vorher genau überlegen, was man auf CVS veröffentlicht und wie man das

gerne strukturiert haben möchte.

Weiter macht es Sinn, nicht einfach den Source Code in das CVS-Root-Verzeichnis des jeweiligen Projekts abzulegen. Man sollte vielmehr eine kleine Struktur verwenden. Die meisten Projekte haben mehrere Verzeichnisse und manchmal sogar noch Informations-Dateien im CVS-Root. Sinnvoll erscheint uns ein Verzeichnis namens `/src` für den Source Code und ein weiteres für die Dokumentation, meistens `/doc` genannt. Hier lohnt es sich, einmal bei schon bestehenden Projekten vorbeizuschauen und sich anzusehen, wie sich diese Projekte organisiert haben. Wenn man Dateien hinzufügt, welche lediglich informativen Charakter besitzen, sollte man diese mit Grossbuchstaben schreiben, da sie von UNIX dann am Anfang der Dateiauflistung dargestellt werden.

SourceForge akzeptiert, wie schon weiter oben erwähnt, nur SSH Übertragungen. Folglich muss auch bei CVS SSH verwendet werden. Um CVS mitzuteilen, dass es SSH benutzen soll, ist es, eine Variable zu exportieren (man kann auch `setenv` (Linux: `export`) verwenden):

```
export CVS_RSH=ssh
```

Damit dieser Eintrag nach jedem Login zur Verfügung steht, empfiehlt es sich, diese Zeile der Datei `/etc/profile.local` hinzuzufügen.

Den Quellcode ins Repository importieren (Import)

Einen CVS Import macht man grundsätzlich nur zu Beginn eines Projekts, um den vielfach schon vorhandenen Code ins CVS-Repository aufzunehmen.

Zuerst wechselt man in das Verzeichnis, dessen Unterverzeichnisse und Dateien ins CVS-Repository importiert werden sollen. Alles was jetzt im aktuellen Verzeichnis und in allen Unterverzeichnissen vorhanden ist, wird ins Repository übernommen!

Man muss folgendes Kommando benutzen um den Code zu importieren:

```
cvsv -d:ext:Loginname@cvs.Projektname.sourceforge.net:/cvsroot/Projektname  
import Verzeichnisname vendor start
```

Wobei der `Loginname` und der `Projektname` entsprechend ersetzt werden müssen. Mit dem `Verzeichnisnamen` ist das Verzeichnis gemeint, in welchem die Daten auf dem CVS-Repository abgelegt werden sollen.

Nachdem man das Passwort eingegeben hat, erhält man noch die Möglichkeit, eine Message einzugeben (es wird der UNIX Editor `vi` geöffnet). Wenn man nichts mitgeben möchte, tippt man `:q!` Um den Editor wieder zu verlassen. Wer eine Message eingeben möchte, sollte sich doch zuerst mit dem `vi`-Editor befassen.

Arbeitskopie aus dem Repository auschecken (Checkout)

Ein grosser Vorteil von CVS ist, dass man von überall auf der Welt an seinem Projekt weiter arbeiten kann. Man muss lediglich ein Checkout machen und schon hat man seine aktuelle Arbeitskopie des Projekts.

Als eingetragener Entwickler:

```
cvsv -d:ext:Loginname@cvs.Projektname.sourceforge.net:/cvsroot/Projektname  
co Verzeichnisname
```

Wobei hier auch wieder der `Loginname` und der `Projektname` entsprechend ersetzt werden müssen. Mit dem `Verzeichnisnamen` ist das Verzeichnis gemeint, welches man vom Repository herunterladen möchte.

Als anonymer Benutzer

```
cvs -d:pserver:anonymous@cvs.Projektname.sourceforge.net:/cvsroot/Projektname  
login
```

Nachdem man sich nun als anonymous eingeloggt hat, kann man das eigentliche Checkout machen:

```
cvs -d:pserver:anonymous@cvs.Projektname.sourceforge.net:/cvsroot/Projektname  
co Verzeichnisname
```

Hier muss auch der **Loginname** und der **Projektname** entsprechend ersetzt werden. Mit dem **Verzeichnisnamen** ist das Verzeichnis gemeint, in welchem die Daten auf dem CVS-Repository abgelegt werden sollen.

Upgedatete Datei zurück ins Repository schreiben (Update, Commit)

Hier geht es darum ein weiter entwickeltes File, oder mehrere Dateien, zurück in das CVS-Repository zu schreiben.

Man sollte sich vorher aber auf eine Update-Policy einigen. Diese Policy sollte regeln, *wann* es sinnvoll ist, eine neue Version ins Repository zurück zu schreiben. Wir haben uns z.B. darauf geeinigt, dass wir immer erst dann ein Update machen, wenn wir eine neue lauffähige Version unseres Moduls oder unserer Funktion haben.

Da man mit dem Checkout-Befehl schon alle Angaben zum Projekt angegeben hat (-d) muss man sie jetzt nicht mehr explizit angeben:

```
cvs update  
cvs commit -m "Message"
```

An die **Message** kann man eine kleine Nachricht mitgeben. Hier drin beschreibt man in kurzen Worten, was man am Code geändert hat. Es gibt auch Projekte, welche hier eine Message-Policy eingeführt haben.

Eine neue Datei ins Repository einfügen (Add, Commit)

Um eine neue Datei ins CVS-Repository aufzunehmen, muss man zuerst einen Checkout des Repositories lokal vorhanden haben. Darin plaziert man zuerst lokal die neue Datei am richtigen Ort im Verzeichnisbaum, programmiert vielleicht schon die Funktionalität hinein und gibt dann folgende Kommandos ein:

```
cvs add [-kb] Dateiname  
cvs commit -m "Message"
```

Mit der **Message** kann man wiederum analog zum Update eine beschreibende Nachricht mitgeben. Die Option **-kb** gibt CVS an, dass es sich um eine binäre Datei handelt. Man muss binäre Dateien (z.B. Word-Dateien, ...) als solche kennzeichnen, weil CVS nicht-binäre Dateien liest und gegebenenfalls gewisse Tags ersetzt.

Webzugriff auf das Repository (Diff, ...)

Um Unterschiede zwischen Versionen von Dateien begutachten zu können, haben wir uns entschieden, das von SourceForge bereitgestellte ViewCVS einzusetzen. Dieser Service erlaubt es, neben weiteren Funktionen, das CVS Repository über das Web anzusehen. Über <http://cvs.sourceforge.net/cgi-bin/viewcvs.cgi/Projektname/> oder auch über die Projekt-Übersichtspage, kommt man darauf. Hier kann man bequem zwischen den Versionen der einzelnen Dateien hin und her wechseln, sie miteinander vergleichen, ihre Anzeige nach bestimm-

ten Kriterien sortieren und man hat einen sehr schönen Überblick über das Projekt-eigene CVS-Repository. Es gibt natürlich für all diese Funktionen auch entsprechende Kommandozeilen-Befehle (siehe Anhang).

Eine Datei oder ein Verzeichnis entfernen

Datei: Dies ist ein heikler Punkt von CVS. Schliesslich will man, auch wenn man eine Datei löscht, nicht deren Entwicklungsbaum verlieren. CVS bietet deshalb einen remove-Befehl an, der die Datei zwar faktisch als gelöscht markiert, sie aber nicht physikalisch vom Repository entfernt. Folgende Schritte muss man abarbeiten um eine Datei zu 'löschen':

1. Sicher stellen, dass es keine noch nicht commiteten Änderungen für das File gibt.
2. Die Datei lokal löschen (rm Dateiname)
3. cvs remove Dateiname
4. cvs commit Dateiname

Verzeichnis: Eigentlich gibt es keinen Weg für SourceForge Benutzer, Verzeichnisse zu löschen, es gibt aber auch hier einen Weg: Man löscht zuerst wie oben beschrieben alle Dateien im Verzeichnis (dabei können auch Wildcards wie z.B. * benutzt werden). Nun kann man, wenn man entweder einen Checkout oder ein Update macht, die Option -P mit angeben, was für CVS heisst, dass es leere Verzeichnisse von der Arbeitskopie entfernen soll. Es wird empfohlen *immer* mit der Option -P zu arbeiten.

Eine Datei verschieben

Um eine Datei zu verschieben, muss man lediglich die alte Datei löschen (siehe oben) und die Neue am neuen Ort erstellen. Hier ein Beispiel:

```
mv old new
cvs remove old
cvs add new
cvs commit -m "old zu new umbenannt" old new
```

CVS Id-Tag

Wenn man ein Programm mit Hilfe von CVS verwaltet, sollte man im Header jeder Programm-Datei folgenden Eintrag als Kommentar einfügen:

```
$Id$
```

CVS wird dann nach jedem Commit an dieser Stelle die aktuelle Versionsnummer, Filenamen, Datum, Zeit und Usernamen einfügen. Bei uns sieht das zum Beispiel so aus:

```
$Id: artikel_def.php,v 1.7 2001/10/26 06:57:48 fontajos Exp $
```

Weitere Services von SourceForge

Aufgaben (Task Management)

Wir sind hier nicht genauer auf das Task-Management eingegangen, weil wir es für die Zeit während unserer Diplomarbeit nicht benutzt haben. Dieser Service ist aber ab einer gewissen Projektgrösse unverzichtbar. Mit ihm kann ein Projekt Administrator noch zu erledigende Aufgaben publizieren und den Entwicklern zuweisen. Auf diese Weise ist immer ersichtlich, wer an welchem Problem arbeitet und was noch zu tun ist.

Surveys

Man kann im Rahmen seines Projekts auch Umfragen starten (Edit Public Info).

Compile Farm

SourceForge bietet seinen Benutzern diesen wirklich aussergewöhnlichen Service an. SourceForge bietet seinen Benutzern Zugriff auf verschiedene Netzwerke und damit auf verschiedene Host Systeme mit unterschiedlichsten Betriebssystemen. Auf diese Weise kann man sein Produkt auch auf anderen Betriebssystemen und Architekturen testen. Weiter lassen sich so bequem spezielle Binaries des eigenen Projekts erstellen.

MySQL Datenbank

Jedem Projekt wird auf Anfrage eine MySQL Datenbank gegeben. Diese kann frei benutzt werden. Oft wird sie dazu benutzt, eine dynamische Webseite des Projekts zu unterstützen.

Virtual Hosts Service

Man kann SourceForge als Hoster benutzen. Per Webinterface kann man eine Anfrage an SourceForge senden, auf einen gewissen DNS Namen zu antworten.

Fragen zu SourceForge

Hier findet man viele Dokumente zu SourceForge: http://sourceforge.net/docman/?group_id=1