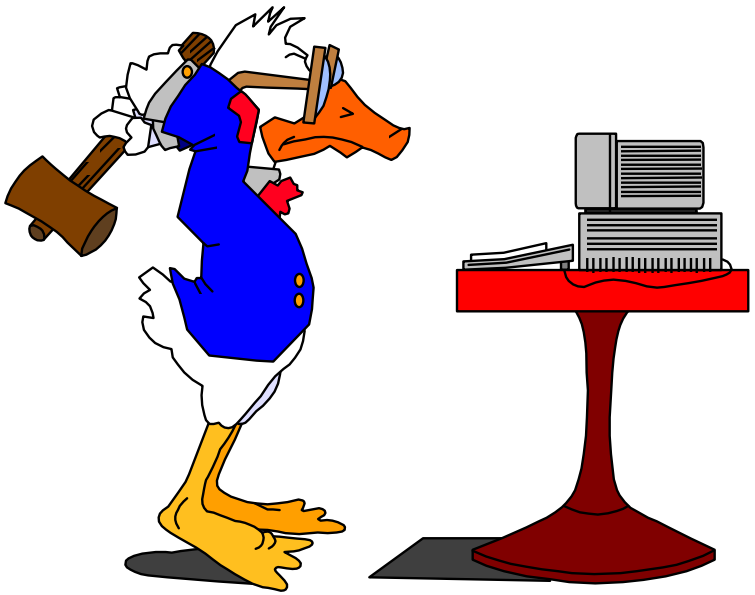
Herzlich Willkommen

Bei der Fortbildung für
CVS

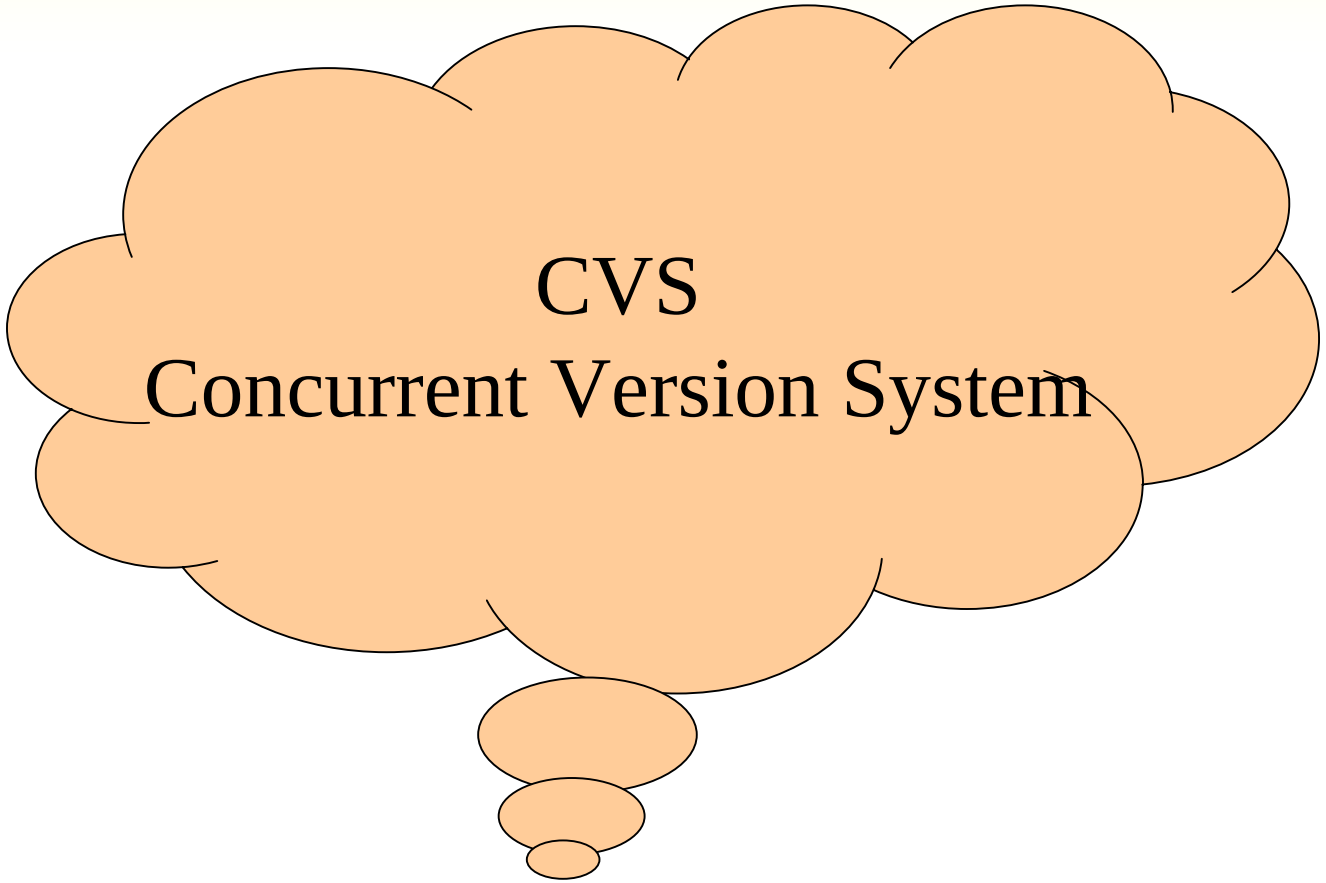
Iabc
nformation GmbH
Phone: +49 30 383983-0
http://www.Berlin.Broker.de

Die Konzentrationsfähigkeit läßt nach 40 Minuten nach!



Möchten Sie in der Mitte jedes Blockes, nach ca. 45 Minuten eine kurze Pause machen?

Iabc
nformation GmbH
Phone: +49 30 383983-0
http://www.Berlin.Broker.de



Iabc
nformation GmbH
Phone: +49 30 383983-0
http://www.Berlin.Broker.de

Zeitplan

- 1.Tag
  - Einführung und Übersicht von CVS
  - CVS Kurzanleitung
  - Die CVS Versionbibliothek
  - praktische Übungen
- 2.Tag

Zeitplan

- 2.Tag
  - Befehlsübersicht
  - Arbeiten an einer Datei
  - Übungen
    - Ein eigenes kleines Projekt
    - Ein gemeinsames kleines Projekt

CVS Concurrent Version System

Iabc
nformation GmbH
Phone: +49 30 383983-0
http://www.Berlin.Broker.de

CVS Concurrent Version System

- CVS besteht aus aufeinander abgestimmten Einzelwerkzeugen und verwendet intern RCS-Funktionen. Normalerweise wird es zusammen mit Make verwendet, um Software-Konfigurationsverwaltung zu betreiben. Es ist leicht mit andern UNIX-Werkzeugen zu kombinieren . Es realisiert eine offene, programmiersprachenunabhängige Versionsverwaltungsumgebung vom Typ Werkzeugkasten.
- Zusammen mit Make realisiert es eher ein Kompositionsmodell als das Revisionsmodell, obwohl es auf RCS basiert. Dies liegt an der systemorientierten Sichtweise bei den verwalteten Konfigurationen.

Iabc
nformation GmbH
Phone: +49 30 383983-0
http://www.Berlin.Broker.de

CVS Concurrent Version System

- Versionskonzepte
  - Gegenüber RCS ergeben sich zwei wesentliche, konzeptionelle Unterschiede:
    - die systemorientierte Sichtweise und
    - die optimistische Zugriffskontrolle.
  - Damit verbunden ist auch die Organisation der Arbeitsbereiche, die zwar außerhalb der Kontrolle von CVS liegt, deren Struktur jedoch von CVS als Abbild der Struktur innerhalb der Versionsbibliothek vorgegeben wird.
  - CVS verwaltet die Quellen eines Programmsystems in einer gemeinsamen Versionsbibliothek.

CVS Concurrent Version System

- Das System kann im Gegensatz zu RCS in Untereinheiten strukturiert werden, indem es auf unterschiedliche Unterverzeichnisse aufgeteilt wird. Diese Struktur spiegelt sich auch in den Arbeitsbereichen wider.
- CVS realisiert zusätzlich zu den RCS-Funktionen, die Komponenten auf Systemebene. Die Operationen AUSLEIHEN und EINBRINGEN (ckech in, check out),beziehen sich auf das ganze Software-System oder spezifizierbare Teile davon (modules oder Unterverzeichnisse).
- Für alle Komponenten des Systems existieren innerhalb der CVS-Versionsbibliothek einzelne „RCS files“, mit denen ein Anwender jedoch nicht direkt operiert (systemorientierte Sichtweise).

Iabc
nformation GmbH
Phone: +49 30 383983-0
http://www.Berlin.Broker.de

CVS Concurrent Version System

- Ein Vergleich zwischen den Versionen kann ebenfalls systemweit ausgeführt werden. Selbst die VERSCHMELZE-Funktion wird von CVS auf Systemebene unterstützt.
- Als ein wesentliches Einsatzgebiet von CVS bezeichnen dessen Autoren die Verwaltung von ``third-party software'', die sowohl lokal als auch vom Vertreiber modifiziert wird.
- Bei der Einspielung neuer Versionen des Vertreibers (Update), werden die lokalen Modifikationen berücksichtigt. Das Konzept von CVS sieht vor, die Original-Software auf einem Seitenast (``vendor branch'') einzuspielen, während die lokalen Versionen die Hauptentwicklungslinie bilden. Sobald ein Update eintrifft, wird dieser als neuer Seitenast eingespielt (cvs import).

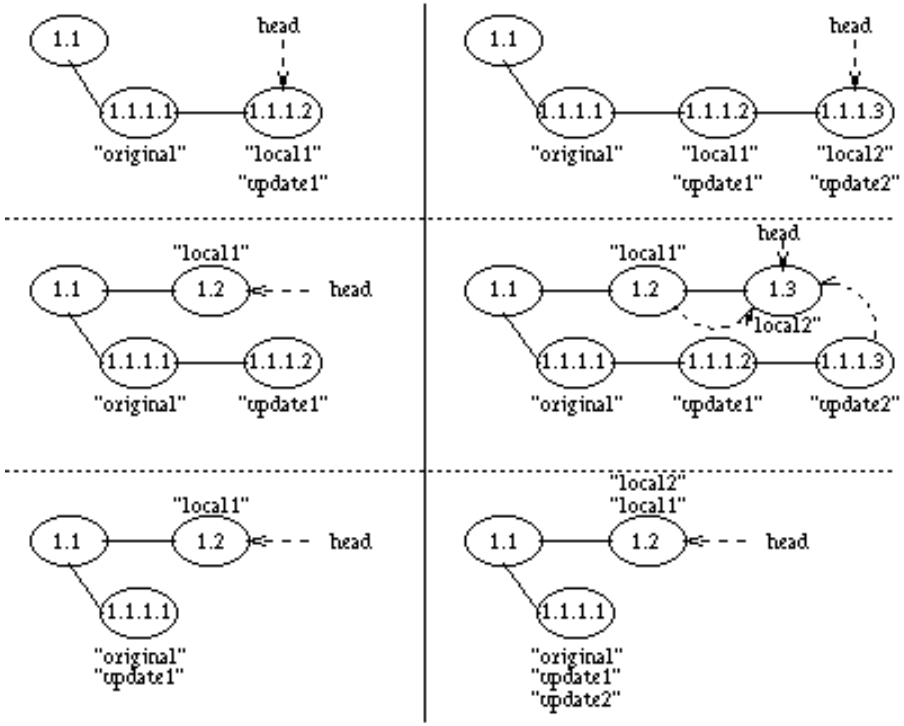
Iabc
nformation GmbH
Phone: +49 30 383983-0
http://www.Berlin.Broker.de

CVS Concurrent Version System

- Alle neuen Dateien und die lokal nicht modifizierten bilden zu diesem Zeitpunkt die aktuelle Revision (``head''), die als Standardargument allen CVS-Operationen übergeben wird. Alle lokal modifizierten Komponenten werden aktiv mit den neuen Komponenten verschmolzen. Hier bilden die so entstehenden neuen, lokalen Revisionen den Kopf der Entwicklungslinie.

CVS, ein Bespiel (1)

- Wirkung des Einspielens eines Updates



Iabc
nformation GmbH
Phone: +49 30 383983-0
http://www.Berlin.Broker.de

CVS, ein Beispiel (2)

- Wirkung des Einspielens eines Updates (siehe Beispiel)
  - Der linke Teil des Beispiels in Abbildung 1.3 stellt drei Komponenten A, B und C eines Programmes dar, das in einer Original-, einer vom Vertreiber aktualisierten (update1) und einer lokalen (local1) Konfiguration vorliegt. Die lokale Konfiguration beinhaltet jeweils die aktuellen Revisionen der Komponenten, A 1.1.1.2, B 1.2 und C 1.2, auf die jeweils auch head verweist. Das Update des Vertreibers hat die Komponenten A und B verändert, während die Komponente C nicht betroffen war. Der rechte Teil der Abbildung zeigt die entstehenden Revisionsgraphen, nachdem eine neue Version des Vertreibers importiert wird. Durch die Berücksichtigung der lokalen Modifikationen entsteht außerdem die lokale Konfiguration local2, in die alle aktuellen Revisionen eingehen. Die lokale Revision 1.3 der Komponenten B entsteht durch die Anwendung der VERSCHMELZE-Funktion cvs update.

CVS Versionsbibliothek

- Die Versionsbibliothek
  - Es gibt einen Wirkungsbereich für Änderungen, die Versionsbibliothek. Alle innerhalb eines Arbeitsbereiches lokalen Modifikationen werden wirksam, wenn die veränderten Komponenten eingebracht werden. Jede Anwenderin arbeitet auf Kopien der Komponenten innerhalb eines privaten Arbeitsbereiches. Im Gegensatz zu RCS verfolgt CVS eine optimistische Kooperationsstrategie. Beim AUSLEIHEN werden keine Sperren vergeben. Mehrere Personen können gleichzeitig auf Kopien der gleichen Revisionen arbeiten. Ein Abgleich mit wirksamen Änderungen ist jederzeit möglich. Der Vergleich und die Aktualisierung der im Arbeitsbereich befindlichen Versionen gegenüber der Versionsbibliothek werden unterstützt (diff und update). Vor dem Einbringen der eigenen Revisionen wird der Abgleich mit den in der Versionsbibliothek befindlichen Revisionen allerdings erzwungen, um zu verhindern, daß parallel durchgeführte und zuvor eingebrachte Modifikationen überschrieben werden.

CVS Funktionalität und RCS

- Allgemein bietet CVS die gleiche Funktionalität wie RCS, nur daß die Funktionen auf Systemebene realisiert werden. Die Identifikation einer Konfiguration erfolgt über die Vergabe eines symbolischen Namens. Intern vergibt CVS diesen Namen für jede an der Konfiguration beteiligte Komponente durch Verwendung der entsprechenden RCS-Funktion. Die Revisionsnummern der Komponenten werden für den Anwender nicht sichtbar.

CVS Änderungskontrolle

- Die eigentliche Änderungskontrolle erfolgt ebenfalls über RCS. Dementsprechend wird der Revisionsgraph für Komponenten über Rückwärtsdeltas speichereffizient verwaltet. Bei der Verwaltung der Entwicklungsgeschichte können zusätzlich die durchgeführten CVS-Operationen protokolliert und jederzeit durch „cvs history“ angezeigt werden.

CVS Revisionsänderung

- Die Änderung beliebiger Revisionen ist möglich. Dazu wird zunächst durch das Kommando „cvs rtag -b“ ein Seitenarm erzeugt, der beim AUSLEIHEN angegeben wird. Alle folgenden Operationen beziehen sich dann auf die Seitenentwicklung, da der head-Zeiger auf diese verweist. Der Mechanismus der parallelen Revisionen kann genau wie bei RCS auch zur Simulation unterschiedlicher Entwicklungen eingesetzt werden.

CVS, parallele Entwicklungen

- Zur Vereinigung paralleler Entwicklungen exisitieren ebenfalls Funktionen auf Systemebene.
  - „cvs diff“ zeigt die Differenzen aller Komponenten des Arbeitsbereiches, die sich gegenüber der aktuellen Revision der Versionsbibliothek unterscheiden.
  - „cvs update“ aktualisiert die ausgeliehenen Revisionen innerhalb des Arbeitsbereiches. Bei gleichzeitig veränderten Komponenten wird intern „rcsmerge“ verwendet, um einen Mischvorschlag zu erzeugen.

CVS Konfigurierung

- Die Unterstützung der Konfigurierung ist, ähnlich spartanisch wie bei RCS, auf die die Vergabe symbolischer Namen beschränkt. Eine Konfiguration wird entweder anhand dieser Namen oder mittels einer Datumsspezifikation identifiziert. Die Vergabe des Namens erfolgt, indem eine aktuelle Arbeitskonfiguration eingebracht und anschließend in der Versionsbibliothek mit einem Namen versehen wird (vgl.rcs -nName):
  - cvs commit modules ...
  - cvs rtag SymbName modules ...

CVS „patch files“

- CVS unterstützt die Freigabeverwaltung durch eine Funktion zur Erstellung eines ``patch files". Diese Datei kann später von dem Programm „patch“ verwendet werden, um eine zuvor freigegebene Konfiguration, die lokal nicht verändert wurde, automatisch auf den neuesten Stand zu bringen. Die „cvs rdiff“-Funktion berechnet die Unterschiede zwischen zwei angegebenen Versionen innerhalb der Versionsbibliothek und bietet damit eine ähnliche Funktionalität, wie die Deltaberechnung im Änderungsmodell.

CVS nicht für Systemmodellierung

- CVS bietet keinen Formalismus zur Systemmodellierung; stattdessen wird das Systemmodell implizit, in Form der Dateistruktur der Ausgangsversion der Software festgelegt. Es kann im Zuge der Weiterentwicklung durch „add“ bzw. „remove“-Kommandos inkrementell verändert werden. Die Konsistenz der Elemente einer Konfiguration und ihre Beziehungen untereinander werden durch CVS nicht geprüft, da es weder den Inhalt der versionierten Objekte betrachtet noch zusätzliche Informationen über deren Beziehung verwaltet.

Einsatz von CVS (1)

- Die Zugriffskontrolle erleichtert die parallele Arbeit an einem Software-System.
- CVS ist gut geeignet, lokale Modifikationen in den neuen Ausgaben der Hersteller nachzuladen. Ein weiterer Pluspunkt von CVS ist dessen einfache Bedienbarkeit. CVS verzichtet dafür bewußt auf einige Funktionen der Software-Konfigurationsverwaltung, die eine kompliziertere Bedienung bedingt hätte, z.B. werden beim AUSLEIHEN sofort Kopien der Revisionen angelegt und nicht nur Verweise eingetragen.

Einsatz von CVS (2)

- CVS ist nicht dazu ausgelegt, den Prozeß der Neuerstellung von Software zu unterstützen, sondern eignet sich eher dazu, bestehende weiterzuentwickeln bzw. lokal anzupassen. Das Projektmanagement bis zur Kontrolle von Fehlermeldungen sind nicht Bestandteil.
  - Dies äußert sich darin, daß keine Systemmodellierung erfolgt und die Veränderung bestehender Systemstrukturen, z.B. durch Umbenennen oder Verschieben von Dateien, nicht möglich sind. Die Koordination paralleler Entwicklungsarbeit wird ebenfalls wenig unterstützt. Es fehlen Konzepte zur Realisierung von Wirkungsbereichen von Änderungen.

CVS Kurzanleitung (1)

- CVS arbeitet mit einer Versionsbibliothek die Ihr Systemadministrator anlegen sollte. Die einzelnen Benutzerdateien des Entwicklungsbaumes werden in ihr abgelegt.
  - „cvs -d <Pfadname> init“ legt diese Struktur an, die von allen Mitgliedern benutzt werden kann.
  - Als Bereich hierfür sollte der Pfad „/usr/local/“ verwendet werden.
- CVS ist auch als Netzvariante für verteilte Entwicklungen verfügbar und kann unter UNIX sowie Windows95/NT betrieben werden.

CVS Kurzanleitung (2)

- Bei echten Client-/Serverbetrieb verwenden wir für den Einloggvorgang in das CVS-System den sogenannten Paßwortserver pserver.
- BEVOR Sie den Server starten, müssen Sie die folgenden Variablen setzen:
  - set CVS_PASSFILE=\hier Pfad zu ihrem lokalen CVS-Home-Verzeichnis\.cvspass
  - set CVSROOT=/usr/local/cvsroot_user.
- Auf einem Unix-Rechner müssen Sie die CVS_PASSFILE Variable nicht setzen. In der ksh setzen Sie also nur CVSROOT wie folgt (eventuell im .login file):
  - CVSROOT=„/usr/local/cvsroot_user“.
- Dann können Sie sich bei bestehender Internetanbindung mittels des login Befehls einloggen:
  - cvs -d :pserver:username@berlin.Broker.de:$CVSROOT login (username ist Ihr Username: BEACHTEN SIE DIE DOPPELPUNKTE!!).

CVS Kurzanleitung (3)

- Falls Sie auf einem Unix-Rechner immer das gleiche CVS repository und das gleiche Projekt verwenden (was bei den meisten von Ihnen der Fall sein wird), ist der login Befehl nur bei der aller ersten Sitzung notwendig.
- *Achtung: den Zusatz „ :pserver:username@berlin.Broker.de:“ benötigen Sie nur, wenn sie über Netzwerk von einem Client auf den Server zugreifen!*
- *Achtung: die Angabe „-d $CVSROOT“ benötigen Sie nur, wenn Ihre Systemvariable „CVSROOT“ nicht gesetzt ist oder ein anderer Pfad benutzt werden soll!*
- Wenn Sie auf Ihrem Rechner das erste Mal mit CVS arbeiten, müssen Sie sich, bevor Sie weitere Befehle eingeben, zuerst einmal eine lokale Version Ihres Projekt-Repository anlegen. Das geschieht mit dem checkout Befehl:
  - **cvs -d :pserver:username@berlin.Broker.de:$CVSROOT checkout gruppenname**
    (username ist Ihr Username, gruppenname ist Ihr Gruppenname)
- Dabei wird ein Projekt-Directory <gruppenname> auf Ihrem lokalen Rechner angelegt, das den gegenwärtigen Inhalt Ihres zentralen Projekt-Repository enthält.

CVS Kurzanleitung (4)

- Danach gehen Sie einfach in dieses Projekt-Directory und können dann weitere CVS Befehle eingeben, bzw. die darin befindlichen Befehle bearbeiten. Wenn Sie unter Unix arbeiten, können Sie diese einfach in der folgenden Form eingeben, also ohne Angabe des -d Parameters:
  - **cvs IHR_BEFEHL**

- Unter Windows 95 müssen Sie im ersten CVS-Befehl nach dem Einloggen noch einmal den -d Parameter verwenden:
  - **cvs -d :pserver:username@Berlin.Broker.de:$CVSROOT IHR_BEFEHL** (username ist Ihr Username: BEACHTEN SIE DIE DOPPELPUNKTE!!).

CVS, Beginn einer Sitzung

- Es muß zwischen einer Client/ Server Verbindung zwischen MS-Windows/ Unix und einer reinen Unix-Host Anwendung unterschieden werden:
  - Unix: Unix „Login“: <Benutzername>; Password: <Passwort>
  - $> **cvs IHR_BEFEHL**
  - MS-Windows: **cvs -d :pserver:username@Berlin.Broker.de: $CVSROOT login**
  - Sie werden danach nach Ihrem Paßwort gefragt. Bemerkung: Dieses login ist unter Unix und gleichbleibendem repository nur das erste Mal notwendig. Danach reicht es, wenn Sie einfach in das Projekt-Directory gehen. Nun können Sie am erscheinenden Prompt weiterarbeiten. CVS wird jeweils angesprochen, wenn es als erster Befehl in der Kommandozeile steht.

Übung CVS, Anlegen einer Versionsbibliothek (1)

- Ihre private Versionsbibliothek soll in dem Verzeichnis „/usr/local/CVS/<Benutzername>“ angelegt werden.
  - Setzen Sie die CVS Systemvariable „CVSROOT“ auf diesen Pfad.
  - Legen Sie Ihre private Versionsbibliothek an.
- Sehen Sie sich bitte auf Systemebene an, welche Verzeichnisse und Dateien angelegt wurden.

CVS, Arbeit an einer Datei „checkout“ und „add“

- Wir gehen davon aus, daß Sie bereits lokal ein Projekt-Directory mit dem folgenden Befehl erzeugt haben:
  - ***cvs -d :pserver:username@berlin.broker.de:$CVSROOT checkout gruppenname***
- Möchten Sie in diesem Projekt-Directory eine neue Datei oder ein neues Verzeichnis anlegen, benutzen Sie das add-Kommando, nachdem Sie die Datei erzeugt haben:
  - ***cvs add Dateiname***
- Sind die Dateien Binärdateien, z.B. WinWord Dokumente oder C++-Compilat, dann muß nach dem add die Option „-kb“ eingefügt werden.

Iabc nformation GmbH
Phone: +49 30 383983-0
http://www.Berlin.Broker.de

CVS, Arbeit an einer Datei „commit“

- Die neuen (oder schon vorhandenen) Dateien können verändert werden. Nach vollendeter Arbeit übertragen Sie die von Ihnen geänderte Datei in das zentrale epository, damit auch andere Mitglieder hiermit arbeiten können.
  - ***cvs commit -m "Eine Anmerkung" Dateiname***
- CVS sagt Ihnen nun, daß Ihre Datei die neueste Versionsnr. erhält und in das Repository aufgenommen wurde.
- Jetzt können Sie noch weitere Dateien bearbeiten etc. Jedesmal, wenn eine Datei Ihrer Ansicht nach fertig ist, benutzen Sie bitte obigen commit-Befehl, um die Datei in das Repository zu übernehmen.

Übung CVS, Anlegen einer Versionsbibliothek (2)

- Legen Sie bitte das Verzeichnis CVS01.d in Ihrem Arbeitsverzeichnis an. In diesem Verzeichnis laden Sie bitte mit dem „cvs“ Befehl die Dateien, b.z.w das Verzeichnis der Versionsbibliothek .
  - Hiermit können Sie die Konfiguration der Versionsbibliothek verändern.
- Geben Sie die Dateien, b.z.w das Verzeichnis der Versionsbibliothek wieder frei!

CVS Befehlssyntax

- Syntax:
  - cvs [cvs_options] cvs_command [command_options] [command_args]
    - *cvs* Name des CVS Programmes
    - *cvs_options* Optionen von CVS
    - *cvs_command* CVS Kommandos (commit, update etc.)
    - *command_options* Optionen der Kommandos
    - command_args Argumente der Kommandos

CVS Befehlsübersicht (1)

- admin Administrations Front-End für rcs
- checkout „check out“ Sourcen zur Bearbeitung
- commit „check in“ der Datei/en in das Repository
- diff Erstellt Deltas zwischen den Revisionen
- export Exportiert Sourcen von CVS, wie „checkout“
- history Zeigt den Status der Files und User
- import Importiert Sourcen in CVS
- log Ausgabe der „Log Informationen“ der Files
- rdiff 'patch' Format für den Unix Befehl „patch“

Iabc
nformation GmbH
Phone: +49 30 383983-0
http://www.Berlin.Broker.de

CVS Befehlsübersicht (2)

- release — Angabe, daß das Modul unbenutzt ist
- rtag — Einen symbolischen Namen (tag) dem Modul hinzufügen
- status — Anzeige der Statusinformation von „checked out“ Dateien
- tag — Einen symbolischen Namen (tag) der „checked out“ Datei hinzufügen
- update — Syncronisation des Arbeitsbereiches incl. der Verzeichnisse mit dem Repository

CVS, Arbeit an einer Datei „release“

- Wenn Sie Ihre Sitzung vollständig beenden möchten und das lokale Projekt-Directory löschen wollen, können Sie dieses dem CVS wie folgt mit dem release Befehl mitteilen:
  - ***cvs release -d gruppenname***
- Dieser Befehl löscht den gesamten Inhalt Ihres lokalen Projekt-Directory, so daß Sie beim nächsten Mal wieder den login sowie den checkout Befehl verwenden müssen. Wenn Sie keine Speicherplatzprobleme haben, ist es daher bequemer, den release Befehl nicht zu verwenden.

CVS, Löschen einer Datei/ eines Verzeichnisses (1)

- Anmerkung: Löschen Sie niemals Ihr Projekt-Directory per Hand!!
  - Löschen Sie in Ihrem lokalen Projekt-Directory die entsprechende Datei / das Verzeichnis. Haben Sie die falsche Datei im lokalen Verzeichnis gelöscht, können Sie diese mittels des update-Kommandos wiederherstellen:
    - ***cvs update Dateiname***
  - Löschen Sie dann mittels des remove-Kommandos die entsprechende Datei aus dem Repository:
    - ***cvs remove Dateiname***

CVS, Löschen einer Datei/ eines Verzeichnisses (2)

- Haben Sie die falsche Datei im Repository gelöscht, können Sie diese mittels des add-Kommandos wiederherstellen:
  - ***cvs add Dateiname***
- Mittels des commit Befehls veranlassen Sie CVS, die Datei / das Verzeichnis nun tatsächlich aus dem Repository zu löschen.
- Bemerkung: Erst der commit Befehl führt alle Änderungen, die mit anderen Befehlen vorgemerkt wurden (wie update, add, remove) tatsächlich im zentralen Repository durch.
  - ***cvs commit -m "Eine Anmerkung" Dateiname***

CVS, der Befehl „update (1)

- Der Update Befehl ist mächtiger, als die vorherigen Beispiele zeigen. Insbesondere ist es mit ihm möglich, Ihr lokales Projekt-Directory als Ganzes mit dem zentralen Projekt-Repository zu synchronisieren. Wenn Sie zum Beispiel in Ihrem Projekt-Directory einfach ***cvs update*** eingeben, wird das gesamte lokale Projekt-Directory mit Ihrem zentralen Projekt-Directory synchronisiert. Dabei gibt CVS für jede Datei, die synchronisiert wurde, die Information aus, welche Art von Änderung durchgeführt wurde. Dieser Befehl ist sehr komfortabel, Sie sollten sich aber die entsprechenden Erklärungen zum Output dieses Kommandos unbedingt durchlesen!

CVS, der Befehl „update (2)

- Da CVS im Falle von parallelen Änderungen durch mehrere Entwickler selbständiges Mergen dieser Änderungen vornimmt, ist bei solch parallelem Arbeiten an ein und derselben Datei besonders der Fall interessant, wenn dadurch Konflikte zwischen den Änderungen auftreten. Diese müssen dann beseitigt werden, bevor die Datei mit dem commit Befehl tatsächlich ins zentrale Repository zurückgespielt werden kann.
- Natürlich müssen Sie jeden update Befehl mit einem commit Befehl abschließen, damit die von Ihnen vorgenommenen Änderungen tatsächlich ins zentrale Repository aufgenommen werden.

Iabc
nformation GmbH
Phone: +49 30 383983-0
http://www.Berlin.Broker.de

Übung: Ihr eigenes kleines Projekt (1)

- Die Dateien die im Abschnitt „RCS“ erstellt wurden, sollen nun unter „CVS“ verwaltet werde.
- Legen Sie in Ihrem Arbeitsbereich ein Verzeichnis „CVS02.d“ an und kopieren Sie die alte Datei „text01.c“ in dieses Verzeichnis.
- Legen Sie die Versionsbibliothek „/usr/local/CVS/<Benutzername>01“ an.
- Geben Sie nun der Versionsbibliothek dieses neue Verzeichnis bekannt, indem sie es einspielen.
- Wiederholen Sie den gleichen Vorgang, in dem Sie die Datei jedoch „Importieren“. (Nehmen Sie die Referenzen als Hilfe)

Übung: Ihr eigenes kleines Projekt (2)

- Stellen Sie Ihre Dateien in der Versionsbibliothek ein und übergeben ihr dies als Release.
- Übernehmen Sie die Versionsbibliothek Ihres Nachbarn in ein neues Verzeichnis. Bitten Sie Ihn, Ihren Benutzernamen mit Berechtigung zu versehen. Nach einer kleinen Änderung an seiner Datei übergeben Sie diese als neues Release.

Übung: Ein gemeinsames kleines Projekt (1)

- Ziel ist es, Teilprodukte eines jeden Teilnehmers in einer gemeinsamen Versionsbibliothek zu erstellen.
- Eine gemeinsame Arbeit und Abstimmung ist hierfür als Team notwendig, unter Definition einer gemeinsamen Versionsbibliothek.
- Aufgabenstellung:
  - In die Versionsbibliothek fügt jeder Teilnehmer eine Datei mit dem Namen seiner Benutzerkennung ein.
  - Machen Sie inzwischen schon ein Update um zu sehen, ob schon Dateien in der Versionsbibliothek übergeben wurden.
  - Geben Sie der Versionsbibliothek Ihre Datei bekannt!

Übung: Ein gemeinsames kleines Projekt (2)

- Wie ist der Status der gemeinsamen Bearbeitung ?
- Wie ist die Historie der Entwicklung ?
- Führen Sie durch ein Update einen Datenausgleich durch, um die aktuellsten Dateien zu erhalten.
- Versuchen Sie die Datei eines anderen Teilnehmers zu verändern, nach dem er Sie berechtigt hat. Was für eine Meldung bekommen Sie ?
- Schließen Sie dieses Release ab und lassen Sie sich die einzelnen Revisionskommentare ausgeben.