

Distributed Version Control with Mercurial

Martin Geisler
<mg@aragost.com>

ClearCase in the 21st Century, Zurich
February 3rd, 2012



About the Speaker

Martin Geisler:

- ▶ core Mercurial developer:
 - ▶ reviews patches from the community
 - ▶ helps users in our IRC channel
- ▶ works at aragost Trifork, Zurich:
 - ▶ offers professional Mercurial support
 - ▶ customization, migration, training
 - ▶ advice on best practices



Outline

Introduction

Mercurial

Centralized vs Distributed

Key Mercurial Concepts

Using Mercurial

Workflows

Branches

Frontends

Comparison with Git

Wrapping Up



Outline

Introduction

Mercurial

Centralized vs Distributed

Key Mercurial Concepts

Using Mercurial

Workflows

Branches

Frontends

Comparison with Git

Wrapping Up



Outline

Introduction

Mercurial

Centralized vs Distributed

Key Mercurial Concepts

Using Mercurial

Workflows

Branches

Frontends

Comparison with Git

Wrapping Up



What is Mercurial?

Main features:

- ▶ fast, distributed revision control system
 - ▶ robust support for branching **and** merging
 - ▶ free and open source
 - ▶ commercial support available: aragost Trifork and others



What is Mercurial?

Main features:

- ▶ fast, distributed revision control system
 - ▶ robust support for branching and merging
 - ▶ free and open source
 - ▶ commercial support available: aragost Trifork and others
- ▶ installers for Windows, Mac OS X, Linux, ...
 - ▶ **TortoiseHg** is a cross-platform graphical frontend
 - ▶ **Machg** and **SourceTree** are fast native Mac OS X frontends
 - ▶ plugins for MS Visual Studio, Eclipse, ...



What is Mercurial?

Main features:

- ▶ fast, distributed revision control system
 - ▶ robust support for branching and merging
 - ▶ free and open source
 - ▶ commercial support available: aragost Trifork and others
- ▶ installers for Windows, Mac OS X, Linux, ...
 - ▶ **TortoiseHg** is a cross-platform graphical frontend
 - ▶ **Machg** and **SourceTree** are fast native Mac OS X frontends
 - ▶ plugins for MS Visual Studio, Eclipse, ...
- ▶ very user-friendly
 - ▶ extensive built-in help for all commands
 - ▶ command set resembles CVS and SVN
 - ▶ destructive commands delegated to extensions



Who is Using it?

Mercurial is used by:

- ▶ Oracle for Java, OpenSolaris, NetBeans, OpenOffice, ...
- ▶ Mozilla for Firefox, Thunderbird, ...
- ▶ Google
- ▶ many more...



Outline

Introduction

Mercurial

Centralized vs Distributed

Key Mercurial Concepts

Using Mercurial

Workflows

Branches

Frontends

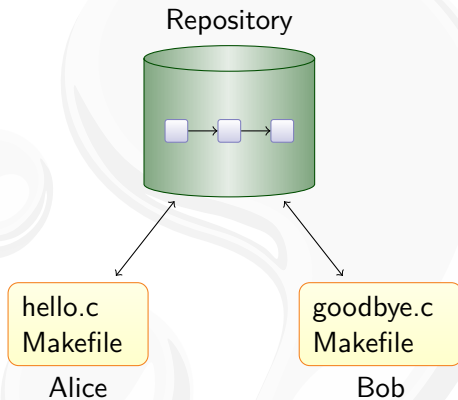
Comparison with Git

Wrapping Up



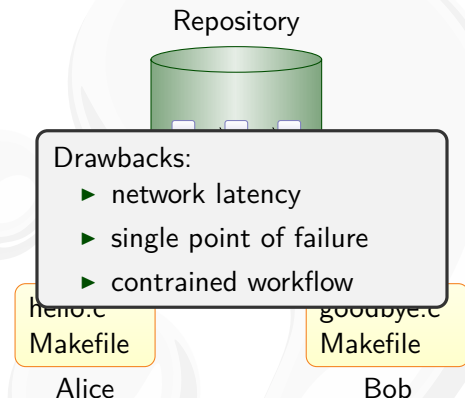
Centralized Revision Control

Single repository, multiple working copies:



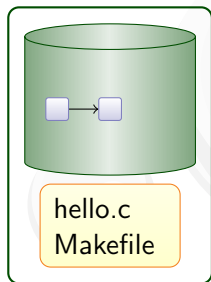
Centralized Revision Control

Single repository, multiple working copies:

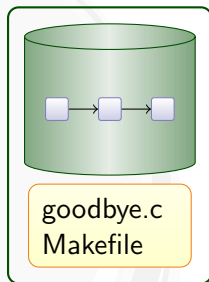


Distributed Revision Control

Mercurial duplicates the history on many servers:



Alice

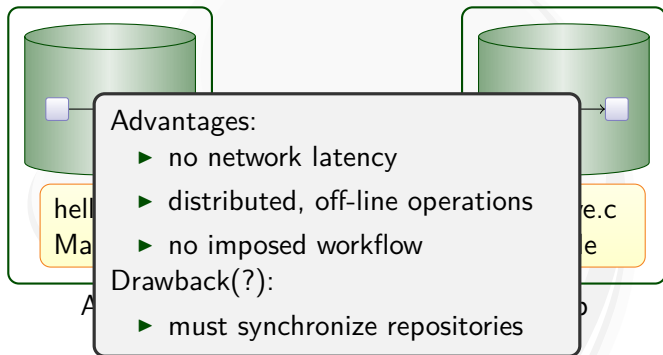


Bob



Distributed Revision Control

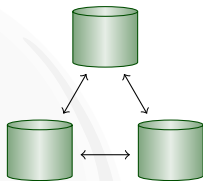
Mercurial duplicates the history on many servers:



Why Distributed?

Distributed revision control gives you:

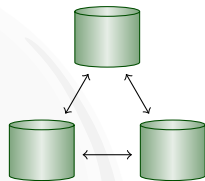
- ▶ offline commits
- ▶ rich set of fast local operations
- ▶ great flexibility



Why Distributed?

Distributed revision control gives you:

- ▶ offline commits
- ▶ rich set of fast local operations
- ▶ great flexibility



Derived effects:

- ▶ fine-grained commits
- ▶ searchable history
- ▶ branching and merging become a natural task (not something to be feared)
- ▶ enables better workflows



Outline

Introduction

Mercurial

Centralized vs Distributed

Key Mercurial Concepts

Using Mercurial

Workflows

Branches

Frontends

Comparison with Git

Wrapping Up



Moving Changesets Around

Pull and merge:

Alice



Bob



Moving Changesets Around

Pull and merge:

Alice



`commit`

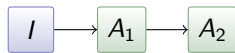
Bob



Moving Changesets Around

Pull and merge:

Alice



`commit`

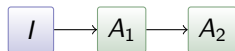
Bob



Moving Changesets Around

Pull and merge:

Alice



Bob

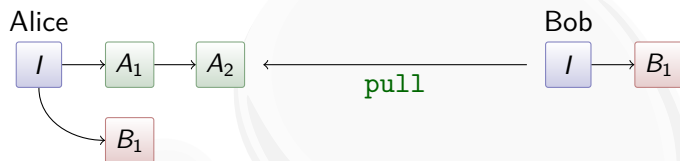


`commit`



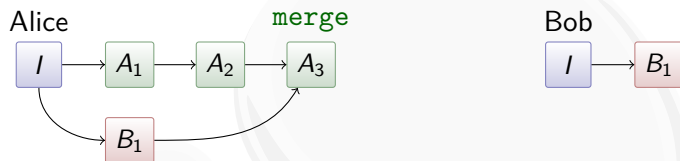
Moving Changesets Around

Pull and merge:



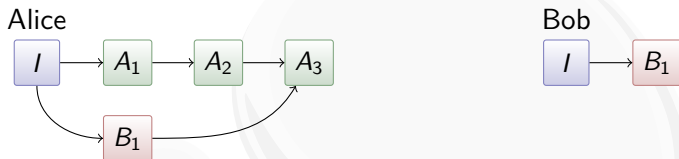
Moving Changesets Around

Pull and merge:



Moving Changesets Around

Pull and merge:



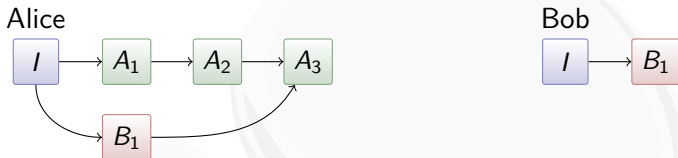
Changesets:

- ▶ atomic repository-wide snapshot
- ▶ very similar to a **baseline** in ClearCase UCM



Moving Changesets Around

Pull and merge:



Changesets:

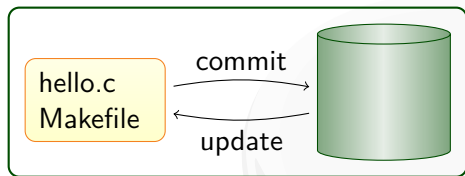
- ▶ atomic repository-wide snapshot
- ▶ very similar to a **baseline** in ClearCase UCM

Merging:

- ▶ find common ancestor of *A*₂ and *B*₁: *I*
- ▶ do three-way merge between *I*, *A*₂, and *B*₁
- ▶ merge often ⇒ common ancestor is close to tip



Key Mercurial Commands



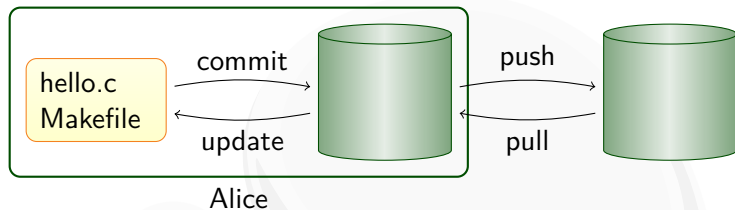
Alice

Local commands:

- ▶ `hg commit`: save a snapshot into the current repository
- ▶ `hg update`: checkout revision into working directory
- ▶ `hg merge`: join different lines of history



Key Mercurial Commands



Local commands:

- ▶ `hg commit`: save a snapshot into the current repository
- ▶ `hg update`: checkout revision into working directory
- ▶ `hg merge`: join different lines of history

Network commands:

- ▶ `hg pull`: retrieve changesets from another repository
- ▶ `hg push`: send your changesets to another repository

Outline

Introduction

Mercurial

Centralized vs Distributed

Key Mercurial Concepts

Using Mercurial

Workflows

Branches

Frontends

Comparison with Git

Wrapping Up



Outline

Introduction

Mercurial

Centralized vs Distributed

Key Mercurial Concepts

Using Mercurial

Workflows

Branches

Frontends

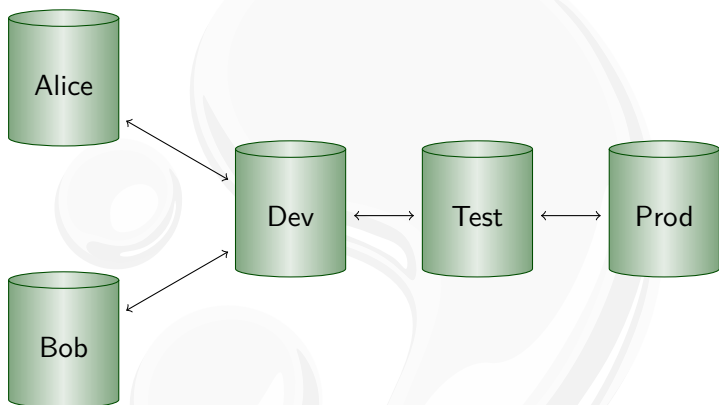
Comparison with Git

Wrapping Up



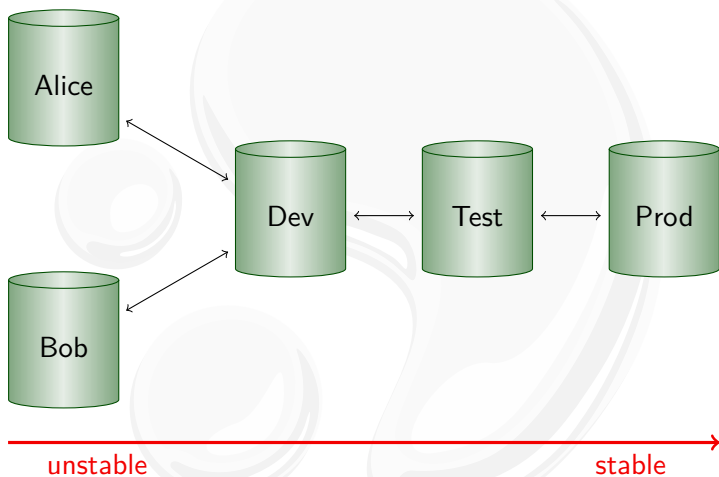
Workflow in a Team

Mercurial scales from a single team. . . :



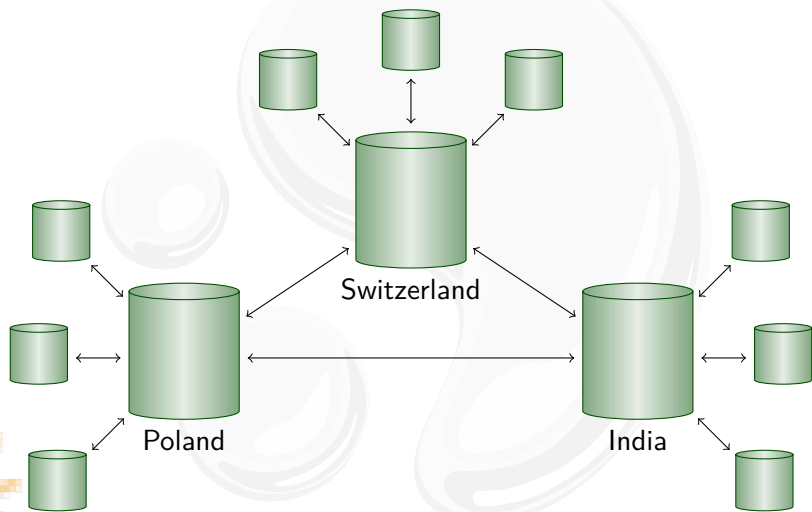
Workflow in a Team

Mercurial scales from a single team. . . :



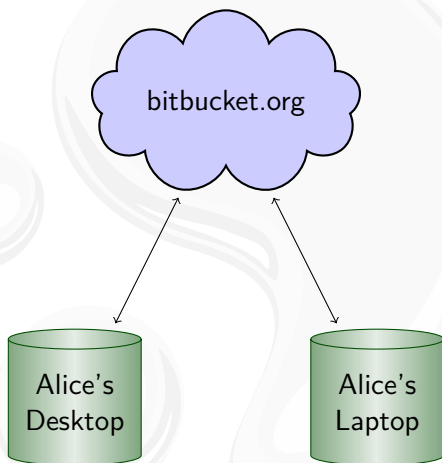
Workflow Between Company Divisions

... to enterprise-wide development. ... :



Workflow Between Two Computers

... to working with yourself:



Outline

Introduction

Mercurial

Centralized vs Distributed

Key Mercurial Concepts

Using Mercurial

Workflows

Branches

Frontends

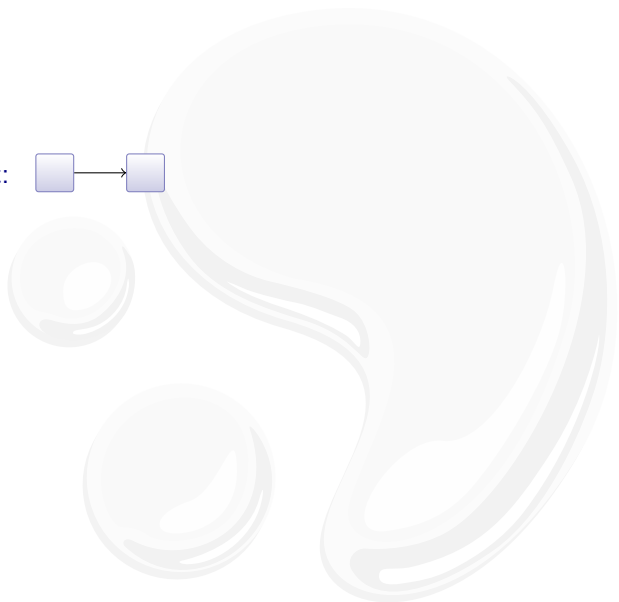
Comparison with Git

Wrapping Up



Release Branches

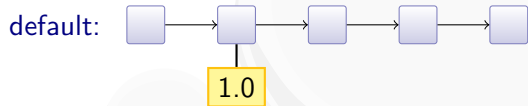
default: 



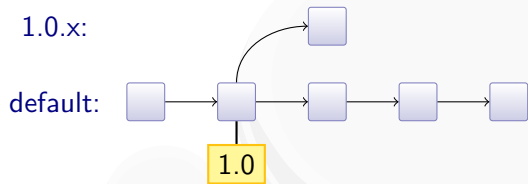
Release Branches



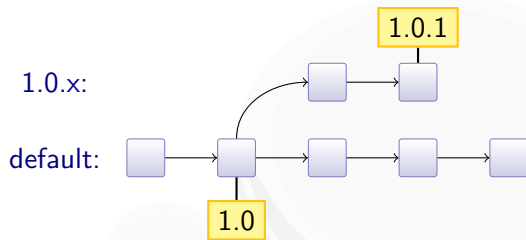
Release Branches



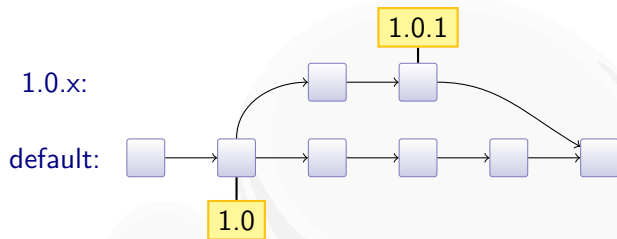
Release Branches



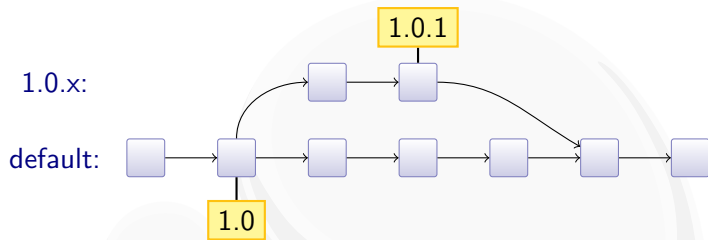
Release Branches



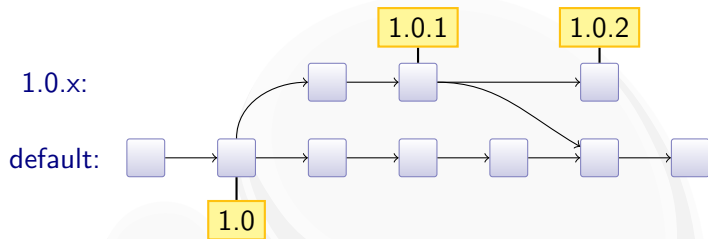
Release Branches



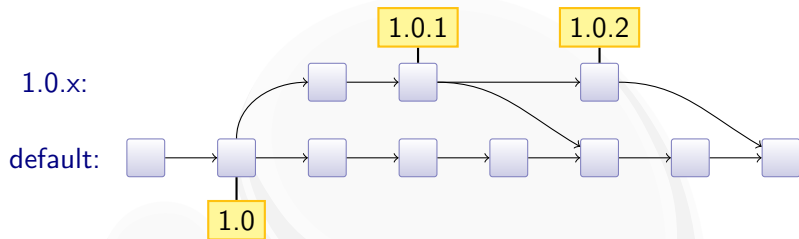
Release Branches



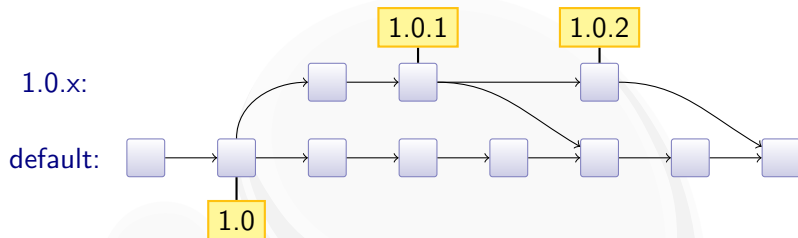
Release Branches



Release Branches



Release Branches



Named branches:

- ▶ heavy-weight branches, integral part of each changeset
- ▶ allow tracking and auditing of old history



Outline

Introduction

Mercurial

Centralized vs Distributed

Key Mercurial Concepts

Using Mercurial

Workflows

Branches

Frontends

Comparison with Git

Wrapping Up



Third-Party Tools

Mercurial is mature and has wide-spread tool support:

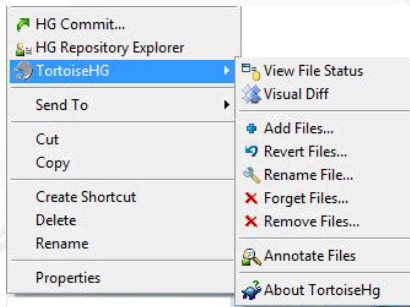
- ▶ Graphical frontends: TortoiseHg, MacHg, SourceTree, ...
- ▶ IDEs: Eclipse, NetBeans, IntelliJ, Visual Studio, ...
- ▶ Project Support: Trac, JIRA, Maven, Hudson, ...



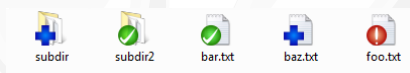
Mercurial for Windows: TortoiseHg



Context menu in Windows Explorer:



Overlay icons:



Mercurial for Windows: TortoiseHg



Browsing history:

The screenshot shows the TortoiseHg Workbench interface. The main window displays a graph of the repository history and a table of recent changesets. The 'Repository Registry' on the left shows the current repository 'cpython' selected. The 'Output Log' at the bottom shows the commit message for changeset 68049.

Rev	Branch	Description	Author	Age	Tags
68042+	default	★ Working Directory ★	Adrian Buehlmann	1 second ago	
68049	trunk	Start working on Python 2.8 :-)	Martin v. Löwis	2 days ago	tip
68048	3.0	Close branch 3.0	Antoine Pitrou	2 days ago	
68047	2.4	Close branch 2.4	Antoine Pitrou	2 days ago	
68046	2.3	Close branch 2.3	Antoine Pitrou	2 days ago	
68045	2.2	Close branch 2.2	Antoine Pitrou	2 days ago	
68044	2.1	Close branch 2.1	Antoine Pitrou	2 days ago	
68043	2.0	Close branch 2.0	Antoine Pitrou	2 days ago	
68042	default	default: Spaces -> tab.	Martin v. Löwis	2 days ago	
68041	trunk...	Close the "trunk" branch.	Georg Brandl	2 days ago	
68040	default	Merge Borland C change.	Georg Brandl	2 days ago	
68039	3.2	Remove Borland C support.	Georg Brandl	2 days ago	
68038	default	Merge 3.2 branch into default.	Georg Brandl	2 days ago	

Changeset: 68049 (5ef18f50529a) Start working on Python 2.8 :-)
Branch: trunk
User: Martin v. Löwis <martin@v.loewis.de>
Date: 2011-02-26 10:45:45 +0100 (2 days ago)
Parent: 68041 (41071f447b15) Close the "trunk" branch.

```
Misc/NEWS
00 -4,6 +4,11 00 Python News

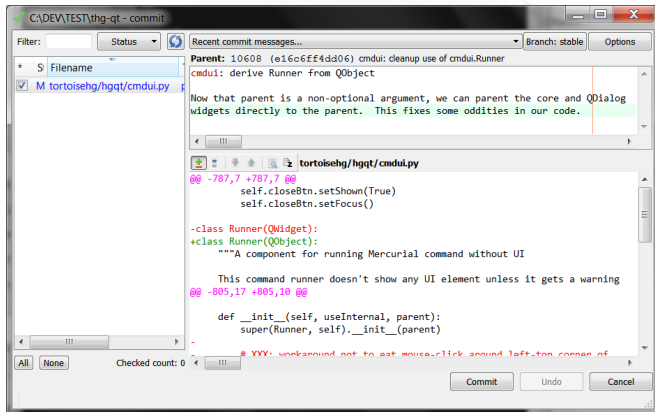
(editors: check NEWS.help for information about editing NEWS using ReST.

+What's New in Python 2.8?
+-----
+
+*Release date: 2010-07-03*
```


Mercurial for Windows: TortoiseHg



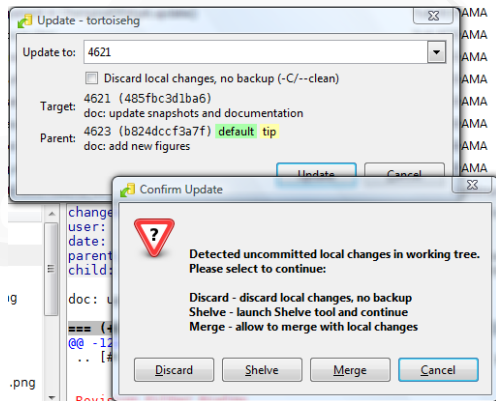
Interactive commit tool:



Mercurial for Windows: TortoiseHg



Update with shelf option:



Mercurial for Windows: TortoiseHg



Detecting renames based on file similarity:

Detect Copies/Renames in thg

Unrevised Files

- tortoisehg/util/cache.py

Candidate Matches

Source	Dest	% Match
tortoisehg/util/cachethg.py	tortoisehg/util/cache.py	100%

Min Similarity: 50% Only consider deleted files

Differences from Source to Dest


Mercurial for Mac OS X: MacHg



Cloning a repository:

Clone "hgcollapse"

Enter a file path of a directory on your local filesystem, along with the shortname you want to use to refer to the cloned repository. Mercurial will clone the source repository to the new given location and start managing the cloned repository.

Source:  hgcollapse

Destination:

ShortName:

Local Path: Browse...

Advanced Options: ▾

Use SSH command

Clone only Revision, Tag or Branch

Clone only Specific Revision(s)

Remote Command

Repository Only Use Pull Protocol Uncompressed



Mercurial for Mac OS X: MacHg



Updating to a revision:

Updating All Files in hg-sources

Select Revision to Update to:

revi...	graph	author	date	labels	description	changeset
10303		FUJIIWARA I	3 mont...		i18n-ja: synchronized with e7727a545c48	35bc3152e67
10304		Georg Bran	3 mont...		i18n: fix typo in German translation	0824aed0079
10305		Martin Geis	2 mont...		Merge with stable.	81211cf5703b
10306		Wagner Bru	2 mont...		i18n-pt_BR: synchronized with b08ffd27dfc8	e7639acd54f6
10307		Wagner Bru	2 mont...		merge with i18n stable	b0298eaeaf7
10308		Matt Mackz	2 mont...	1.4.3	Merge with i18n	4aa619c4c2c0
10309		Matt Mackz	2 mont...		Merge with i18n	0df9e63f0c87
10310		Matt Mackz	2 mont...		Added tag 1.4.3 for changeset 4aa619c4c2c0	61750d3aa00c
10311		Matt Mackz	2 mont...		Added signature for changeset 4aa619c4c2c0	7c5eb0988e7a
10312		Matt Mackz	2 mont...		Merge with stable	131a012aa878

Changeset: 10310 : 61750d3aa00c
Parents: 10308:4aa619c4c2c0
Author: Matt Mackall
Date: 2 months ago (2010-02-01 09:34 -0600)
Modified: .hgtags
Description: Added tag 1.4.3 for changeset 4aa619c4c2c0

Clean (delete modified files)

What Will Happen: The repository will be restored to the state of version 10310. Any modified files will be moved aside.

Mercurial for Mac OS X: MacHg



Collapsing changesets:

Collapsing Selected Revisions in "myrepo"

Select Changesets to Collapse:

revision	graph	author	date	labels	description	changeset
7		jfh	8 weeks...		- change for CD	6a45701c259c
8		jfh	8 weeks...		- change for CC	0fe4cd5df0ef
9		jfh	8 weeks...		- change for CB	6ed1596d9af7
10		jfh	8 weeks...		- change for CA	a2fd0700743f
11		jfh	8 weeks...		- BugFix for CH	c53e5e06c0e
12		jfh	8 weeks...		- BugFix for CI	d2d1f3c8ba70
13		jfh	8 weeks...		- change for CG	36191356c746
14		jfh	8 weeks...		- change for CF	32a2c3c01d32
15		jfh	8 weeks...		- change for CE	dbf40643867d
16		jfh	31 hour...	default, tip	- A simple change	3adf10560070

Changeset: 16 : 3adf10560070
Tags: tip
Branch: default
Parents: 7:6a45701c259c
Author: jfh

What Will Happen: The revisions from 13 through to 14 will be removed and combined into a single revision which will be inserted in place of the selected range of revisions. This collapse operation is destructive; it rewrites history. Therefore you should never collapse any revisions that have already been pushed to other repositories, unless you really know what you are doing.

Cancel Collapse



Mercurial for Mac OS X: MacHg



Removing changesets from a repository:

Stripping Selected Revisions in "myrepo"

Select Changesets to Strip:

revision	graph	author	date	labels	description	changeset
3		jfh	8 weeks...		- minor change	3d2c3a4e43a0
4		jfh	8 weeks...		- change for four	8d9f9888drcf
5		jfh	8 weeks...		- change for five	11492430c116
6		jfh	8 weeks...		- change for six	6f3d5bcfa00b
7		jfh	8 weeks...		- change for CD	6a45701c259c
8		jfh	8 weeks...		- change for CC	0fe4cd5df0ef
9		jfh	8 weeks...		- change for CB	6ed1596d9af7
10		jfh	8 weeks...		- change for CA	a2fd8700743f
11		jfh	8 weeks...		- BugFix for CH	c53e5e0a6c0e
12		jfh	8 weeks...		- BugFix for CI	d2d1f3c8ba70
13		jfh	8 weeks...		- change for CG	36191356c746
14		jfh	8 weeks...		- change for CF	32a2c3c01d32
15		jfh	8 weeks...		- change for CE	dbf40643867d
16		jfh	31 hour...		- A simple change	3adf10560070
17		jfh	31 hour...	default, tip	- Commit the merge	15c059c9c3ce

Commit: 9 (jfh), 8 weeks ago
Description: - change for CB
Commit: 10 (jfh), 8 weeks ago
Description: - change for CA
Commit: 11 (jfh), 8 weeks ago
Description: - BugFix for CH
Commit: 12 (jfh), 8 weeks ago
Description: - BugFix for CI

What Will Happen: The selected revisions within 9 through to 17 will be removed from the repository. This strip operation is destructive; it rewrites history. Therefore you should never strip any revisions that have already been pushed to other repositories, unless you really know what you are doing.

Cancel Strip

Outline

Introduction

Mercurial

Centralized vs Distributed

Key Mercurial Concepts

Using Mercurial

Workflows

Branches

Frontends

Comparison with Git

Wrapping Up



Git

Git was created by Linus Torvalds for Linux Kernel development:

- ▶ focus on speed
- ▶ builds on the same concepts as Mercurial:
 - ▶ fully distributed with push/pull between clones
 - ▶ commits are organized in a changeset graph



Git

Git was created by Linus Torvalds for Linux Kernel development:

- ▶ focus on speed
- ▶ builds on the same concepts as Mercurial:
 - ▶ fully distributed with push/pull between clones
 - ▶ commits are organized in a changeset graph

Git is extremely flexible, but flexibility comes at a price:

- ▶ `git log` has 150 different flags, log help text is 1496 lines
- ▶ default command set includes destructive commands
- ▶ steep learning curve — must learn about the “staging area”



Git

Git was created by Linus Torvalds for Linux Kernel development:

- ▶ focus on speed
- ▶ builds on the same concepts as Mercurial:
 - ▶ fully distributed with push/pull between clones
 - ▶ commits are organized in a changeset graph

Git is extremely flexible, but flexibility comes at a price:

- ▶ `git log` has 150 different flags, log help text is 1496 lines
- ▶ default command set includes destructive commands
- ▶ steep learning curve — must learn about the “staging area”

Git is still somewhat Linux-centric:

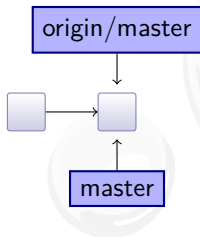
- ▶ started as a tool for Linux Kernel hackers, made by Linux Kernel hackers
- ▶ Git has since then been ported to Windows
- ▶ Windows performance is still reported to be sub-par



Branches in Git

Similar to using bookmarks in Mercurial:

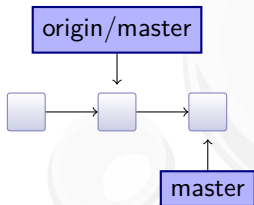
- ▶ branch pointer advances on commit:



Branches in Git

Similar to using bookmarks in Mercurial:

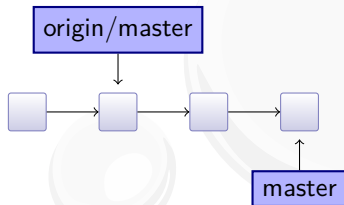
- ▶ branch pointer advances on commit:



Branches in Git

Similar to using bookmarks in Mercurial:

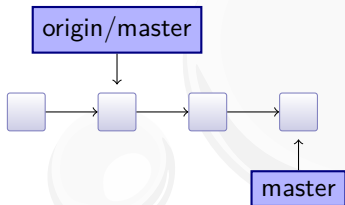
- ▶ branch pointer advances on commit:



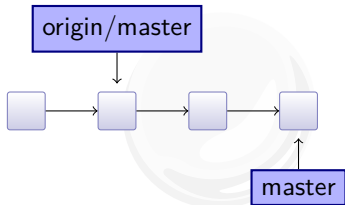
Branches in Git

Similar to using bookmarks in Mercurial:

- ▶ branch pointer advances on commit:



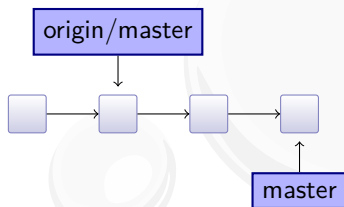
- ▶ pull and merge of changesets from others:



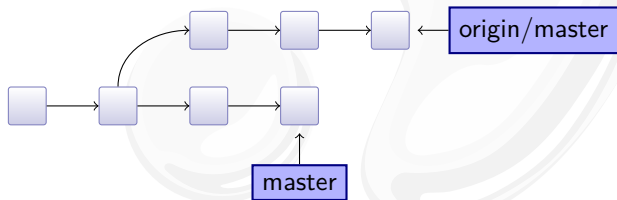
Branches in Git

Similar to using bookmarks in Mercurial:

- ▶ branch pointer advances on commit:



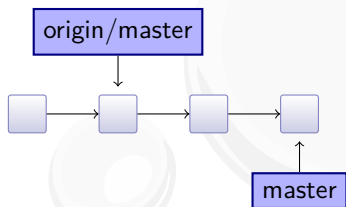
- ▶ pull and merge of changesets from others:



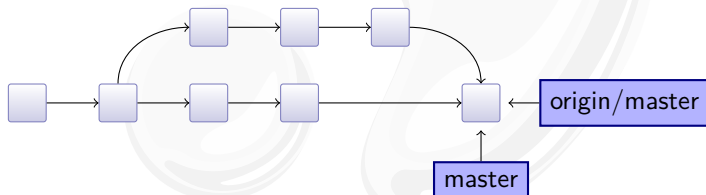
Branches in Git

Similar to using bookmarks in Mercurial:

- ▶ branch pointer advances on commit:



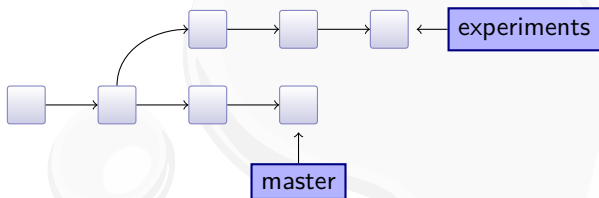
- ▶ pull and merge of changesets from others:



Deleting a Git Branch

Git can **garbage collect** changesets:

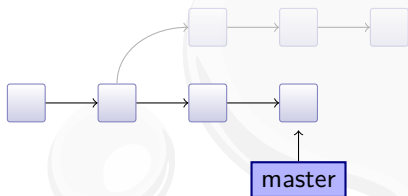
- ▶ repository with experimental branch:



Deleting a Git Branch

Git can **garbage collect** changesets:

- ▶ repository with experimental branch:



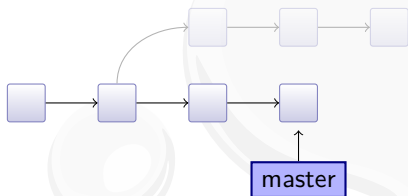
- ▶ delete branch with `git branch -D experiments`
- ▶ faint commits may disappear on next `git gc`



Deleting a Git Branch

Git can **garbage collect** changesets:

- ▶ repository with experimental branch:



- ▶ delete branch with `git branch -D experiments`
- ▶ faint commits may disappear on next `git gc`
- ▶ **delete on server** with `git push origin :experiments`



Outline

Introduction

Mercurial

Centralized vs Distributed

Key Mercurial Concepts

Using Mercurial

Workflows

Branches

Frontends

Comparison with Git

Wrapping Up



Mercurial in a Nutshell

Mercurial changes the way you develop:

- ▶ simple yet strong model for **both** branching and merging
- ▶ power tool instead of necessary evil
- ▶ light-weight and snappy



More Information

- ▶ Mercurial homepage:
<http://mercurial.selenic.com/>
- ▶ *Mercurial: The Definitive Guide*:
<http://hgbook.red-bean.com/>
- ▶ Getting Started:
<http://mercurial.aragost.com/kick-start/>
<http://mercurial.ch/>
<http://hginit.com/>
- ▶ Some free Mercurial hosting sites:
<http://bitbucket.org/>
<http://code.google.com/>
<http://sourceforge.net/>
<http://www.codeplex.com/>



Mercurial Contributors

From <http://ohloh.net/p/mercurial/map:>



Showing 50 of [325 contributors](#).



Mercurial Contributors

From <http://ohloh.net/p/mercurial/map:>



Showing 50 of [325 contributors](#).

Outline

Performance Study: OpenOffice

Subversion and Branches

The Underlying Model

Using History

Changing History

Talking to Other Systems



OpenOffice

Fairly large repository:

- ▶ 70,000 files, 2,0 GB of data
- ▶ 270,000 changesets, 2,3 GB of history



Mercurial is fast on a repository of this size:

```
$ time hg status
0.45s user 0.15s system 99% cpu 0.605 total
$ time hg tip
0.28s user 0.03s system 99% cpu 0.309 total
$ time hg log -r DEV300_m50
0.30s user 0.04s system 99% cpu 0.334 total
$ time hg diff
0.74s user 0.16s system 88% cpu 1.006 total
$ time hg commit -m 'Small change'
1.77s user 0.25s system 98% cpu 2.053 total
```

Outline

Performance Study: OpenOffice

Subversion and Branches

The Underlying Model

Using History

Changing History

Talking to Other Systems



Branches in SVN

Subversion knows nothing about branches!

- ▶ but SVN has a cheap copy mechanism
- ▶ used for tags and branches



Branches in SVN

Subversion knows nothing about branches!

- ▶ but SVN has a cheap copy mechanism
- ▶ used for tags and branches

r10

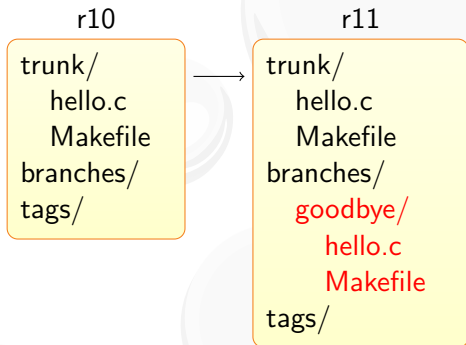
```
trunk/  
  hello.c  
  Makefile  
branches/  
tags/
```



Branches in SVN

Subversion knows nothing about branches!

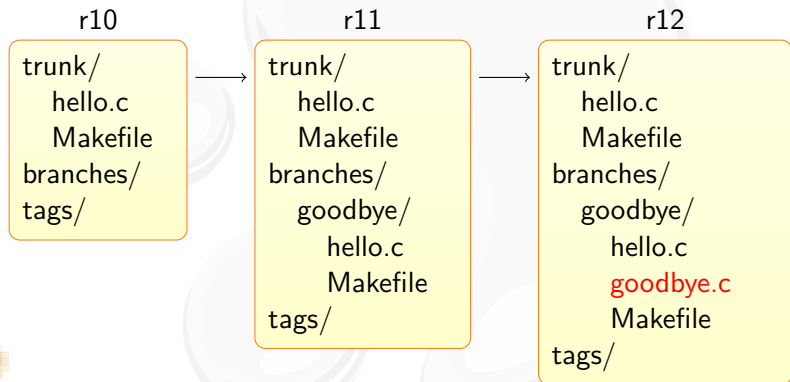
- ▶ but SVN has a cheap copy mechanism
- ▶ used for tags and branches



Branches in SVN

Subversion knows nothing about branches!

- ▶ but SVN has a cheap copy mechanism
- ▶ used for tags and branches



Merging Branches in SVN

The support is incomplete and fragile:

- ▶ renamed files are not merged correctly
- ▶ old clients will not update the merge info



Merging Branches in SVN

The support is incomplete and fragile:

- ▶ renamed files are not merged correctly
- ▶ old clients will not update the merge info

From the SVN Book:

The bottom line is that Subversion's merge-tracking feature has an **extremely complex** internal implementation, and the `svn:mergeinfo` property is the only window the user has into the machinery. Because the feature is **relatively new**, a numbers of edge cases and possible unexpected behaviors may pop up. —*Version Control with Subversion*

(Mercurial has robust built-in support for merging branches.)



Outline

Performance Study: OpenOffice

Subversion and Branches

The Underlying Model

Using History

Changing History

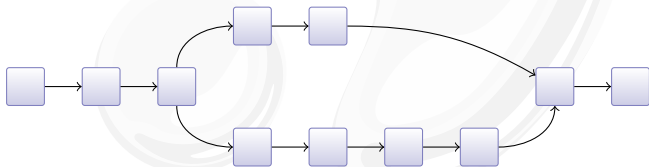
Talking to Other Systems



The Underlying Model

A Mercurial changeset conceptually consist of:

- ▶ 0–2 parent changeset IDs:
 - ▶ root changeset has no parents
 - ▶ normal changesets have one parent
 - ▶ merge changesets have two parents
- ▶ date, username, commit message
- ▶ difference from first parent changeset
- ▶ changeset ID is computed as SHA-1 hash of the above
- ▶ makes it impossible to inject **malicious code** on server



Immutable History

SHA-1 hashes as changeset IDs have some consequences:

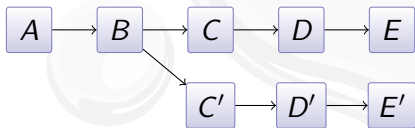
- ▶ a changeset ID is a hash of the entire history
- ▶ changing history changes subsequent changesets
- ▶ history is immutable, you can only make new history:



Immutable History

SHA-1 hashes as changeset IDs have some consequences:

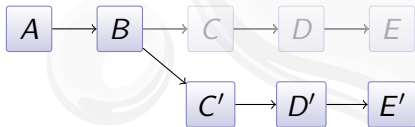
- ▶ a changeset ID is a hash of the entire history
- ▶ changing history changes subsequent changesets
- ▶ history is immutable, you can only make new history:



Immutable History

SHA-1 hashes as changeset IDs have some consequences:

- ▶ a changeset ID is a hash of the entire history
- ▶ changing history changes subsequent changesets
- ▶ history is immutable, you can only make new history:



Outline

Performance Study: OpenOffice

Subversion and Branches

The Underlying Model

Using History

Changing History

Talking to Other Systems



Browsing the History of a File

The `hg annotate` command is invaluable:

- ▶ you see when each line was introduced
- ▶ you can quickly jump back to earlier versions

History of Mercurial's README file:

```
3942: Basic install:
  445:
3942: $ make                # see install targets
3942: $ make install        # do a system-wide install
3942: $ hg debuginstall      # sanity-check setup
3942: $ hg                    # see help
  0:
# ...
```

Better interface in `hg serve`



Searching File Content

Ever wondered when a function was introduced?

- ▶ `hg grep` can help you!

Example: When was `hg forget` introduced?

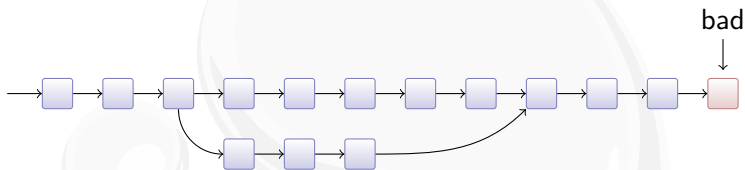
```
$ hg grep --all 'def forget' commands.py
commands.py:8902:+:def forget(ui, repo, *pats, **opts):
commands.py:3522:-:def forget(ui, repo, *pats, **opts):
commands.py:814:-:def forget(ui, repo, file1, *files):
commands.py:814:+:def forget(ui, repo, *pats, **opts):
# ...
```



Revision Graph Bisection

You've found a bug! When was it first introduced?

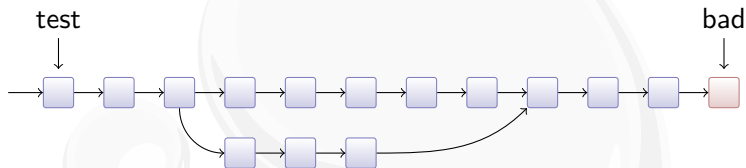
Use `hg bisect` to mark good and bad revisions:



Revision Graph Bisection

You've found a bug! When was it first introduced?

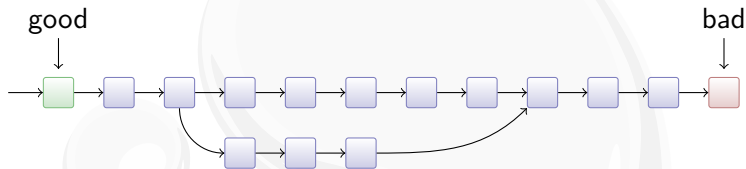
Use `hg bisect` to mark good and bad revisions:



Revision Graph Bisection

You've found a bug! When was it first introduced?

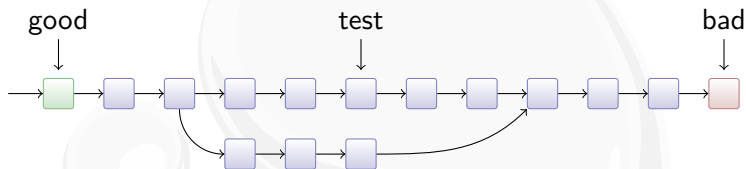
Use `hg bisect` to mark good and bad revisions:



Revision Graph Bisection

You've found a bug! When was it first introduced?

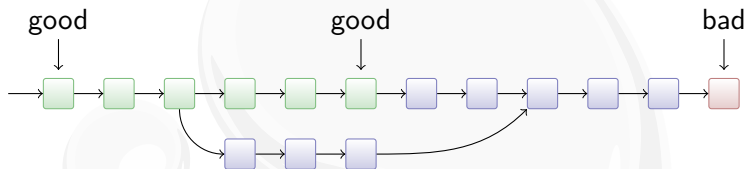
Use `hg bisect` to mark good and bad revisions:



Revision Graph Bisection

You've found a bug! When was it first introduced?

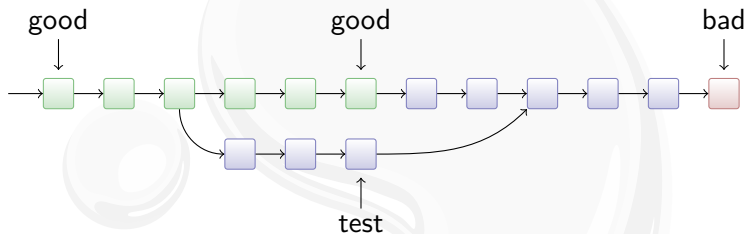
Use `hg bisect` to mark good and bad revisions:



Revision Graph Bisection

You've found a bug! When was it first introduced?

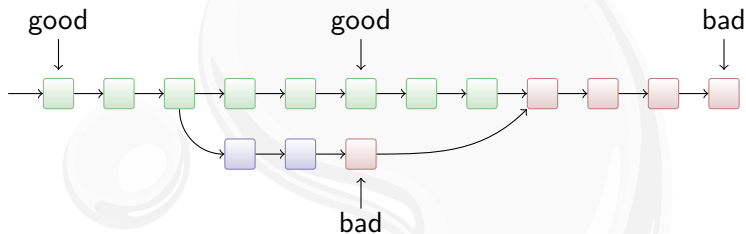
Use `hg bisect` to mark good and bad revisions:



Revision Graph Bisection

You've found a bug! When was it first introduced?

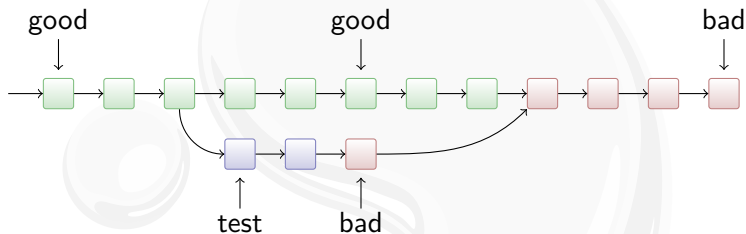
Use `hg bisect` to mark good and bad revisions:



Revision Graph Bisection

You've found a bug! When was it first introduced?

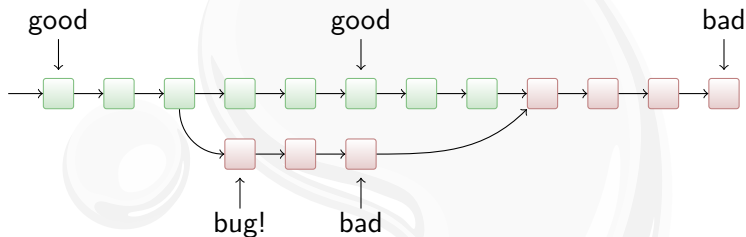
Use `hg bisect` to mark good and bad revisions:



Revision Graph Bisection

You've found a bug! When was it first introduced?

Use `hg bisect` to mark good and bad revisions:



Outline

Performance Study: OpenOffice

Subversion and Branches

The Underlying Model

Using History

Changing History

Talking to Other Systems



Moving Changesets Around

Tired of all those merges? Use the **rebase** extension!

- ▶ Revision graph:



Moving Changesets Around

Tired of all those merges? Use the **rebase** extension!

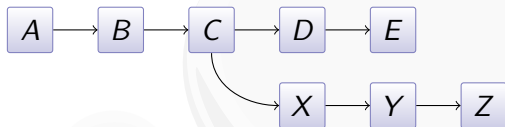
- ▶ Revision graph:



Moving Changesets Around

Tired of all those merges? Use the **rebase** extension!

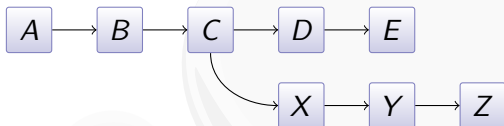
- Revision graph:



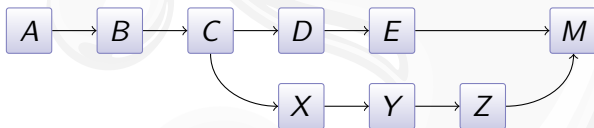
Moving Changesets Around

Tired of all those merges? Use the **rebase** extension!

- ▶ Revision graph:



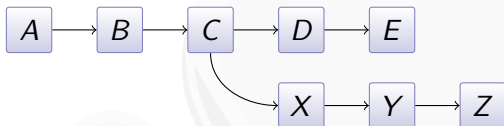
- ▶ Merge:



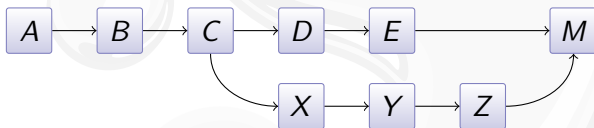
Moving Changesets Around

Tired of all those merges? Use the **rebase** extension!

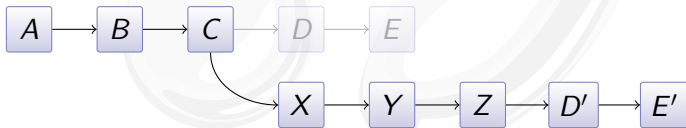
- ▶ Revision graph:



- ▶ Merge:



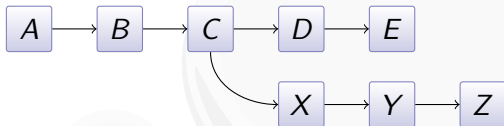
- ▶ Rebase:



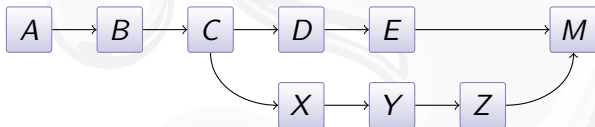
Moving Changesets Around

Tired of all those merges? Use the **rebase** extension!

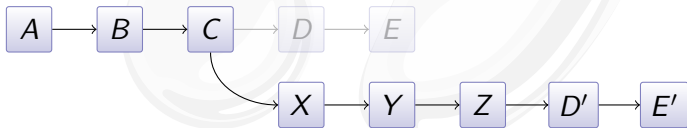
- ▶ Revision graph:



- ▶ Merge:



- ▶ Rebase:



- ▶ Beware: public changes should never be rebased.



Maintaining Patch Series

The `mq` extension makes it easy to maintain a patch series:



Works nicely for local modification for upstream sources.



Maintaining Patch Series

The `mq` extension makes it easy to maintain a patch series:



The diagram illustrates a patch series. At the top is a green rounded rectangle labeled 'code'. Below it is a purple square with an upward-pointing arrow. Below the arrow are three vertical dots. This represents a sequence of patches being applied to the code.

Works nicely for local modification for upstream sources.



Maintaining Patch Series

The `mq` extension makes it easy to maintain a patch series:

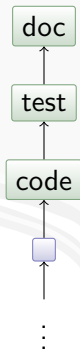


Works nicely for local modification for upstream sources.



Maintaining Patch Series

The `mq` extension makes it easy to maintain a patch series:

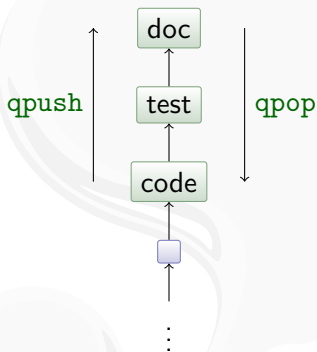


Works nicely for local modification for upstream sources.



Maintaining Patch Series

The `mq` extension makes it easy to maintain a patch series:



Works nicely for local modification for upstream sources.



Editing History

Inspired by `git rebase -i`, `histedit` lets you

- ▶ reorder changesets:



Editing History

Inspired by `git rebase -i`, `histedit` lets you

- ▶ reorder changesets:



- ▶ fold changesets:



Editing History

Inspired by `git rebase -i`, `histedit` lets you

- ▶ reorder changesets:



- ▶ fold changesets:



- ▶ drop changesets:



Editing History

Inspired by `git rebase -i`, `histedit` lets you

- ▶ reorder changesets:



- ▶ fold changesets:



- ▶ drop changesets:



- ▶ edit changesets:



Outline

Performance Study: OpenOffice

Subversion and Branches

The Underlying Model

Using History

Changing History

Talking to Other Systems



Migrating History

The **convert** extension can import history:

- ▶ CVS, SVN, Git, Bazaar, Darcs, ...
- ▶ incremental conversion
- ▶ many options for fiddling with branches, authors, ...



Migrating History

The `convert` extension can import history:

- ▶ CVS, SVN, Git, Bazaar, Darcs, ...
- ▶ incremental conversion
- ▶ many options for fiddling with branches, authors, ...

Interestingly, `convert` can import from Mercurial:

- ▶ `--filemap` lets you exclude and rename files
- ▶ `--branchmap` lets you rename branches



Interfacing with Subversion

The `hgsubversion` extension let's you:

- ▶ use `hg clone` on a SVN URL
- ▶ use `hg pull` to convert new SVN revisions
- ▶ use `hg push` to commit changesets to SVN server

Goal: make `hg` a better Subversion client than `svn`!



Interfacing with Git

Need to work on a Git repository? Try `hg-git`!

- ▶ Mercurial extension: you get the nice `hg` command line
- ▶ round-tripping: changeset hashes are preserved

