

CVS

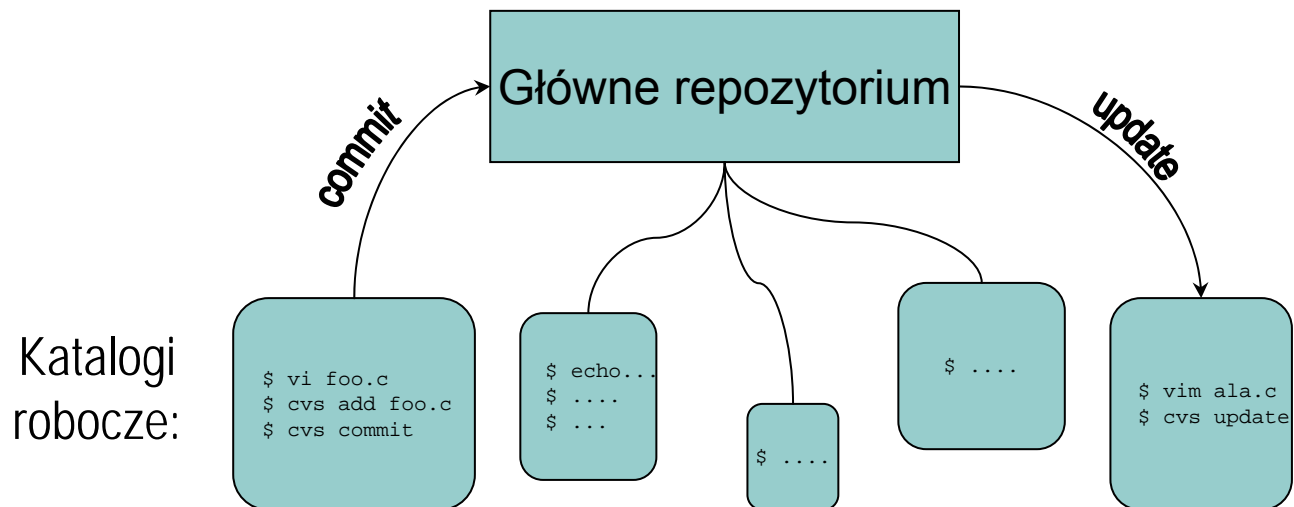
Concurrent Versions System

Prezentacja ZPP

Grupa Sumery

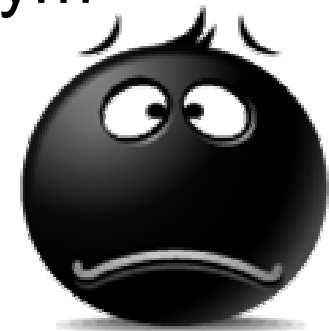
Czym jest CVS?

- System kontroli wersji, umożliwia:
 - Zachowywanie historii zmian w plikach
 - Jedoczesną pracę nad projektem kilku osobom
- Ogólna idea:



Do czego CVS się nie nadaje?

- nie jest systemem budowy oprogramowania, nie ma nic wspólnego z make i pochodnymi;
- nie służy do zarządzania projektem;
- nie jest przeznaczony do komunikacji między członkami zespołu;
- nie jest rozbudowanym i skomplikowanym systemem, czyli:
 - to nie jest Bugzilla
 - nie wspomaga automatycznego testowania;



Rys historyczny

- Lipiec 1986 r., Dick Grune –
comp.source.unix – pierwsze skrypty
shellowe;
- kwiecień 1989 - Brian Berliner - pierwsza
wersja;
- najnowsze wersje stabilna (2004 10 21)
 - Stabilna - 1.11.17;
 - Rozwojowa - 1.12.9;

Obsługa CVS

Pierwszy import do repozytorium

import

1. Tworzymy lokalną strukturę katalogów

```
$ mkdir -p alama/{asa,kota}
```

2. Wysyłamy ją do repozytorium

```
$ cd alama  
$ cvs import -d /usr/local/cvsroot -m "początkowa struktura  
katalogow" alama alasoftware start
```

```
cvs import: Importing /usr/local/cvsroot/alama/alama  
cvs import: Importing /usr/local/cvsroot/alama/alama/asa  
cvs import: Importing /usr/local/cvsroot/alama/alama/kota
```

```
No conflicts created by this import
```

alama – katalog
alasoftware – vendor name
start – release tag

Adres repozytorium

- Określenie adresu repozytorium:
 - Opcja `-d 'directory'`
`& cvs -d /usr/local/cvsroot checkout alama`
 - Zmienna `CVSROOT`, `CVS_RSH`
`$ export CVS_RSH=ssh`
`$ export CVSROOT=fuksa@cvs.alama.sf.net:/cvsroot/alama`
 - Pliki administracyjne w katalogu CVS

Obsługa CVS

Pobranie plików z repozytorium checkout

- tworzy lokalną kopię repozytorium - katalog roboczy w którym działamy;
- konieczne przed dalszym używaniem cvs;

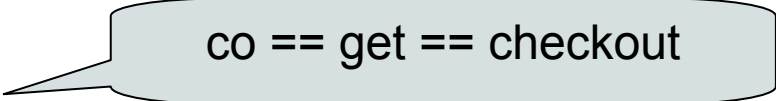
```
$ cvs co alama
```

```
cvs checkout: Updating alama
```

```
cvs checkout: Updating alama/alama
```

```
cvs checkout: Updating alama/alama/asa
```

```
cvs checkout: Updating alama/alama/kota
```



co == get == checkout

Dodawanie nowych plików

add

- tworzymy nowy plik

```
$ cat << KONIEC > kicikici.c  
> int main(){  
> puts("miau");> }  
> KONIEC
```

- dodajemy do lokalnego repozytorium

```
$ cvs add kicikici.c  
cvs add: scheduling file `kicikici.c' for addition  
cvs add: use 'cvs commit' to add this file permanently
```

- wysyłamy zmiany do głównego repozytorium

```
$ cvs commit  
...
```

Obsługa CVS

Wysyłamy nasze zmiany na serwer commit

- Zmiany w plikach są trzymane lokalnie, commit dopiero synchronizuje repozytorium z naszym katalogiem

```
$ cvs commit -m „zmieniamy odbiorcow na milosnikow psow.”  
cvs commit: Examining .  
cvs commit: Examining asa  
cvs commit: Examining kota  
Checking in kicikici.c;  
/usr/local/cvsroot/alama/alama/kicikici.c,v <-- kicikici.c  
new revision: 1.3; previous revision: 1.2  
done
```



Dokumentujemy zmiany - logi

- Przełącznik `-m „log”` lub zewnętrzny edytor (zmienne: `CVSEEDITOR`, `VISUAL`, `EDITOR`)

```
pierwsza wersja naszego killer appa.
```

```
CVS: -----  
CVS: Enter Log. Lines beginning with `CVS:' are removed automatically  
CVS:  
CVS: Committing in .  
CVS:  
CVS: Modified Files:  
CVS:   kicikici.c  
CVS: -----  
~  
~
```

```
1,0-1   Wszystko
```

commit – co może pójść źle

- commit na nieaktualnym katalogu roboczym

```
$ cvs commit
cvs commit: Examining .
cvs commit: Up-to-date check failed for `kicikici.c`
cvs commit: Examining asa
cvs commit: Examining kota
cvs [commit aborted]: correct above errors first!
```



Dlatego zawsze należy zrobić **update zanim robi się **commit!!!****

```
$ cvs update
cvs update: Updating .
U kicikici.c
cvs update: Updating asa
cvs update: Updating kota
```

Obsługa CVS

Synchronizacja z głównym repozytorium update

- zmieniamy kicikici.c:

```
#include <stdio.h>
int main(){
puts("hau");
}
```

- Próbujemy zsynchronizować katalog roboczy z
głównym repozytorium

```
$ cvs update
cvs update: Updating .
M kicikici.c
cvs update: Updating asa
cvs update: Updating kota
```

'M' – 'Modified' – ten plik jest
zmieniony (nowszy) niż
odpowiadający plik w repozytorium

Obsługa CVS

Synchronizacja z głównym repozytorium cd update z konfliktem

- Czasami może wystąpić konflikt pomiędzy naszymi plikami a plikami w repozytorium, np:

Plik kicikici.c lokalnie:

```
int main(){  
puts("muuuu!");  
}
```

Plik kicikici.c w repozytorium:

```
int main(){  
puts(„buhibhui!");  
}
```

```
$ cvs update  
cvs update: Updating .  
RCS file: /usr/local/cvsroot/alama/alama/kicikici.c,v  
retrieving revision 1.3  
retrieving revision 1.4  
Merging differences between 1.3 and 1.4 into kicikici.c  
rcsmerge: warning: conflicts during merge  
cvs update: conflicts found in kicikici.c  
C kicikici.c  
cvs update: Updating asa  
cvs update: Updating kota
```

Obsługa CVS

Synchronizacja z głównym repozytorium cd update z konfliktem cd

- Po takim nieudanym **update** otrzymamy:

```
int main(){
<<<<<< kicikici.c
puts("buhibuhi!");
=====
puts("muuuu!");
>>>>>> 1.4
}
```

- Status pliku również jednoznacznie wskazuje na konflikt

```
$ cvs status kicikici.c
```

```
=====
File: kicikici.c      Status: File had conflicts on merge
Working revision:    1.6      Result of merge
Repository revision: 1.6      /usr/local/cvsroot/alama/alama/kicikici.c,v
...
```

- W takim pliku musimy ręcznie ustalić, co ma się w nim znaleźć, a następnie wysłać zmiany do repozytorium

Obsługa CVS

Różnice pomiędzy kat. roboczym a repozytorium diff

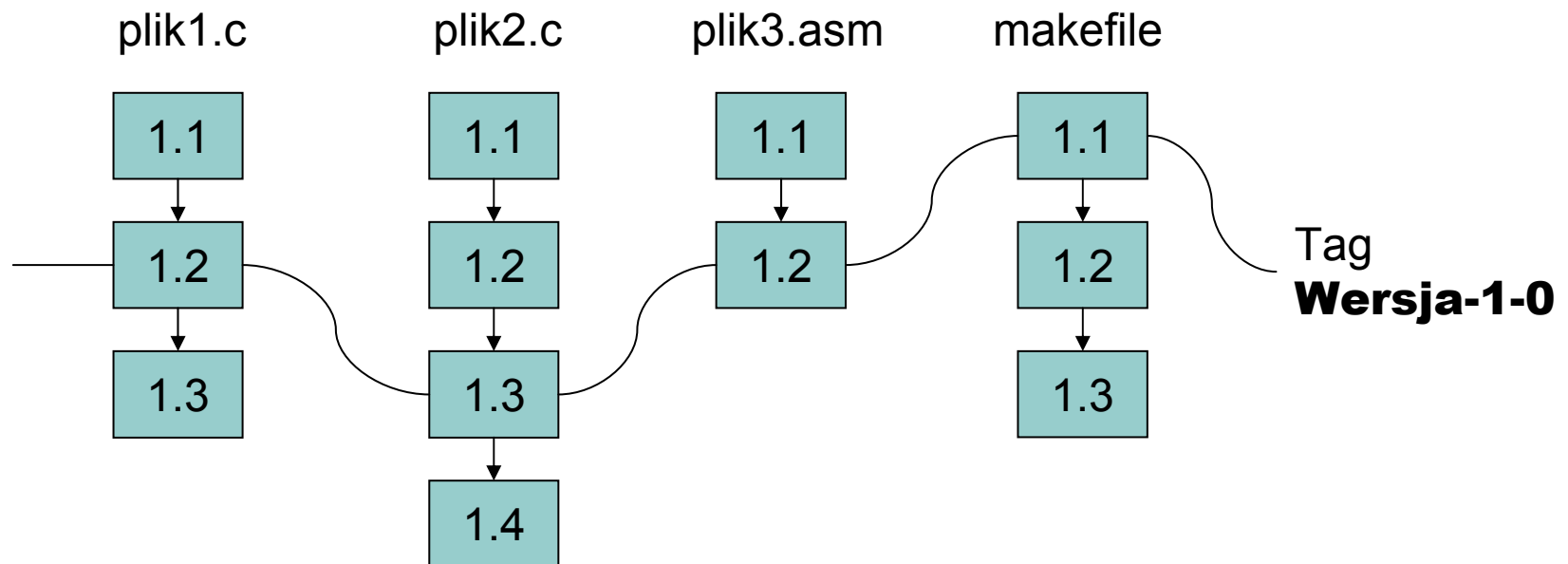
- Posiadamy plik, i chcemy zobaczyć, jakie są różnice pomiędzy nim a plikiem w repozytorium

```
$ cvs diff
cvs diff: Diffing .Index: kicikici.c
=====
====
RCS file: /usr/local/cvsroot/alama/alama/kicikici.c,v
retrieving revision 1.5
diff -r1.5 kicikici.c
3c3
< puts("hau");
---
> puts("mrrau!");
cvs diff: Diffing asa
cvs diff: Diffing kota
```

Tagi – a co to?

tag

- Tagami można połączyć wiele różnych plików w jeden obiekt logiczny (np. Wersja-1-0)



- Poprzez tagi można potem przeglądać repozytorium, np.
`$cvs co -r wersja-2-0 alama`

Tag – ciąg dalszy

- Dodawanie tagu do plików (katalogów)

```
$ cvs tag wersja-2-0 .  
cvs Tag: tagging .  
T kicikici.c  
cvs Tag: tagging asa  
cvs Tag: tagging kota
```

- Sprawdzenie statusu pliku

```
$ cvs status -v kicikici.c  
=====
```

File: kicikici.c	Status: Up-to-date	
Working revision:	1.8	Mon Oct 19 16:10:59 2004
Repository revision:	1.8	/usr/local/cvsroot/alama/alama/kicikici.c,v
Sticky Tag:	(none)	
Sticky Date:	(none)	
Sticky Options:	(none)	

```
Existing Tags:  
wersja-2-0 (revision: 1.8)
```

Obsługa CVS

Sprawdzenie historii

log

- Czasami chcemy dowiedzieć się, jak plik ewoluował

```
$ cvs log kicikici.c
RCS file: /usr/local/cvsroot/alama/alama/kicikici.c,v
Working file: kicikici.c
head: 1.3
branch:
locks: strict
access list:
symbolic names:
    wersja-2-0: 1.2
keyword substitution: kv
total revisions: 2;      selected revisions: 2
description:
    -----
revision 1.2
date: 2004/10/18 16:06:26;  author: af209240;  state: Exp;  lines: +2 -1
zmieniamy odbiorcow na milosnikow psow.
-----
revision 1.1
date: 2004/10/18 15:47:52;  author: af209240;  state: Exp;
pierwsza wersja naszego killerapa
=====
```

Dane o plikach w plikach

„keyword substitution”

- Czasami chcemy , aby pliki lokalnie mogły podać dane np. o swojej historii (bez łączenia się z serwerem CVS)
- Słowa kluczowe zostaną przy update i co zastąpione odpowiednimi rozwinięciami

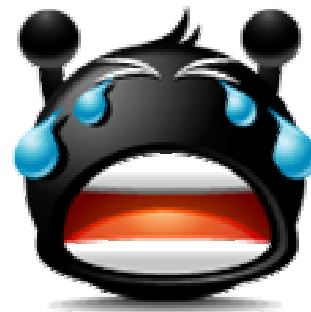
```
/* $Id$  
*/  
int main(){  
puts("muuuu!");  
}
```

```
/* $Id: kicikici.c,v 1.8 2004/10/19 21:52:38 af209240 Ext $  
*/  
int main(){  
puts("muuuu!");  
}
```

Pliki binarne a CVS

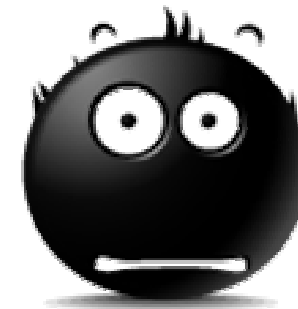
- Niestety CVS NIE był tworzony dla plików binarnych
- Najlepsze co można zrobić, to je przechowywać w repozytorium (przełącznik `-kb`), ale sami musimy zadbać o badanie różnic, mergowanie itd.

```
$ cvs add -kb -m „test do binarki” mruczek
```



Jednak nie taki fajny...

- CVS posiada też wiele wad, m.in.:
 - operacje zmian nazw plików i katalogów są trudne i niebezpieczne (przepisywanie historii zmian przez pliki administracyjne) należy przyjąć, iż **NIE DA SIĘ ICH WYKONAĆ!** (Pewnym rozwiązaniem jest `cvs remove`; `cvs add`;))
 - brakuje wersjonowania katalogów
 - brak atomowego commit (!)
 - słaba obsługa plików binarnych



Dodatkowe informacje

CVS w sieci

- www.cvshome.org - oficjalna strona, bardzo dobra dokumentacja;
- gdzie założyć darmowe repozytorium:
 - www.sourceforge.net
 - berlios.de
 - savannah.gnu.org
 - I najlepsze: własny serwer



Dodatkowe informacje

Ponad CVS

- Istnieje wiele alternatyw dla CVS, aczkolwiek nie wszystkie są darmowe. Warto wymienić:
 - subversion
 - bitkeeper
 - arch

THE END

Credits:

Zespół SUMERY



Special Thanks:

Rokey of Netease company for his icons.

www.rokey.net