Parallele Programmentwicklung mit CVS

Jan Hendrik Dithmar

03.11.2005

Einführung

Einführung

Bisherige Arbeitsweise:
Es gibt ein Verzeichnis, auf das alle Entwickler Zugriff haben und arbeiten können. Dabei „gewinnt“ immer die letzte Speicherung einer Datei.

Einführung

Bisherige Arbeitsweise:
Es gibt ein Verzeichnis, auf das alle Entwickler Zugriff haben und arbeiten können. Dabei „gewinnt“ immer die letzte Speicherung einer Datei.

Vorschlag der zukünftigen Arbeitsweise:
CVS (Concurrent Versions System)

Vor- und Nachteile der bisherigen Arbeitsweise

Vor- und Nachteile der bisherigen Arbeitsweise

Vorteile:

- relativ einfache Installation und Wartung
- einfache Bedienung (Arbeiten wie mit lokalen Verzeichnissen und/oder Laufwerken)

Vor- und Nachteile der bisherigen Arbeitsweise

Vorteile:

- relativ einfache Installation und Wartung
- einfache Bedienung (Arbeiten wie mit lokalen Verzeichnissen und/oder Laufwerken)

Nachteile:

- Ändern 2 Entwickler etwas an der gleichen Datei (beispielsweise der eine am Anfang, der andere am Ende) so geht die Änderung eines Entwicklers beim Speichern verloren
- Frühere Konfigurationen des Softwaresystems müssen über recht komplizierte Wege wiederhergestellt werden (z.B. mit umständlichen Backups)

Versionierung eines Softwaresystems

Wird ein Softwaresystem aus mehreren Komponenten versioniert, gilt es, zwei Ziele zu verfolgen:

Versionierung eines Softwaresystems

Wird ein Softwaresystem aus mehreren Komponenten versioniert, gilt es, zwei Ziele zu verfolgen:

Konsistenz: Es muss stets gesichert sein, dass die in Anspruch genommenen Dienste einer Schnittstelle auch tatsächlich zur Verfügung gestellt werden. D.h.: Wird eine Schnittstelle verändert, so müssen diese Änderungen *gekoppelt* sein, so dass sie nicht unabhängig voneinander angewandt werden können.

Versionierung eines Softwaresystems

Wird ein Softwaresystem aus mehreren Komponenten versioniert, gilt es, zwei Ziele zu verfolgen:

Konsistenz: Es muss stets gesichert sein, dass die in Anspruch genommenen Dienste einer Schnittstelle auch tatsächlich zur Verfügung gestellt werden. D.h.: Wird eine Schnittstelle verändert, so müssen diese Änderungen *gekoppelt* sein, so dass sie nicht unabhängig voneinander angewandt werden können.

Flexibilität: So lange die Schnittstelle unverändert bleibt, muss es möglich sein, beliebige Versionen einer Komponente auszuwählen und so ein System passend für einen bestimmten Zweck oder eine bestimmte Umgebung zu *konfigurieren*.

Versionierung eines Softwaresystems

Wird ein Softwaresystem aus mehreren Komponenten versioniert, gilt es, zwei Ziele zu verfolgen:

Konsistenz: Es muss stets gesichert sein, dass die in Anspruch genommenen Dienste einer Schnittstelle auch tatsächlich zur Verfügung gestellt werden. D.h.: Wird eine Schnittstelle verändert, so müssen diese Änderungen *gekoppelt* sein, so dass sie nicht unabhängig voneinander angewandt werden können.

Flexibilität: So lange die Schnittstelle unverändert bleibt, muss es möglich sein, beliebige Versionen einer Komponente auszuwählen und so ein System passend für einen bestimmten Zweck oder eine bestimmte Umgebung zu *konfigurieren*.

Konsistenz ist eine absolut notwendige Qualitätsbedingung für Softwaresysteme. Durch das Benennen von Konfigurationen können konsistente Konfigurationen identifiziert und zu einem späteren Zeitpunkt wiederhergestellt werden.

Was bedeutet Konsistenz bei Softwaresystemen?

Was bedeutet Konsistenz bei Softwaresystemen?

Beispiel (abstrakt):
Bei einem Softwaresystem gibt es die Komponente A und eine Komponente A', die eine Revision von A mit erweiterter Funktionalität darstellt. Eine Komponente B nutzt diese erweiterte Funktionalität von A'. B kann nicht (mehr) mit A kombiniert werden.

Was bedeutet Konsistenz bei Softwaresystemen?

Beispiel (abstrakt):
Bei einem Softwaresystem gibt es die Komponente A und eine Komponente A', die eine Revision von A mit erweiterter Funktionalität darstellt. Eine Komponente B nutzt diese erweiterte Funktionalität von A'. B kann nicht (mehr) mit A kombiniert werden.

Beispiel (konkreter):
Ein Programm „Taschenrechner“ hat eine Klasse, in der alle Berechnungen stattfinden (Komponente A). Die grafische Oberfläche (Komponente B) ruft die jeweiligen Prozeduren der Komponente A auf. A wird jetzt um die Möglichkeit erweitert, das letzte berechnete Zwischenergebnis zu speichern (und wird zu Komponente A'). Die grafische Oberfläche nutzt diesen Vorteil aus und stellt das Zwischenergebnis immer in einem Feld dar. B kann nicht mehr mit A kombiniert werden, da A diese Funktionalität nicht aufweist.

Vor- und Nachteile von CVS

Vor- und Nachteile von CVS

Vorteile:

- CVS kann ganze Dateibäume versionieren
- CVS berücksitigt das Anlegen und Löschen von Dateien
- es können mehrere Entwickler parallel entwickeln, keine Änderung geht verloren; es kann höchstens ein Konflikt auftreten (später mehr dazu)

Vor- und Nachteile von CVS

Vorteile:

- CVS kann ganze Dateibäume versionieren
- CVS berücksitigt das Anlegen und Löschen von Dateien
- es können mehrere Entwickler parallel entwickeln, keine Änderung geht verloren; es kann höchstens ein Konflikt auftreten (später mehr dazu)

Nachteile:

- Eingewöhnungszeit in die neue Arbeitsweise

CVS-Kommandos

`cvs checkout`	-	Kopieren der Quellen des Projekts aus dem CVS-Archiv in den Arbeitsbereich
`cvs commit`	-	Kopieren der aktuellen Dateiversionen aus dem Arbeitsbereich in das CVS-Archiv
`cvs tag`	-	Vergabe eines Namens für eine bestimmte Konfiguration
`cvs add`	-	Hinzufügen einer Datei in das CVS-Archiv (erst bei `commit` wird die Datei in das CVS-Archiv übernommen)
`cvs remove`	-	Löschen einer Datei aus dem CVS-Archiv (ebenfalls erst nach `commit` wirksam)

CVS-Kommandos (2)

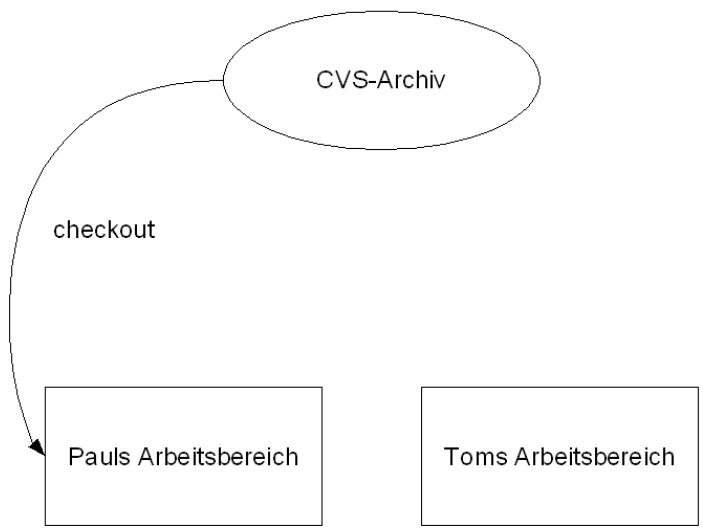
`cvs release`	-	Prüfung ob alle lokalen Änderungen in das CVS-Archiv übernommen wurden (wenn nicht, wird eine Warnung ausgegeben; mit einer speziellen Option kann man den Arbeitsbereich löschen)
`cvs update`	-	Arbeitsverzeichnis auf den neusten Stand bringen
`cvs edit`	-	Meldet eine Datei zum Bearbeiten an; gleichzeitig wird der Benutzer als Beobachter der Datei eingetragen

Auf den nächsten Folien folgen Beispiele an Hand eines Projektes „Taschenrechner“ für die Nutzung von CVS mit den Entwicklern Paul und Tom.

Beispiel 1: cvs checkout

Entwickler Paul kopiert das CVS-Archiv in seinen Arbeitsbereich:

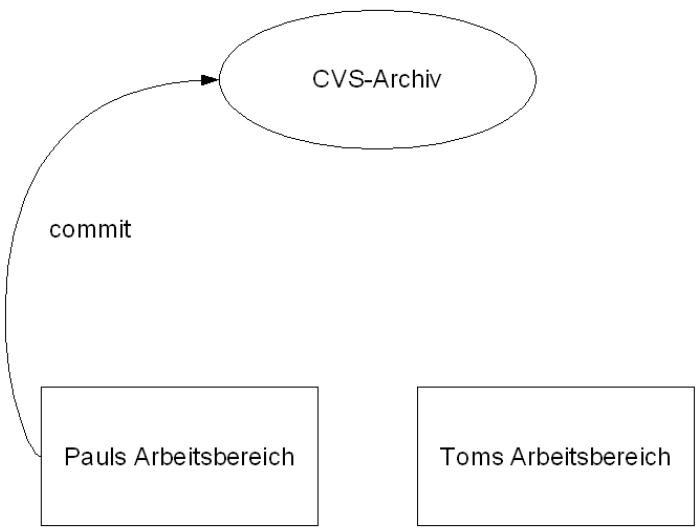
```
paul$ cvs checkout rechner
cvs checkout: Updating rechner
U rechner/Makefile
U rechner/calc.h
U rechner/add.cpp
U rechner/sub.cpp
paul$ _
```



Beispiel 2: cvs commit

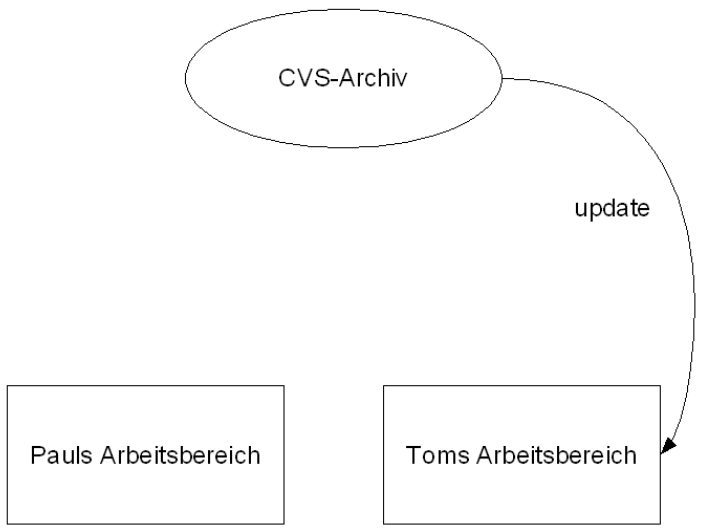
Entwickler Paul will eine Änderung in das CVS-Archiv kopieren.
Mit dem folgenden Befehl kopiert er die neue Revision:

```
paul$ cvs commit -m "Korrektur einer Berechnung" add.cpp
Checking in add.cpp;
/usr/share/CVS/rechner/add.cpp,v <-- add.cpp
new revision: 1.3; previous revision: 1.2
done
paul$ _
```



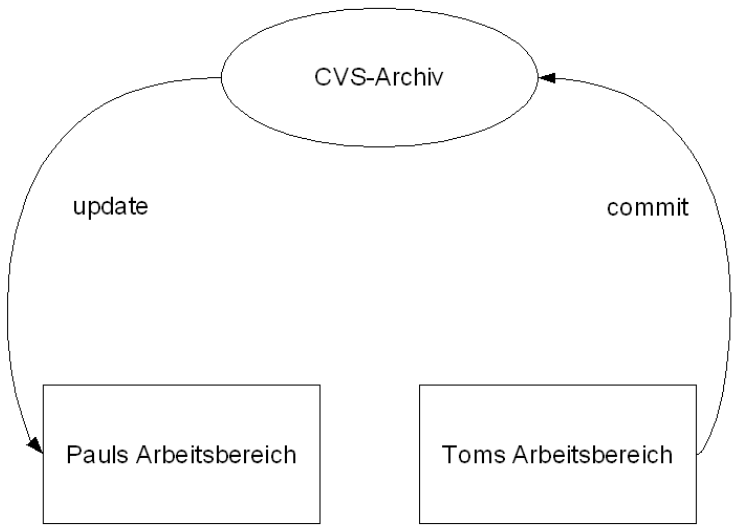
Beispiel 3: cvs update

Entwickler Tom macht nun `checkout`. Paul arbeitet an der Datei `sub.cpp` und Tom beginnt seine Arbeit an `calc.h`. Paul macht einen `commit`. Wenn Tom danach ebenfalls einen `commit` machen will, stellt CVS fest, dass die Datei `sub.cpp` in Toms Arbeitsbereich nicht mehr aktuell ist. Der `commit`-Befehl schlägt fehl. Um dem `commit`-Befehl gerecht zu werden, muss Tom seinen Arbeitsbereich auf den neusten Stand bringen. Dazu gibt er `cvs update` ein. Der `update`-Befehl bringt sein Arbeitsverzeichnis auf den neusten Stand.



Beispiel 3: cvs update (2)

Nach der Aktualisierung seines Arbeitsbereiches kann Tom mit `commit` seine Änderungen in das CVS-Archiv kopieren. Paul muss dann - genau wie Tom vorher - seinen Arbeitsbereich mit `update` aktualisieren, um wieder einen `commit` zu machen.



Beispiel 4: cvs tag

Das Produkt soll in der jetzigen Konfiguration an den Kunden ausgeliefert werden. Dazu kann Paul ihr einen Namen geben, damit sie später wiederhergestellt werden kann (z.B. zur Fehlersuche).

Paul wählt den Namen `rechner1` für die aktuelle Konfiguration:

```
paul$ cvs tag rechner1
cvs tag: Tagging.
T Makefile
T calc.h
T add.cpp
T sub.cpp
paul$ _
```

Wichtige Hinweise zum Umgang mit CVS

- Die Entwickler sollten sich absprechen und die Arbeit aufteilen, um das Risiko eines Konfliktes zu verhindern.
- Jeder Entwickler sollte regelmäßig seinen Arbeitsbereich mit dem CVS-Archiv abgleichen. Je länger kein Abgleich stattgefunden hat desto größer die Unterschiede zwischen den einzelnen Arbeitsbereichen und desto höher die Wahrscheinlichkeit eines Konfliktes.

Konflikte und Absprachen

Haben zwei Entwickler nicht nur in der gleichen Datei, sondern auch an der gleichen Stelle was geändert, kommt es zu einem Konflikt. CVS fügt dann beide Änderungen in die betreffende Datei ein und markiert die Konfliktstelle. Der Konflikt muss nun von Hand aufgelöst werden.

Konflikte und Absprachen

Haben zwei Entwickler nicht nur in der gleichen Datei, sondern auch an der gleichen Stelle was geändert, kommt es zu einem Konflikt. CVS fügt dann beide Änderungen in die betreffende Datei ein und markiert die Konfliktstelle. Der Konflikt muss nun von Hand aufgelöst werden.

In der Praxis treten solche Konflikte selten auf, wenn die Entwickler koordiniert vorgehen und ihre Arbeit untereinander absprechen.

Konflikte und Absprachen

Haben zwei Entwickler nicht nur in der gleichen Datei, sondern auch an der gleichen Stelle was geändert, kommt es zu einem Konflikt. CVS fügt dann beide Änderungen in die betreffende Datei ein und markiert die Konfliktstelle. Der Konflikt muss nun von Hand aufgelöst werden.

In der Praxis treten solche Konflikte selten auf, wenn die Entwickler koordiniert vorgehen und ihre Arbeit untereinander absprechen.

`edit` stellt eine Vorbeugemaßnahme von CVS dar. Durch den `edit`-Befehl meldet der Benutzer zum einen an, die angegebene Datei bearbeiten zu wollen, zum anderen wird er auf die Liste der Benutzer gesetzt, die diese Datei beobachten. Wendet man nun mit `commit` eine Änderung an, so wird man von CVS automatisch als Beobachter ausgetragen. Alle anderen Beobachter werden benachrichtigt, dass die Datei geändert wurde.

Zusammenfassung

CVS bietet gegenüber der bisherigen Arbeitsweise einige Möglichkeiten, die das Arbeiten an Projekten erleichtern (können). Hier noch einmal kurz die Vorzüge:

Zusammenfassung

CVS bietet gegenüber der bisherigen Arbeitsweise einige Möglichkeiten, die das Arbeiten an Projekten erleichtern (können). Hier noch einmal kurz die Vorzüge:

- Änderungen von mehreren Entwicklern an einer Datei gehen nicht verloren, d.h. mehrere Entwickler können parallel an einem Projekt arbeiten

Zusammenfassung

CVS bietet gegenüber der bisherigen Arbeitsweise einige Möglichkeiten, die das Arbeiten an Projekten erleichtern (können). Hier noch einmal kurz die Vorzüge:

- Änderungen von mehreren Entwicklern an einer Datei gehen nicht verloren, d.h. mehrere Entwickler können parallel an einem Projekt arbeiten
- es lässt sich ohne viel Mühe jede Änderung rückgängig machen

Zusammenfassung

CVS bietet gegenüber der bisherigen Arbeitsweise einige Möglichkeiten, die das Arbeiten an Projekten erleichtern (können). Hier noch einmal kurz die Vorzüge:

- Änderungen von mehreren Entwicklern an einer Datei gehen nicht verloren, d.h. mehrere Entwickler können parallel an einem Projekt arbeiten
- es lässt sich ohne viel Mühe jede Änderung rückgängig machen
- man kann Konfigurationen unter einem Namen ablegen; dies ist z.B. hilfreich, wenn eine bestimmte Konfiguration an einen Kunden ausgeliefert wird und nachher für eventuelle Fehlersuche wiederhergestellt werden soll

Zusammenfassung (2)

Zusammenfassung (2)

- ein CVS-System ermöglicht die Versionierung von kompletten Dateibäumen, d.h. CVS berücksichtigt auch das Anlegen und Löschen von Dateien

Zusammenfassung (2)

- ein CVS-System ermöglicht die Versionierung von kompletten Dateibäumen, d.h. CVS berücksichtigt auch das Anlegen und Löschen von Dateien
- ein Entwickler kann eine oder mehrere Dateien beobachten und wird informiert, sobald Änderungen in das CVS-Archiv eingeflossen sind

Zusammenfassung (2)

- ein CVS-System ermöglicht die Versionierung von kompletten Dateibäumen, d.h. CVS berücksichtigt auch das Anlegen und Löschen von Dateien
- ein Entwickler kann eine oder mehrere Dateien beobachten und wird informiert, sobald Änderungen in das CVS-Archiv eingeflossen sind
- man kann dem CVS-System auch mitteilen, dass man gerne eine Datei bearbeiten möchte; dabei werden alle anderen Entwickler, die diese Datei beobachten, informiert

CVS im Internet

- http://www.nongnu.org/cvs/
- http://de.wikipedia.org/wiki/CVS
- http://en.wikipedia.org/wiki/Concurrent_Versions_System