AFS

Andrew File System

rozproszony system plików

Założenie projektowe

standardowe

- przezroczystość dostępu
- przezroczystość położenia
- przezroczystość współbieżności
- przezroczystość awarii
- przezroczystość wydajności
- przezroczystość sprzętu i systemu operacyjnego
- bezpieczeństwo
- *skalowalność*

Obserwacje

dla środowisk akademickich

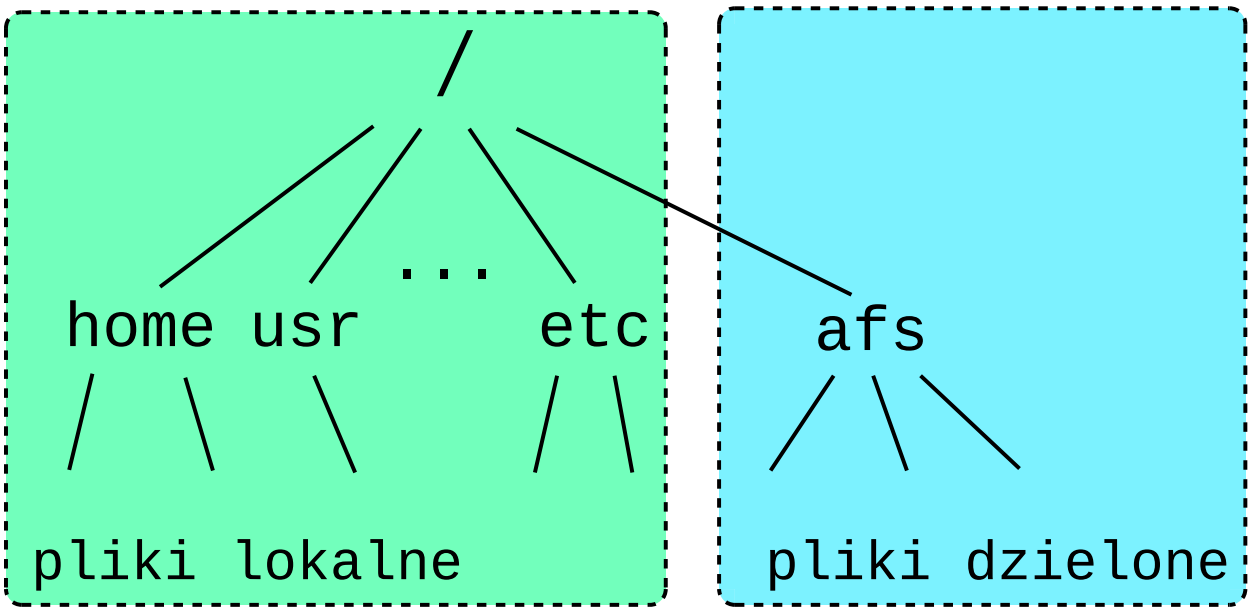
- większość plików jest mała
- czytanie jest częstsze niż pisanie
- współdzielenie plików jest rzadkie
- dostęp do plików jest sekwencyjny
- odwołania do plików są skumulowane

Założenie projektowe

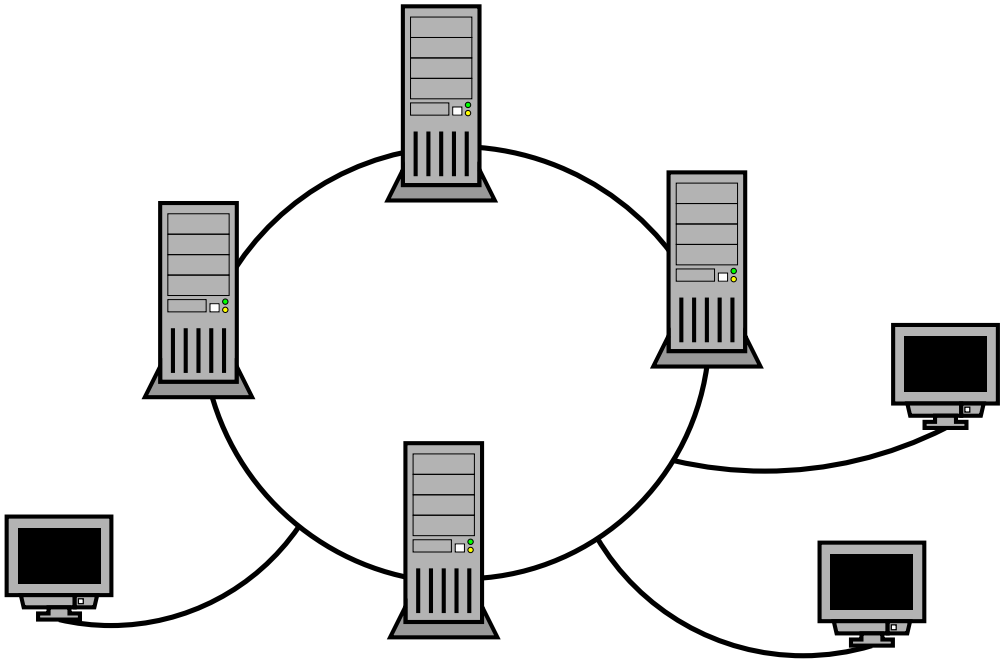
dodatkowe

- przezroczystość zwielokrotnienia
- przezroczystość wędrówki
- architektura klient–serwer
- semantyka sesji
- semantyka jednej kopii
- *usługi całoplikowe*
- system niebazodanowy

Jednolita przestrzeń nazw



Komórka (cell)



- serwery
- maszyny klienckie
- konta użytkowników

- komputery współdzielą ustawienia konfiguracyjne
- komórki można łączyć

Klient (Venus)

- daje użytkownikom dostęp do plików dzielonych
- przechowuje kopie plików zdalnych lokalnie w *cache'u*
- z każdym plikiem wiąże wskaźnik ważności

Serwer (Vice)

- przechowuje pliki dzielone
- udostępnia pliki dzielone klientom
- daje klientom obietnice zawiadomienia (*callback promise*)

Typowy scenariusz

1. Użytkownik chce otworzyć plik do zapisu
2. *Venus* nie znajduje w *cache'u* ważnej kopii pliku
3. *Venus* wyszukuje serwer przechowujący plik
4. *Venus* ściąga plik z tego serwera
5. *Vice* daje *Venus* obietnicę zawiadomienia
6. *Venus* zapisuje plik w *cache'u*
7. Użytkownik pracuje na pliku lokalnie
8. Użytkownik zamyka plik
9. *Venus* odsyła zmieniony plik *Vice*
10. Plik NIE JEST usuwany z *cache'u*
11. *Vice* zapisuje zmieniony plik
12. *Vice* rozsyła zawiadomienia do innych klientów
13. Inni klienci unieważniają lokalną kopię pliku

Konsekwencje

- *cache* radykalnie zmniejszenia ilość przesyłanych danych
- komunikacja klient–serwer tylko w czasie otwierania i zamykania
- brak kontroli współbieżnych aktualizacji

Callback:

- nie trzeba sprawdzać wersji pliku podczas otwierania
- lepsza niż znaczniki czasu skalowalność
- utrata bezstanowości serwerów

Semantyka aktualizacji

otwarcie pliku

plik jest aktualny

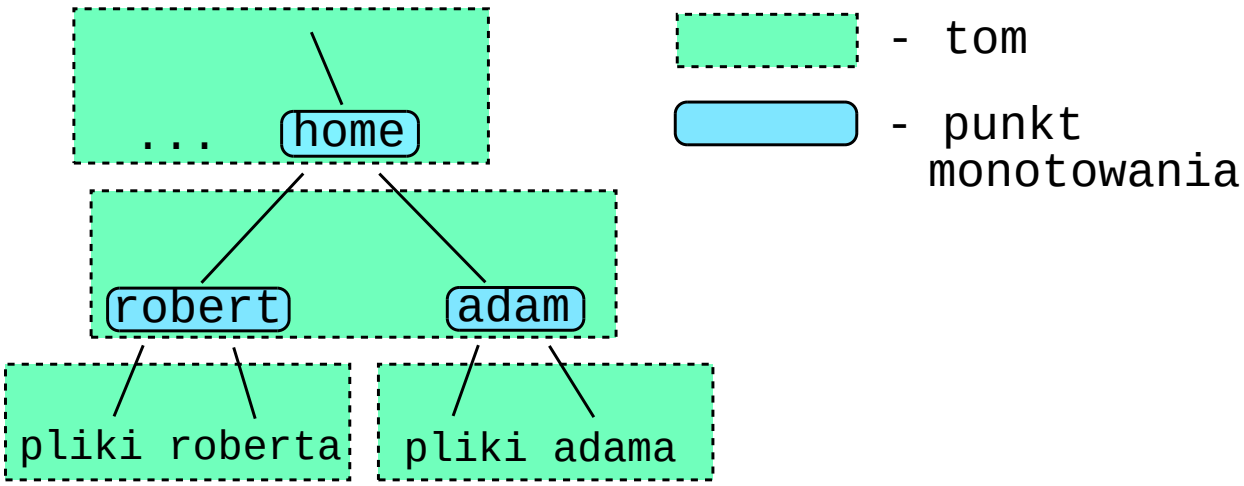
lub

- zawiadomienie zostało zagubione w ciągu ostatnich T sekund
- plik znajduje się w *cache'u*
- plik jest przestarzały o nie więcej niż T sekund

T – czas po jakim obietnice powiadomień są odświeżane

Tomy (Volumes)

- zbiór katalogów, plików i *punktów montowania*
- poddrzewo struktury plików
- zawartość powinna tworzyć logiczną całość
- każdy plik należy do dokładnie jednego tomu
- rozmiar tomu jest ograniczony (*quota*)



- serwer przechowuje bazę położeń tomów, nie plików
- tomy mogą być przenoszone między serwerami
- tomy mogą być *replikowane*

Replikowanie tomów

- tom posiada oryginał – jedyną wersję zapisywalną
- repliki tomu to kopie tylko do odczytu

Zalety replikacji:

- zwiększa dostępność tomu
- pomaga równomiernie rozkładać obciążenie serwerów
- zwiększa odporność na awarie

Wady replikacji:

- zmiana oryginału wymusza zmianę kopii

Access Control Lists

- związane z katalogiem
- dotyczą katalogu i plików w nim zawartych
- nadawane względem grupy/użytkownika/maszyny
- dla jednego katalogu 20 „slotów”

nadawane prawa

	dotyczące katalogu	
l	*lookup*	pozwala obejrzeć zawartość katalogu
i	*insert*	wstawianie pliku do katalogu
d	*delete*	usuwanie pliku z katalogu
a	*administer*	zmiana ACL
	dotyczące zawartych w nim plików	
r	*read*	czytanie
w	*write*	pisanie
k	*lock*	zakładanie blokady

można nadawać prawa negatywne

Podsumowanie

Zalety AFS

- dobra wydajność
- sensowna semantyka
- dobra skalowalność

Coda

rozproszony system plików

Coda powstała na bazie AFS.

Projektanci mieli na celu wyeliminować wady AFS:

- małą odporność na awarie
- brak replikacji tomów zapisywalnych
- brak wspomagania komputerów przenośnych

Operacja odłączenia

Przyczyny:

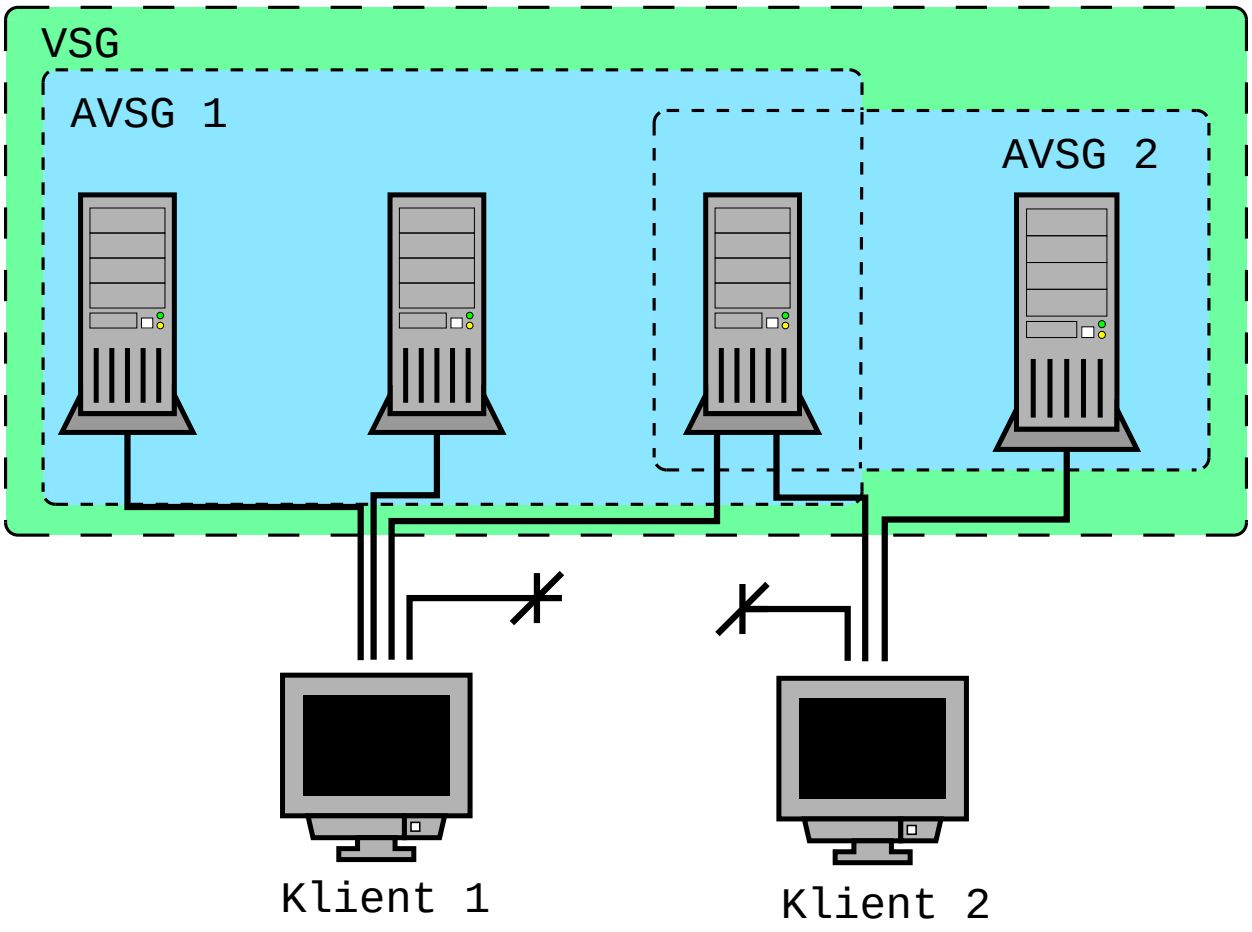
- awarie serwerów lub sieci
- odłączanie komputera przenośnego od sieci

Problemy:

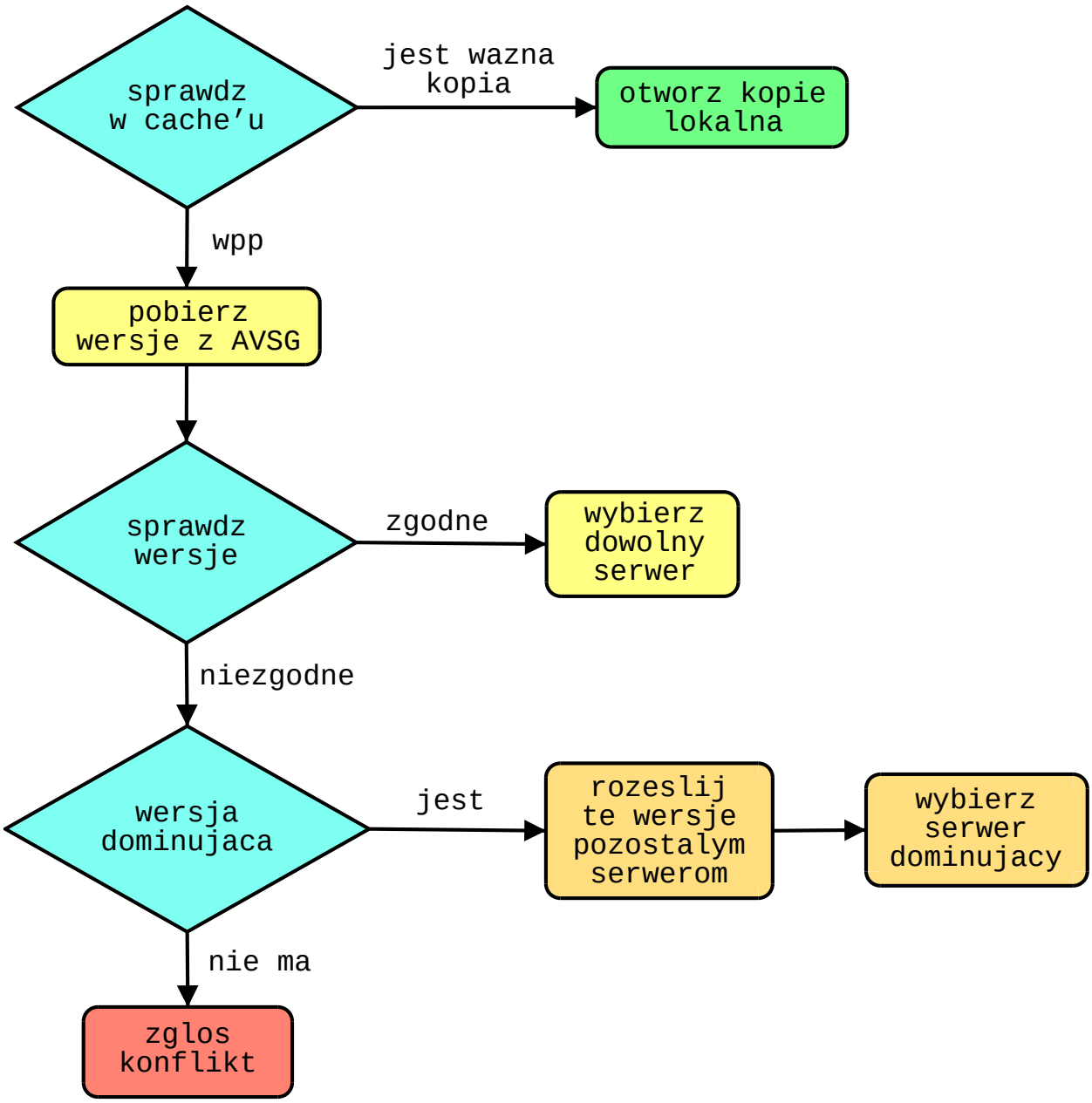
- zagwarantowanie obecność potrzebnych plików – *hoarding*
- reintegracja po ponownym połączeniu

VSG i AVSG

- pojęcia dotyczące tomu
- *Volume Storage Group* – globalne
- *Available Volume Storage Group* – lokalne



Otwieranie


sprawdz
w cache'u
jest wazna
kopia
otworz kopie
lokalna
wpp
pobierz
wersje z AVSG
sprawdz
wersje
zgodne
wybierz
dowolny
serwer
niezgodne
wersja
dominujaca
jest
rozeslij
te wersje
pozostalym
serwerom
wybierz
serwer
dominujacy
nie ma
zglos
konflikt

Otwieranie c.d.

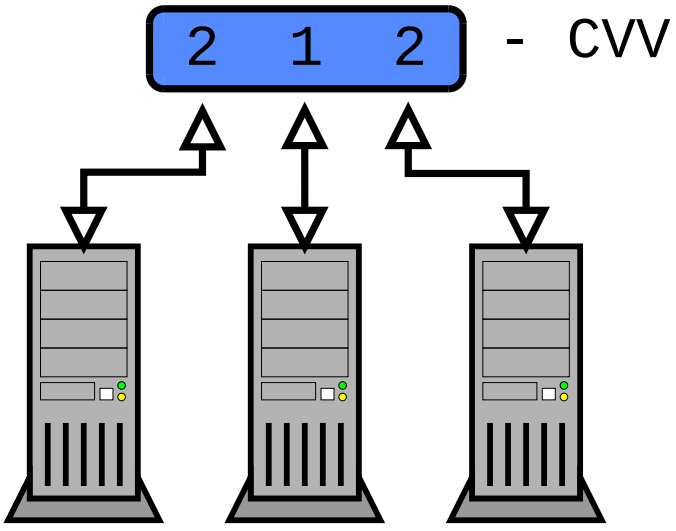
- *Venus* ściąga z wybranego serwera plik.
- tylko wybrany serwer daje obietnicę powiadomienia.

Zamykanie

- *Venus* odsyła plik do wszystkich serwerów z AVSG
- *Venus* aktualizuje wersje plików na tych serwerach
- *Venus* wykrywa konflikty i zgłasza je użytkownikowi

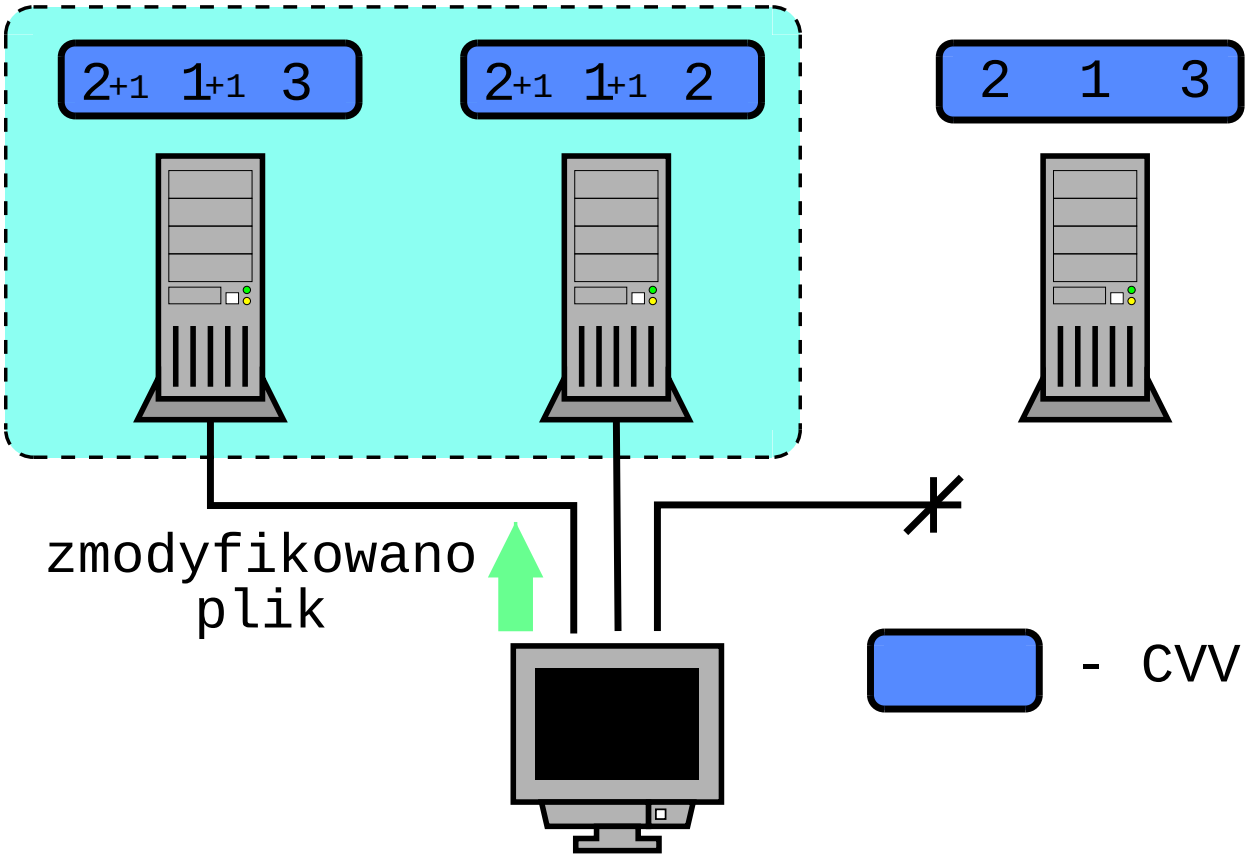
Coda Version Vectors

- jeden CVV dla każdego pliku
- wektor długości równej wielkość VSG pliku
- elementy wektora to liczby naturalne
- jeden element odpowiada jednemu serwerowi
- element szacuje liczbę modyfikacji pliku na danym serwerze



Coda Version Vectors

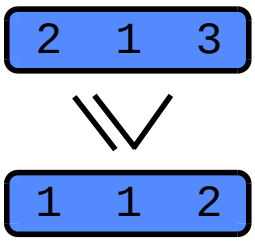
- CVV jest aktualizowany podczas zamykania pliku
- CVV może być różny na różnych serwerach



Coda Version Vectors

dominacja

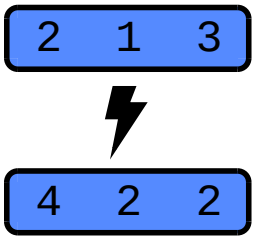
CVV1 dominuje nad CVV2 jeżeli każdy element CVV1 jest nie mniejszy niż odpowiedni element CVV2



możliwe jest automatyczne uspójnienie wersji na serwerach

konflikt

w przeciwnym wypadku CVV1 jest w konflikcie z CVV2



konieczna jest ręczna ingerencja

Semantyka aktualizacji

udane otwarcie pliku

- *AVSG* jest niepuste
- plik jest aktualny

lub

- *AVSG* jest niepuste
- plik jest przestarzały o co najwyżej T sekund
- plik jest w *cache'u*
- przez ostatnie T sekund stracono zawiadomienia

lub

- *AVSG* jest puste
- plik jest w *cache'u*

Semantyka aktualizacji

nieudane otwarcie pliku

- *AVSG* jest niepuste
- są konflikty

lub

- *AVSG* jest puste
- pliku nie ma w *cache'u*

Semantyka aktualizacji

udane zamknięcie pliku

- *AVSG* jest niepuste
- plik pomyślnie zaktualizowano

lub

- *AVSG* jest puste

nieudane zamknięcie pliku

- *AVSG* jest niepuste
- wykryto konflikty

Podsumowanie

- dobra odporność na awarie
- wspomaganie komputerów przenośnych
- dobra wydajność (trochę gorsza niż AFS)
- sensowna semantyka
- dobra skalowalność
- możliwość pojawienia się konfliktów