

# To Mercurial SCM

Γιώργος Κεραμίδας  
keramida@FreeBSD.org

FreeBSD Project

30 Νοέμβρη 2007  
Τμήμα Πληροφορικής  
Πανεπιστήμιο Πειραιά

# Θέμα της ομιλίας

- Τι είναι το Mercurial
- Συνεργασία μεταξύ μιας ομάδας
- Πώς να δουλέψετε αποδοτικά
- Γιατί είναι γρήγορο το Mercurial
- Γιατί είναι σημαντικά τα εργαλεία μας

# Το Κοινό της Ομιλίας

- Θέλετε να διαχειριστείτε κάποια αρχεία
  - ▶ Πηγαίος κώδικας
  - ▶ Web site
  - ▶ Το επόμενο βιβλίο σας
- Δε σας φοβίζει το command prompt

# Τι είναι το Mercurial

- Ξεκίνησε τον Απρίλη του 2005
- Γραμμένο κυρίως σε Python
  - ▶ 20.000 γραμμές σε Python, 1.000 γραμμές σε C
  - ▶ 93% Python, 7% C
- Πολύ δημοφιλές, κυρίως λόγω ταχύτητας και ευκολίας χρήσης

# Source Configuration Management

- Wikipedia
  - ▶ 22 free SCM συστήματα
  - ▶ 20 proprietary SCM συστήματα

# Τι Θέλει Ένας Developer

- Εχω δουλειά να κάνω!
- Θέλω κάτι απλό, που δουλεύει σωστά
- Τα SCM εργαλεία μου πρέπει:
  - ▶ Να είναι εύκολα στη χρήση
  - ▶ Να με βοηθούν στη συνεργασία μου με άλλους
  - ▶ Να είναι γρήγορα

# Ευκολία Χρήσης

# Ευκολία Χρήσης του Mercurial

- Υπάρχουν μόνο 3 πράγματα σε ένα “repository”
  - ▶ Repository
  - ▶ Changesets
  - ▶ Περιοχή εργασίας

# Τι είναι ένα Repository;

- Απλότητα
  - ▶ Απλά ένα directory με το history του project
  - ▶ Ούτε database, ούτε απαιτήσεις για server/client.
- Ταχύτητα
  - ▶ Το να γίνει “clone” ένας κατάλογος είναι εύκολο και γρήγορο
- Περισσότερη απλότητα
  - ▶ Όλη η δουλειά γίνεται μέσα σε μια περιοχή εργασίας
  - ▶ Κάθε repository “ανήκει” στον developer που το έφτιαξε
  - ▶ Τα προσωπικά repositories είναι εύκολο και να γίνουν “clone”, αλλά και να καθαριστούν.

# Τι υπάρχει μέσα σε ένα repository;

- Ένα “changelog”
  - ▶ Το πλήρες ιστορικό όλων των αλλαγών
- Ένα “manifest”
  - ▶ Αντιστοιχία changesets και αρχείων
- Δεδομένα για κάθε αρχείο
  - ▶ Το ιστορικό κάθε αρχείου στο repository

# Σύγκριση παραδοσιακών μοντέλων repository

	<b>Παραδοσιακό SCM</b>	<b>Mercurial</b>
Central repo	Μόνο ένα	Όσα χρειάζεται
Bottleneck	Server	Κανένα
Load mgmt	Δύσκολο ή αδύνατο	Mirrors οπουδήποτε, μικρό κόστος
Απομακρυσμένοι χρήστες	Αργή απόκριση	Γρήγορη local απόκριση
Αποτυχία server	Καταστροφική	Πλήρες backup σε κάθε clone
Δικτ. Σύνδεση	Χρειάζεται πάντα	Έντελώς προαιρετική

# Τι είναι ένα changeset;

- Ένα snapshot από το project μια δεδομένη στιγμή
- Καταγράφει:
  - ▶ Ποιός έκανε την αλλαγή
  - ▶ Πότε έγινε η αλλαγή
  - ▶ Γιατί έγινε η αλλαγή (περιγραφή)
  - ▶ Ποιά αρχεία άλλαξαν και πως
  - ▶ Ποιό ήταν το “parent changeset”
- Η δημιουργία ενός changeset είναι ένα “commit”

# Τι είναι η περιοχή εργασίας

- Ένα αντίγραφο του repository από ένα changeset
  - ▶ Αυτό το changeset είναι το *parent* της περιοχής εργασίας
- Μπορούμε να αλλάξουμε οποιοδήποτε αρχείο στην περιοχή εργασίας
  - ▶ Οι αλλαγές θα αποτελούν μέρος του επόμενου changeset
  - ▶ Μπορούμε να προσθέσουμε, να αφαιρέσουμε, να μετονομάσουμε και να αντιγράψουμε αρχεία στην περιοχή εργασίας
- Μπορούμε να δούμε τι αλλάξαμε και πως

# Tutorial-αστραπή: Το Hg σε ένα λεπτό

Δημιουργία repository	<i>hg init newrepo</i>
Είσοδος στο repository	<i>cd newrepo</i>
Δημιουργία αρχείου	<i>emacs file.c</i>
Προσθήκη του αρχείου	<i>hg add file.c</i>
Τι έγινε τώρα; Νέο αρχείο	<i>hg status</i> <i>A file.c</i>
Αποθήκευση αλλαγών	<i>hg commit</i>

# Παράλληλη εργασία

- Οι developers γενικά δουλεύουν *παράλληλα*
  - ▶ Τα περισσότερα revision control συστήματα κάνουν τα πράγματα δύσκολα
- Κάνω κάποιες αλλαγές
- Πάω να τις κάνω commit
- Κι αν κάποιος άλλος έχει κάνει commit ήδη;

# Προβλήματα την ώρα του commit

- Αν κάποιος άλλος έχει κάνει commit πριν από μένα;
- Πρέπει να κάνω *merge* τις αλλαγές του *πριν* κάνω commit τις δικές μου
- **Δεν υπάρχει αποθηκευμένο αντίγραφο από τις δικές μου αλλαγές ακόμα.**
- Ένα λάθος κατά τη διάρκεια του merge μπορεί να **χάσει** τις αλλαγές μου

# Το Μοντέλο του Hg: Branching

- Θυμάστε ότι ένα changeset έχει ένα parent;
- Δύο changesets με το ίδιο parent είναι ένα *branch*
- Κι αυτό είναι όλο.
  - ▶ Τίποτα πιο μπερδεμένο ή περίεργο

# Το Μοντέλο του Hg: Merging

- Τι κάνουμε με τα branches;
- Κάποια changesets έχουν δύο parents
- Αυτά λέγονται *merge changesets*
- Ένα merge changeset λέει απλά *έτσι έκανα merge τα changesets A και B*

# Ευκολία στη Συνεργασία με Άλλους

- Το Mercurial υποστηρίζει εύκολα τη συνεργασία με τους άλλους
- Κάνω commit τις δικές μου αλλαγές όποτε βολεύει εμένα
- Οι αλλαγές μου είναι καθαρές και αυτόνομες
- Δεν κάνω merge με τις δικές σου αλλαγές παρά μόνο *αφού* έχω κάνει commit τις δικές μου αλλαγές με ασφάλεια

# Merging Χωρίς Άγχος

- Κι αν κάνω λάθος κατά τη διάρκεια ενός merge;
- Οι αλλαγές μου είναι ήδη committed. Και οι δικές σου
- **Δεν χάνεται υπάρχουσα δουλειά για κανένα λόγο**
- Απλά ξανακάνω το merge από την αρχή

# Συνεργασία και Changesets

- Ενσωματωμένος web server
  - ▶ CGI-server για integration με το Apache
- SSH tunneling για remote access
- Δουλεύει σωστά με networked file systems (NFS, Samba)
- Ανταλλαγή changesets με offline τρόπους (USB flash, email, ...)

# Η Συνεργασία είναι Συμμετρική

- Δημιουργώ ένα τοπικό *clone*
- Κάνω *pull* κάποιες αλλαγές
- Κάνω *push* τις δικές μου
- Μετά το *push*, το *remote repository* είναι ακριβές αντίγραφο από το τοπικό *repository*

# Tutorial-αστραπή για Ανταλλαγή Changesets

<i>hg commit</i>	Τέλειωσα με τις αλλαγές μου
<i>hg push</i>	Στέλνω τις αλλαγές μου
<i>abort: unsynced remote changes!</i>	Πρέπει να κάνω πρώτα merge
<i>hg update</i>	Ανανεώνω την περιοχή εργασίας
<i>hg pull</i>	Τραβάω τις remote-only αλλαγές
<i>added 1 changeset with 1 change to 1 file</i>	
<i>(run 'hg heads' to see heads, 'hg merge' to merge)</i>	
<i>hg merge</i>	Κάνω merge με τις δικές μου
<i>hg commit</i>	Αποθηκεύω το merge changeset
<i>hg push</i>	Στέλνω τις αλλαγές μου
<i>added 2 changesets with 2 changes to 2 files</i>	

# Το Mercurial είναι Γρήγορο. Πολύ Γρήγορο

- Η αρχιτεκτονική του ελαχιστοποιεί τα file seeks
- Οι πιο κοινές εργασίες έχουν μικρό κόστος
  - ▶ Δε χρειάζεται να περιμένετε ώρες
- Όλα τα δεδομένα είναι τοπικά
  - ▶ Δε χρειάζεται να περιμένετε για το δίκτυο
- Τα meta-data μοιράζονται μεταξύ των τοπικών repositories
  - ▶ Χρησιμοποιούνται hard-links όπου είναι διαθέσιμα

# Το μέγεθος του repository είναι σημαντικό

- Ένα πλήρες check-out του “HEAD” branch από το CVS είναι περίπου 500 MB
- Όλο το history του FreeBSD src/ “HEAD” branch είναι 400 MB
  - ▶ Πάνω από 15 χρόνια αλλαγές
  - ▶ Πάνω από 40.000 αρχεία
  - ▶ Πάνω από 500 committers
  - ▶ Πάνω από 100.000 commits
- Το πλήρες history στο repository είναι μικρότερο από ένα checkout του τελευταίου source!

# Η Ταχύτητα σε Δράση

- Νούμερα από το laptop μου (Core Duo, 500 MB RAM)
- Mirror από το FreeBSD doc/ tree

## Συχνά tasks

## Χρόνος

hg status	1.33 re 0.65 usr 0.65 sys
hg commit	0.71 re 0.42 usr 0.31 sys
hg annotate	2.76 re 1.59 usr 1.08 sys
hg clone -U	1.67 re 1.23 usr 1.37 sys

## Λιγότερο συχνά tasks

## Χρόνος

hg checkout	6.86 re 2.52 usr 2.24 sys
hg clone http://path	4.18 re 2.86 usr 0.88 sys

# Το Mercurial με Βοηθάει να Είμαι πιο Αποδοτικός

- Η απλή λειτουργία με αφήνει να κάνω αυτό που θέλω
  - ▶ Λιγότερη σκέψη για το πως δουλεύει το SCM, περισσότερη για τη δουλειά μου
- Τα commits και τα merges είναι ξεχωριστά
  - ▶ Είναι πιο δύσκολο να χάσω ή να χαλάσω ότι έχω κάνει επειδή έκανα λάθος merge
- Το μικρό κόστος ενός clone (σε χρόνο και χώρο) βοηθάει
  - ▶ Feature-based clones
  - ▶ Bug-based clones
- Τα local repositories με βοηθούν να δουλεύω οπουδήποτε
  - ▶ Στο τραίνο
  - ▶ Με αργή σύνδεση (dialup)

# Γιατί είναι γρήγορο το Mercurial;

- Κάθε I/O optimized
- Απλά formats
- Αποφεύγει το disk I/O αν μπορεί
- Προτίμηση για “streaming” formats

# Απλά Formats

- Μόνο δύο on-disk data structures
  - ▶ Revlog (file metadata, manifest, changelogs)
  - ▶ Dirstate (περιοχή εργασίας)
  - ▶ Μικρότερη ποικιλία, έμφαση στην ταχύτητα αυτών των δύο δομών
- Τα file formats μπορούν να γίνουν γρήγορα parse με “pure Python”
  - ▶ Λιγότερα Python conditionals: “if” σημαίνει “αργό”
  - ▶ Εκτενής χρήση των struct.pack, string.split

# Αποφυγή disk I/O

- Τα metadata είναι indexed
- Μικρά αρχεία = index & data μαζί
- Dirstate: γρήγορο status χωρίς open/read
  - ▶ Μέγεθος, mtime, mode
  - ▶ Αρκεί η `os.stat()` για τα περισσότερα status checks

# Streaming I/O

- Αλφαβητική σειρά πρόσβασης αρχείων
  - ▶ Τα περισσότερα συστήματα έχουν optimizations για αυτό τον τρόπο προσπέλασης
- Τα metadata είναι γραμμένα με ειδικό τρόπο για linear read
  - ▶ Forward deltas, με περιοδική επανάληψη του full text
- Αλγοριθμικά Optimizations
  - ▶ Αντί για “backwards seek” και διάβασμα
  - ▶ Forward διάβασμα & reverse
  - ▶ x1000 ταχύτητα από το seek bw & read

# Άλλες Χρήσιμες Πληροφορίες

- Web UI: pull / log / diff support
- Hooks: {pre,post}{commit,push,pull,. . . }
- Extensibility με Python modules
  - ▶ GUI εργαλεία
  - ▶ Διαχείριση patches
  - ▶ Αυτοματοποιημένο regression search

# Γιατί Έχουν Σημασία τα Εργαλεία Μας

- Επηρεάζουν τον *τρόπο που εργαζόμαστε*
  - ▶ RCS/SCCS: Μόνιμα σύνδεση με remote host
  - ▶ CVS/SVN: Πρέπει να είμαστε online για να δουλέψουμε
  - ▶ Κατανεμημένα εργαλεία: Μπορούμε να δουλέψουμε *οπουδήποτε και οποτεδήποτε*
- Επηρεάζουν τον τρόπο που οι *άλλοι* συνεργάζονται με μας
- RCS/SCCS: οι “εξωτερικοί” συνεργάτες δεν βλέπουν το ιστορικό
- CVS/SVN: οι εξωτερικοί συνεργάτες μπορούν να δουν το ιστορικό αλλά δε μπορούν να κάνουν καμία αλλαγή
- Κατανεμημένα εργαλεία: Δεν υπάρχουν “outsiders”

# Συνηθισμένες Αντιρρήσεις

- “Φοβάμαι ότι το project μου θα κάνει fork”
  - ▶ Μπορώ ήδη να πάρω όλο το ιστορικό σας από το CVS & SVN, να το κάνω import και να κάνω fork
  - ▶ Το *forking* είναι κυρίως κοινωνικό πρόβλημα, δε λύνεται με τεχνικά μέσα
  - ▶ Τα distributed tools κάνουν την *επανασύνδεση* μετά από ένα fork πιο εύκολη
- “Θέλω κάτι απλό”
  - ▶ Τα καλά distributed εργαλεία είναι *πιο απλά* από τα κεντρικοποιημένα αντίστοιχα
  - ▶ Προτιμάτε ένα κεντρικά ελεγχόμενο μοντέλο; Δε σας εμποδίζει τίποτα

# Τα Κατανεμημένα Εργαλεία και το Free Software

- **Εγγενώς και πολύ καλύτερα** από τα κεντρικοποιημένα εργαλεία
- **Αναιρούν τον διαχωρισμό** μεταξύ insiders/outsideers
- **Επιτρέπουν να διαλέξετε εσείς** το μοντέλο που θέλετε
- **Κάθε χρήστης** είναι ένας εν δυνάμει developer
- **Η καλή υποστήριξη για branching** σας αφήνει να πειραματιστείτε ελεύθερα, και χωρίς scalability προβλήματα

# Το Μέλλον του Mercurial

- Καλό GUI support για Windows
- Integration με τα δημοφιλή IDE (Eclipse, NetBeans)

# User quotes: η κοινότητα

- Οι developers είναι πάρα πολύ καλοί, και με βοηθάνε πάντα
- Η κοινότητα του Mercurial είναι από τις πιο ευγενικές που έχω συναντήσει

## User quotes: το Mercurial

- Μου πήρε ακριβώς 5 λεπτά να το μάθω. Είναι φανταστικό!
- Δεν πίστευα ότι το clone έγινε. Το ξανάτρεξα, γιατί νόμισα ότι δεν έκανε τίποτα. Τόσο γρήγορο ήταν...
- Με εργαλεία σαν το Mercurial, δε χρειάζεται κανείς πλέον να φοβάται τα merges.

# Ευχαριστώ!