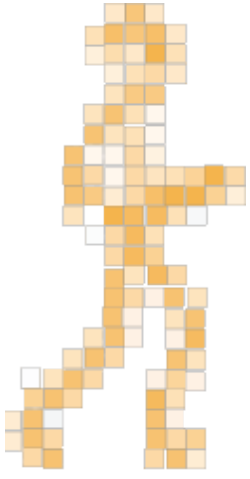
Fast, Flexible and Fun: Revision Control with Mercurial

Martin Geisler
⟨mg@aragost.com⟩

UBS GeekNight
April 27, 2010



About the Speaker

Martin Geisler:

- core Mercurial developer:
  - reviews patches from the community
  - helps users in our IRC channel
- PhD in Computer Science from Aarhus University, Denmark
- now working at aragost Trifork, Switzerland

Outline

Outline

What is Mercurial?

Main features:

- fast, distributed revision control system
- robust support for branching and merging
- very flexible and extensible

Strong focus on back- and forwards compatibility:

- new clients can read/write all old formats on disk
- old clients can read/write to all new servers

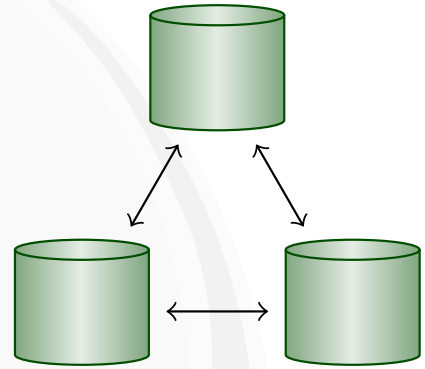
Strong focus on data safety:

- files are not overwritten, only appended to
- easier to recover from disk crashes

Why Distributed?

Distributed revision control gives you:

- offline commits
- rich set of fast local operations

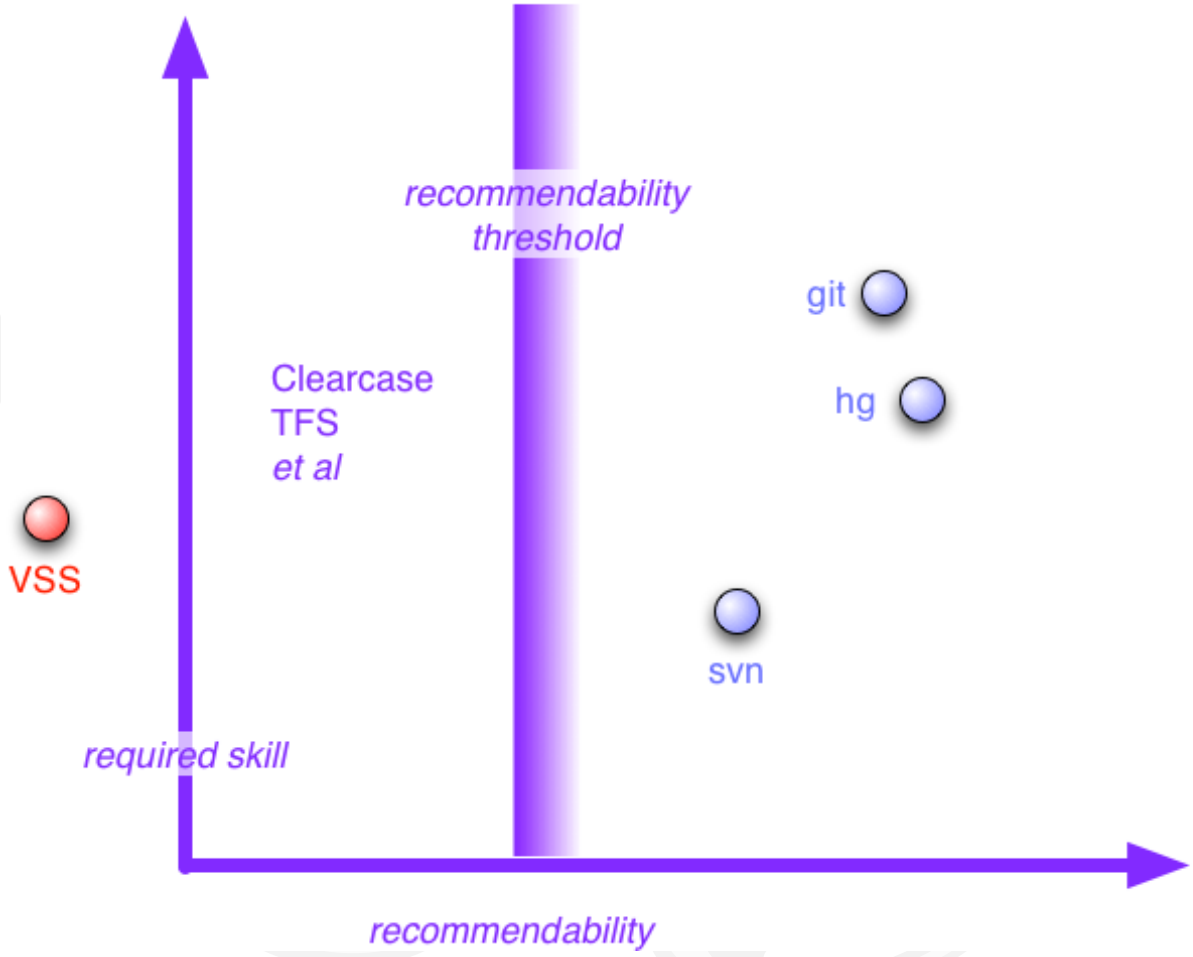


Derived effects:

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

Testimonials

Martin Fowler, 2010: http://martinfowler.com/



Testimonials

Joel Spolsky, 2010: http://www.joelonsoftware.com/

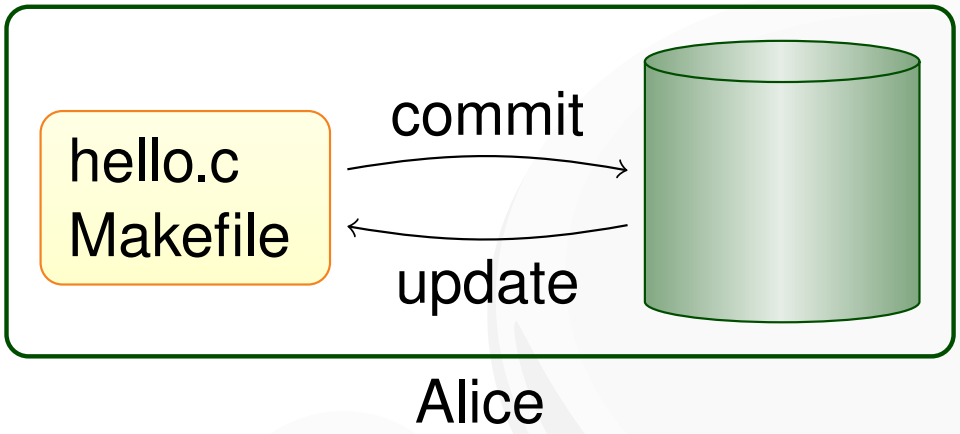
> Mercurial is better than Subversion.
> It is a better way of working on source code with a team. It is a better way of working on source code by yourself.
> It is just better.

Who is Using it?

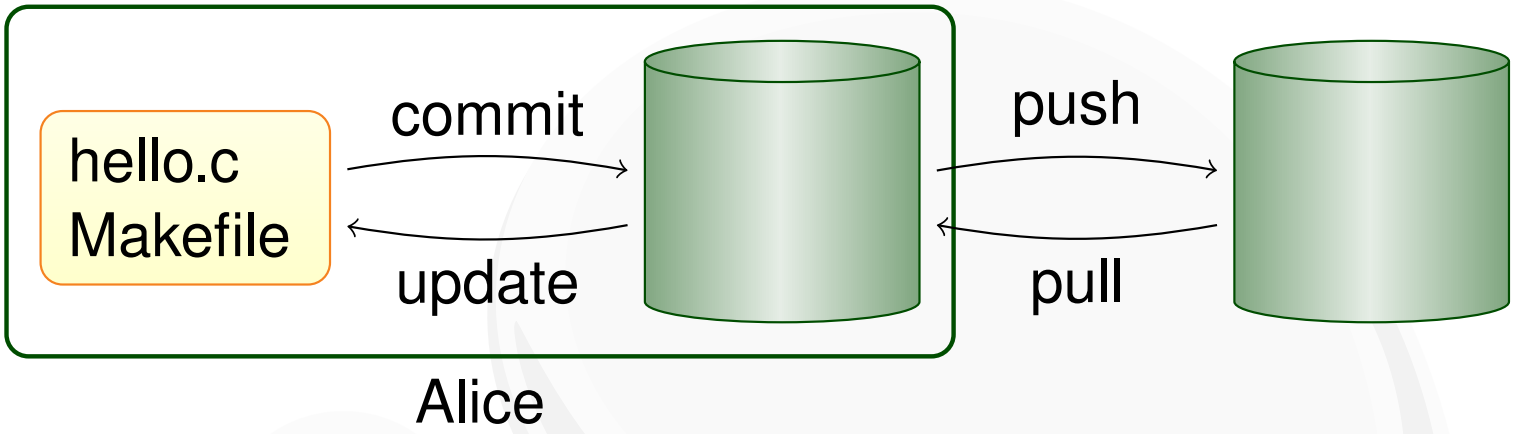
Mercurial is used by:

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

Key Concepts



Key Concepts



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.

Moving Changesets Around

People have read-only access (e.g., `hg serve`):

Alice

0

Bob

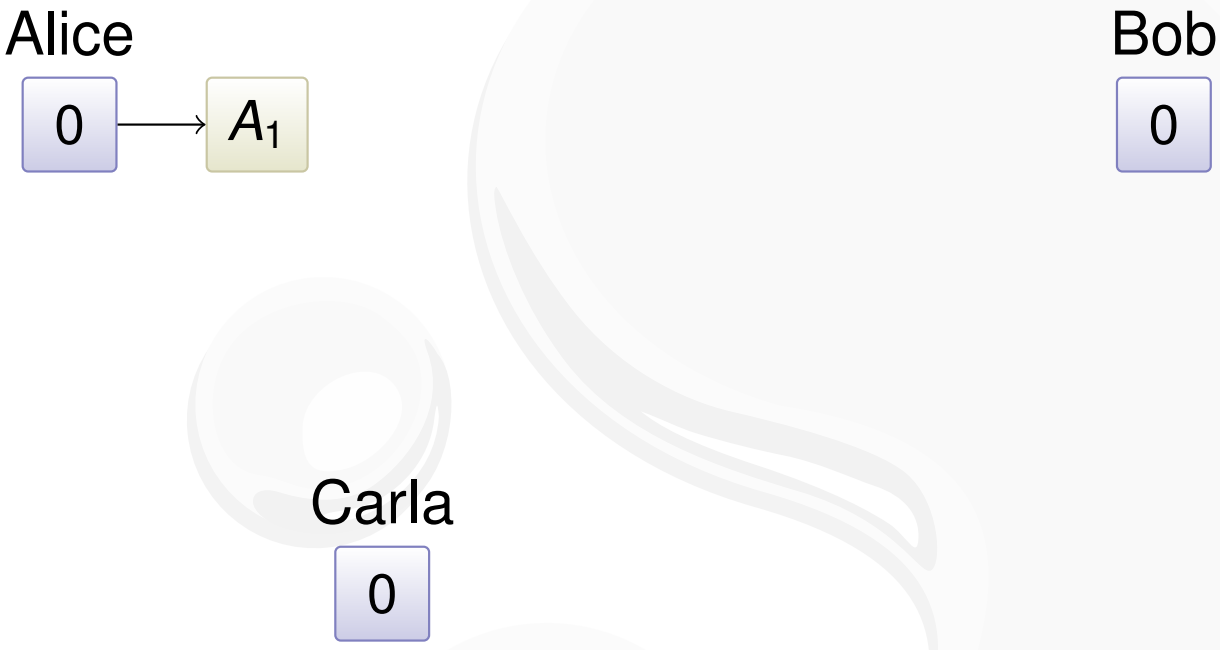
0

Carla

0

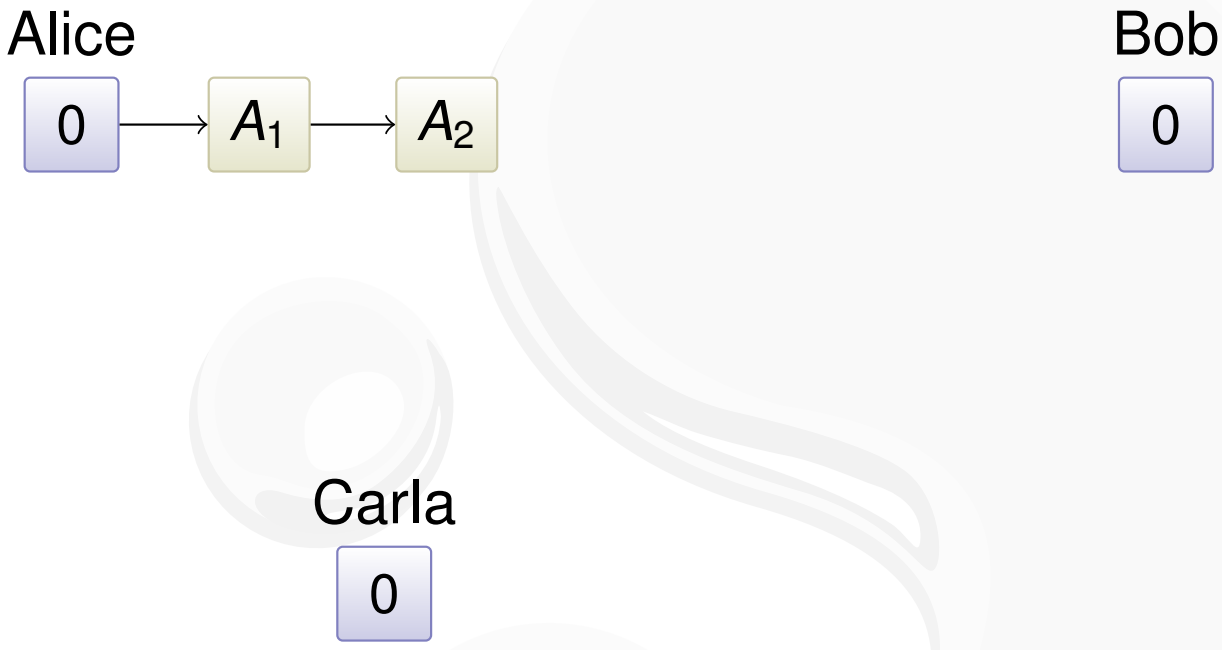
Moving Changesets Around

People have read-only access (e.g., `hg serve`):



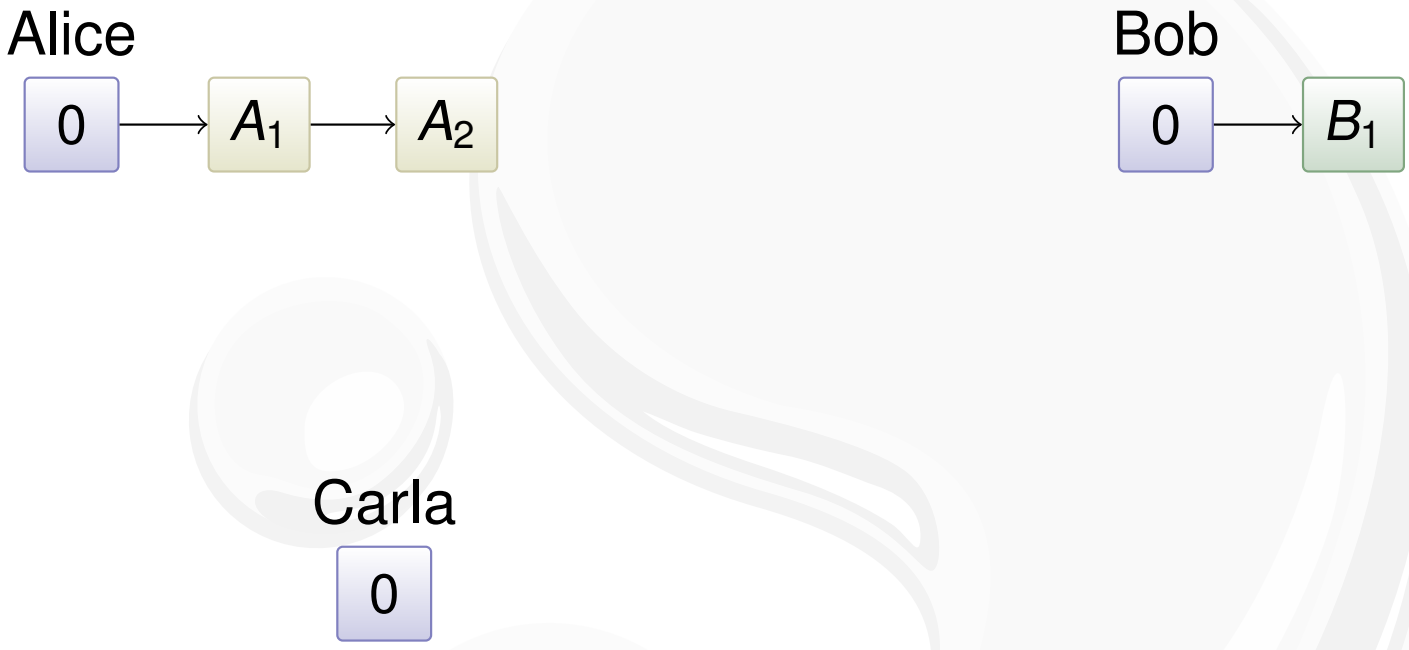
Moving Changesets Around

People have read-only access (e.g., `hg serve`):



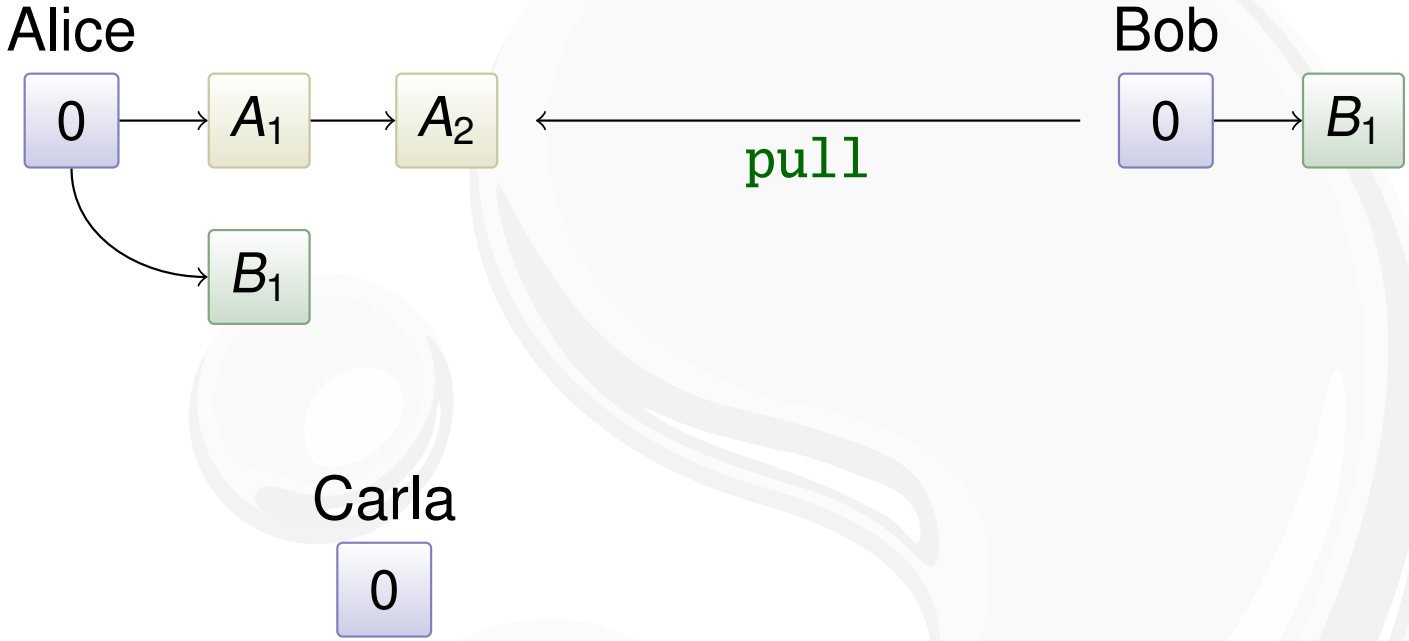
Moving Changesets Around

People have read-only access (e.g., `hg serve`):



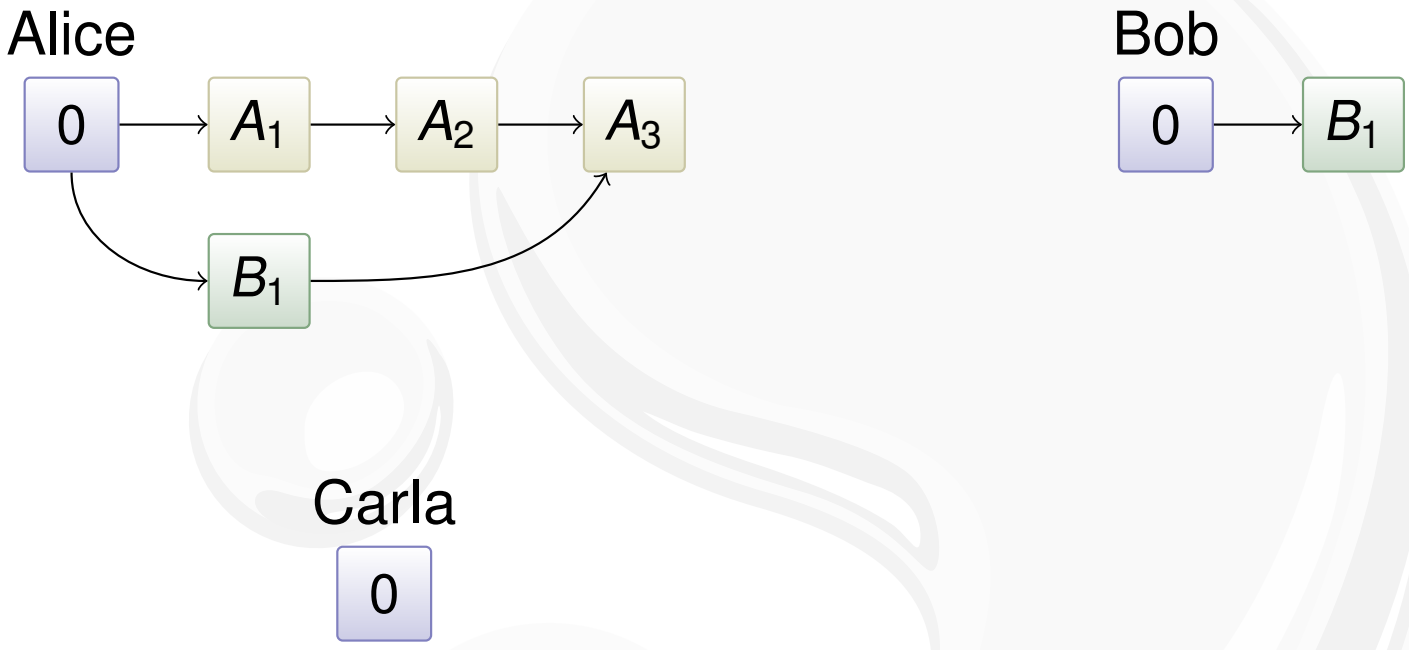
Moving Changesets Around

People have read-only access (e.g., `hg serve`):



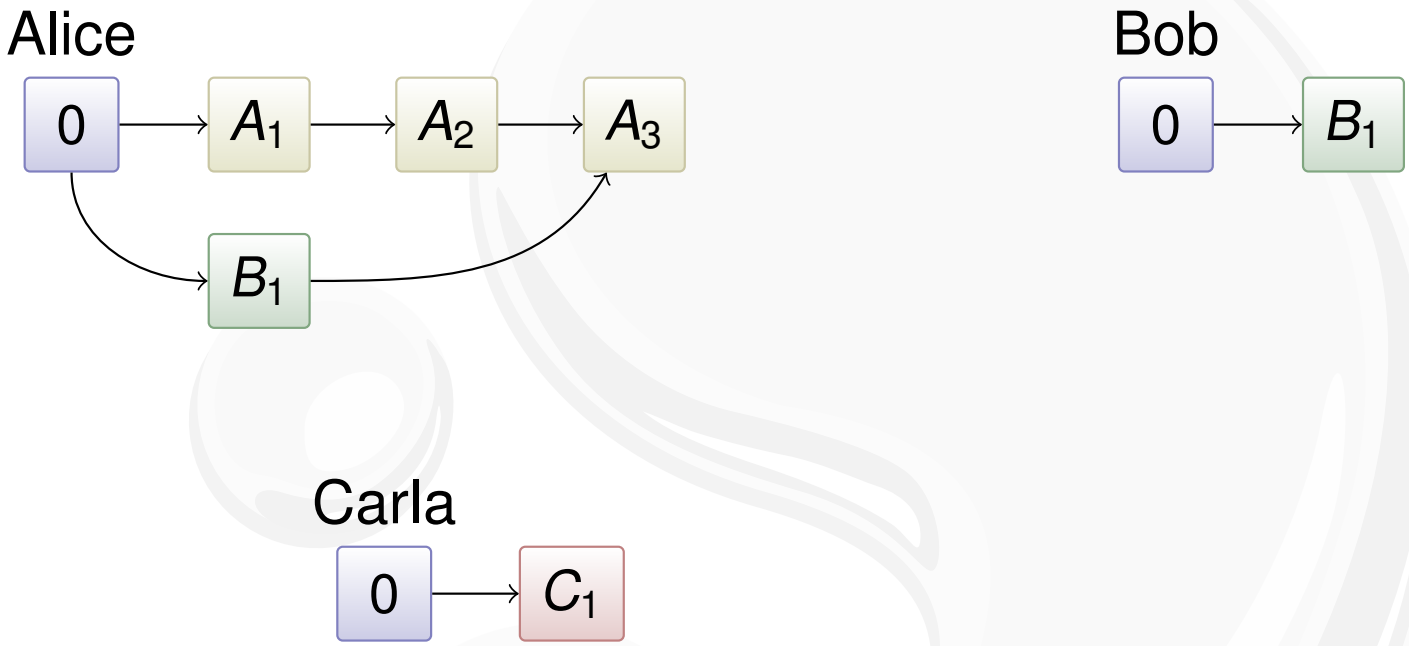
Moving Changesets Around

People have read-only access (e.g., `hg serve`):



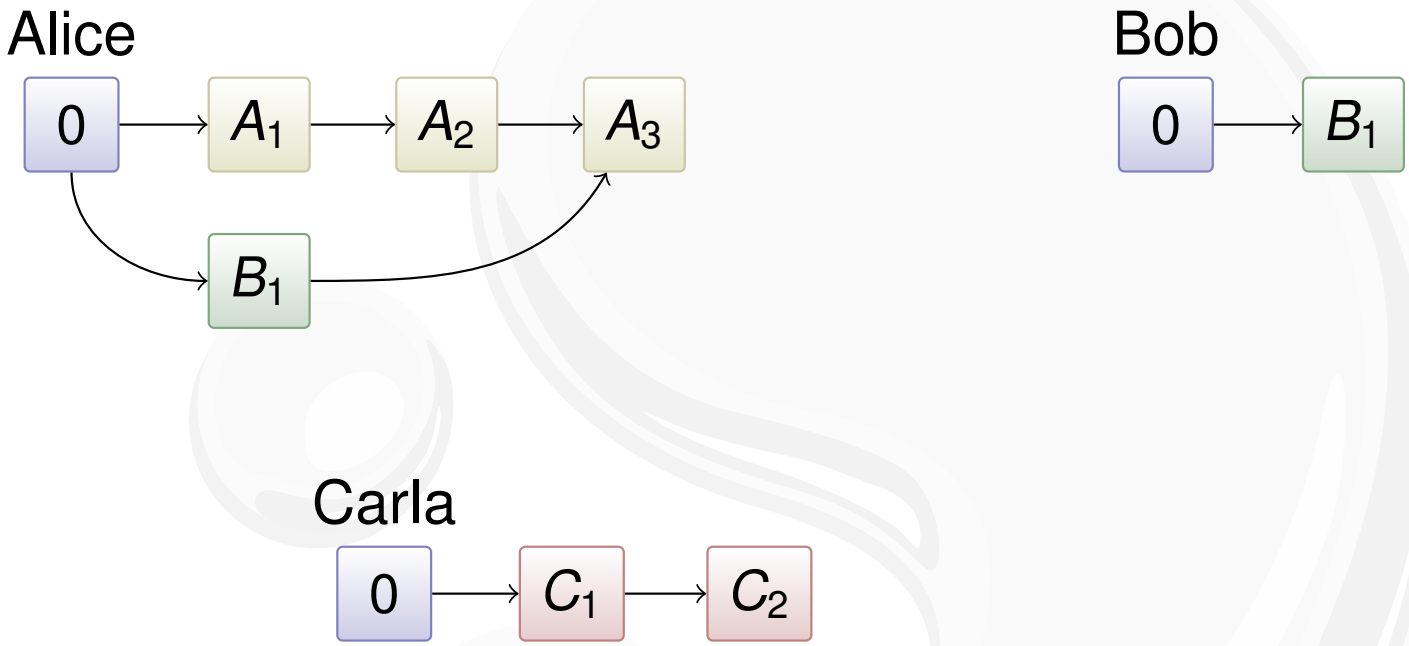
Moving Changesets Around

People have read-only access (e.g., `hg serve`):



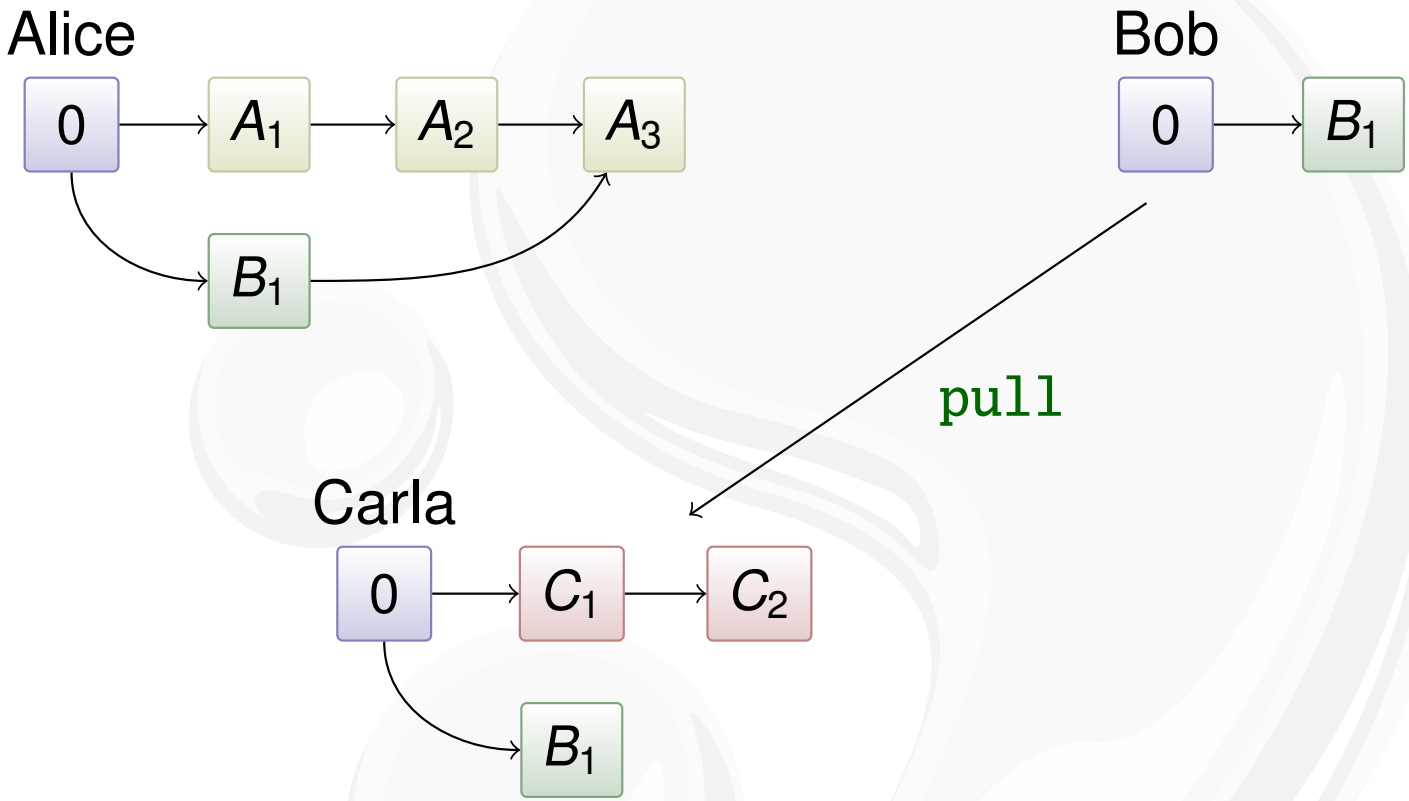
Moving Changesets Around

People have read-only access (e.g., `hg serve`):



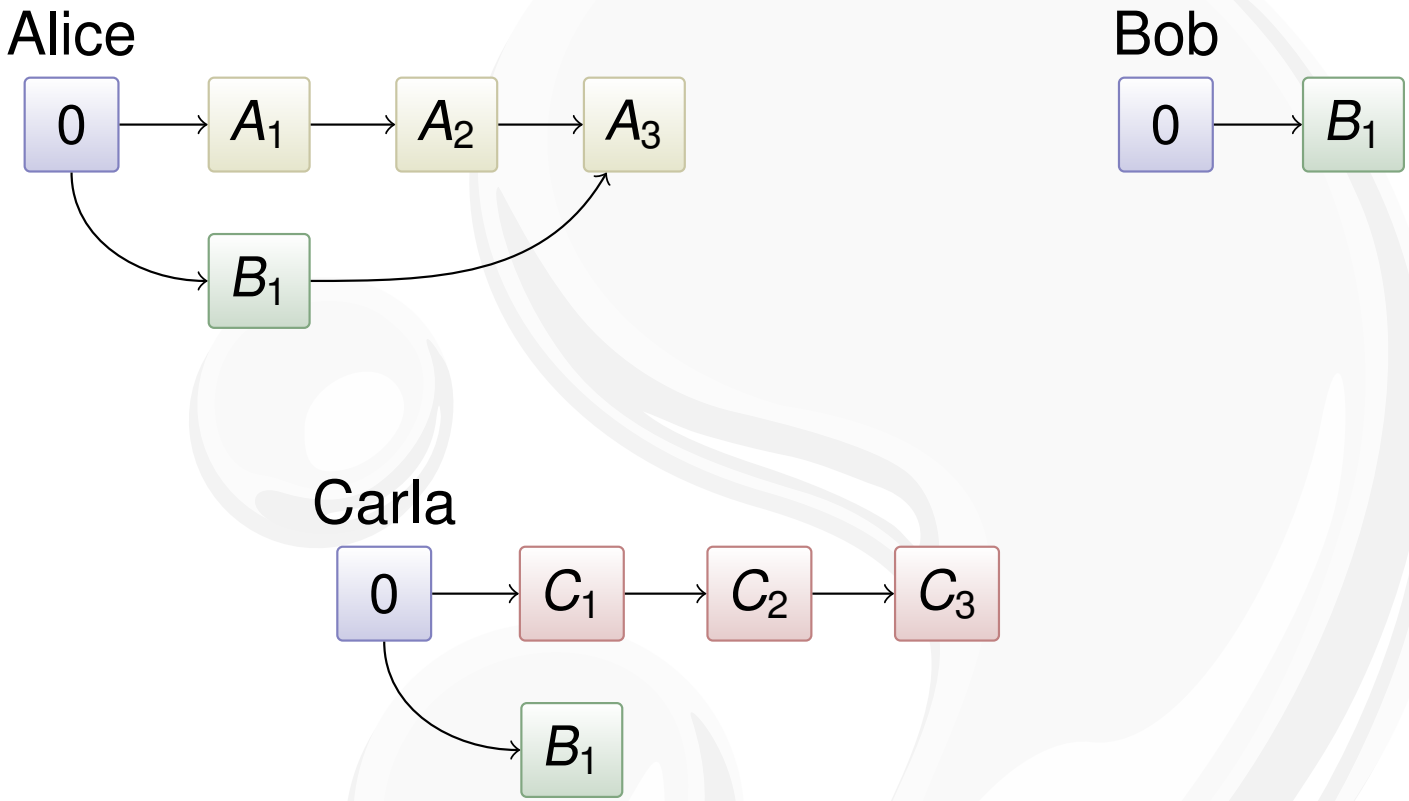
Moving Changesets Around

People have read-only access (e.g., `hg serve`):



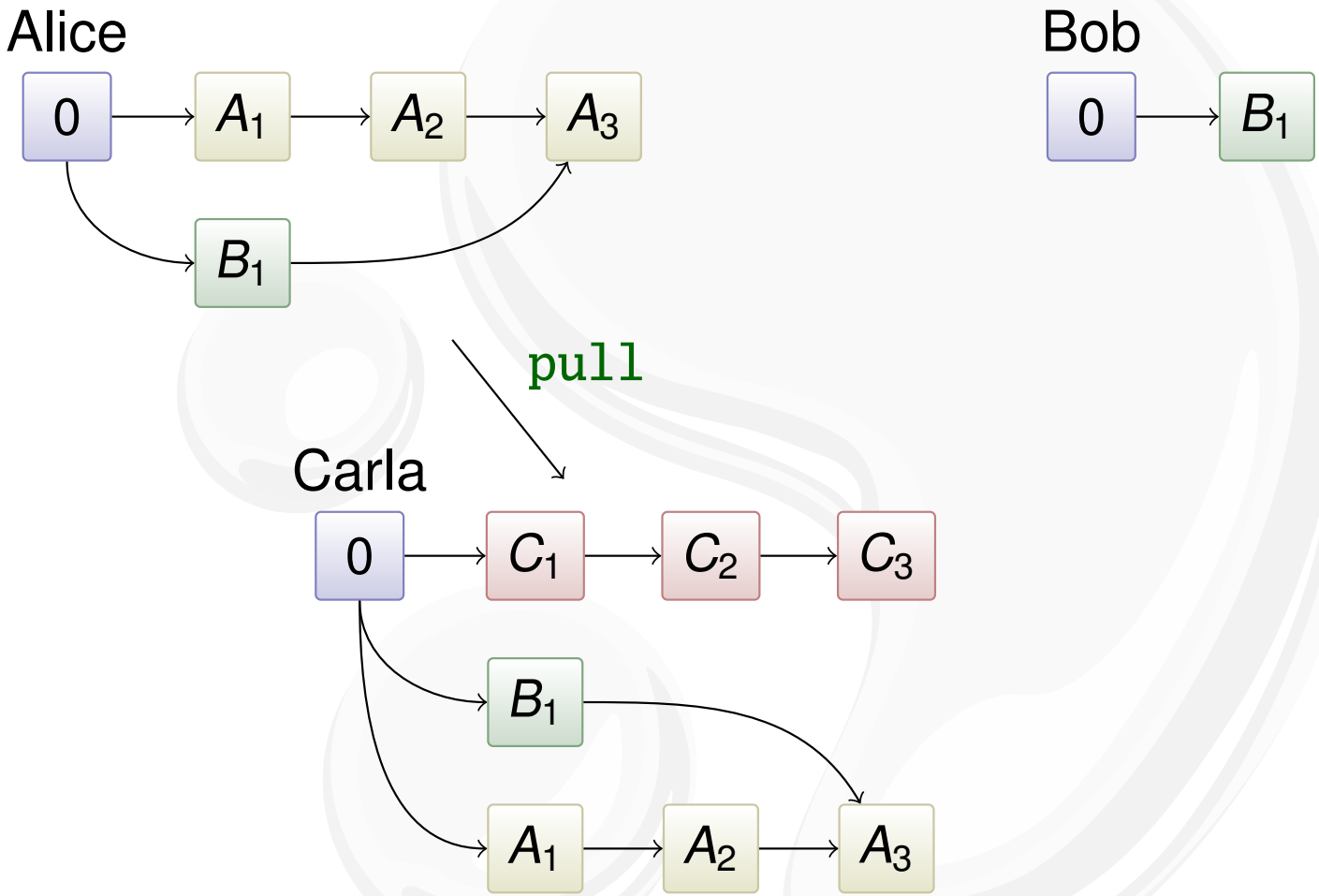
Moving Changesets Around

People have read-only access (e.g., `hg serve`):



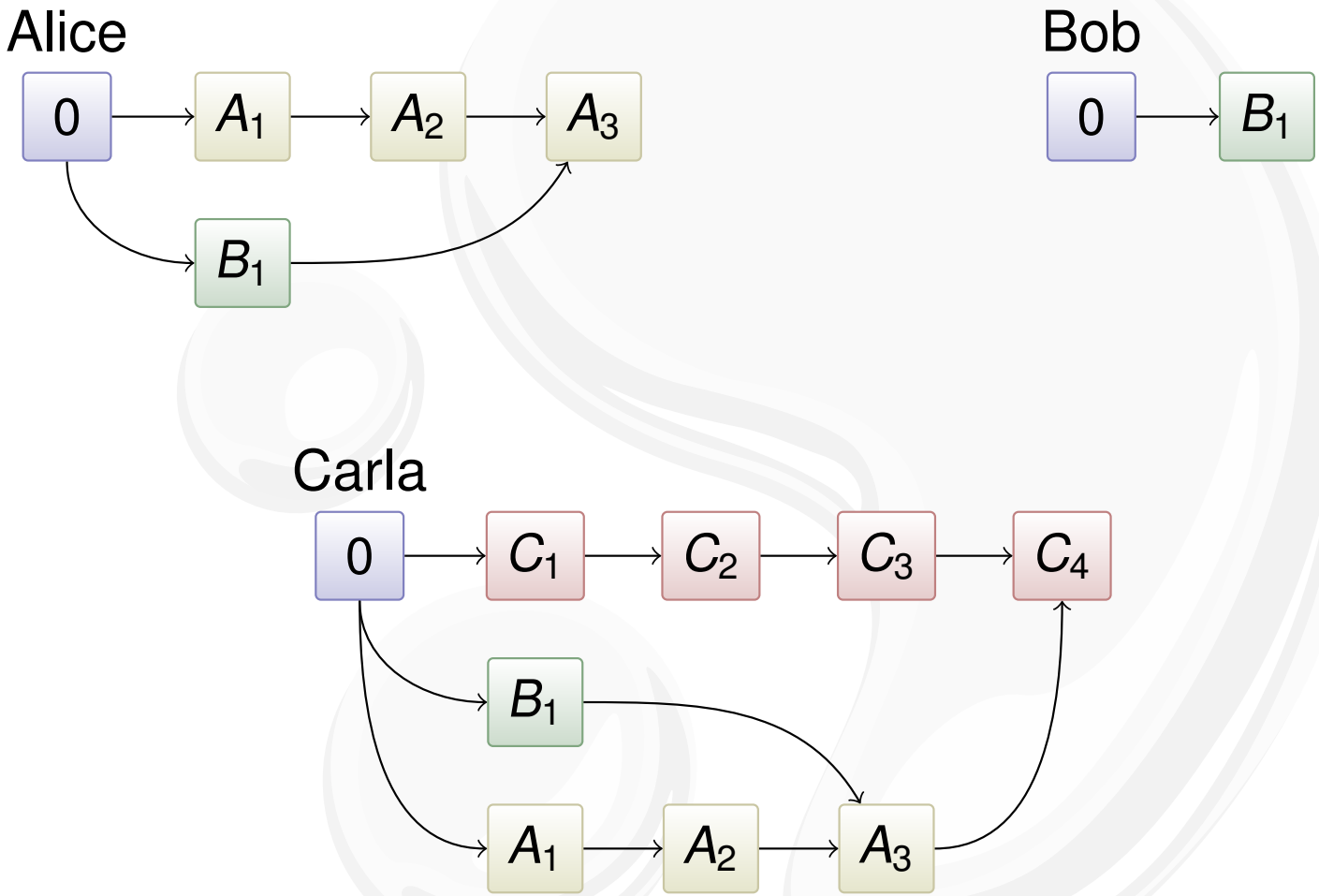
Moving Changesets Around

People have read-only access (e.g., `hg serve`):



Moving Changesets Around

People have read-only access (e.g., `hg serve`):

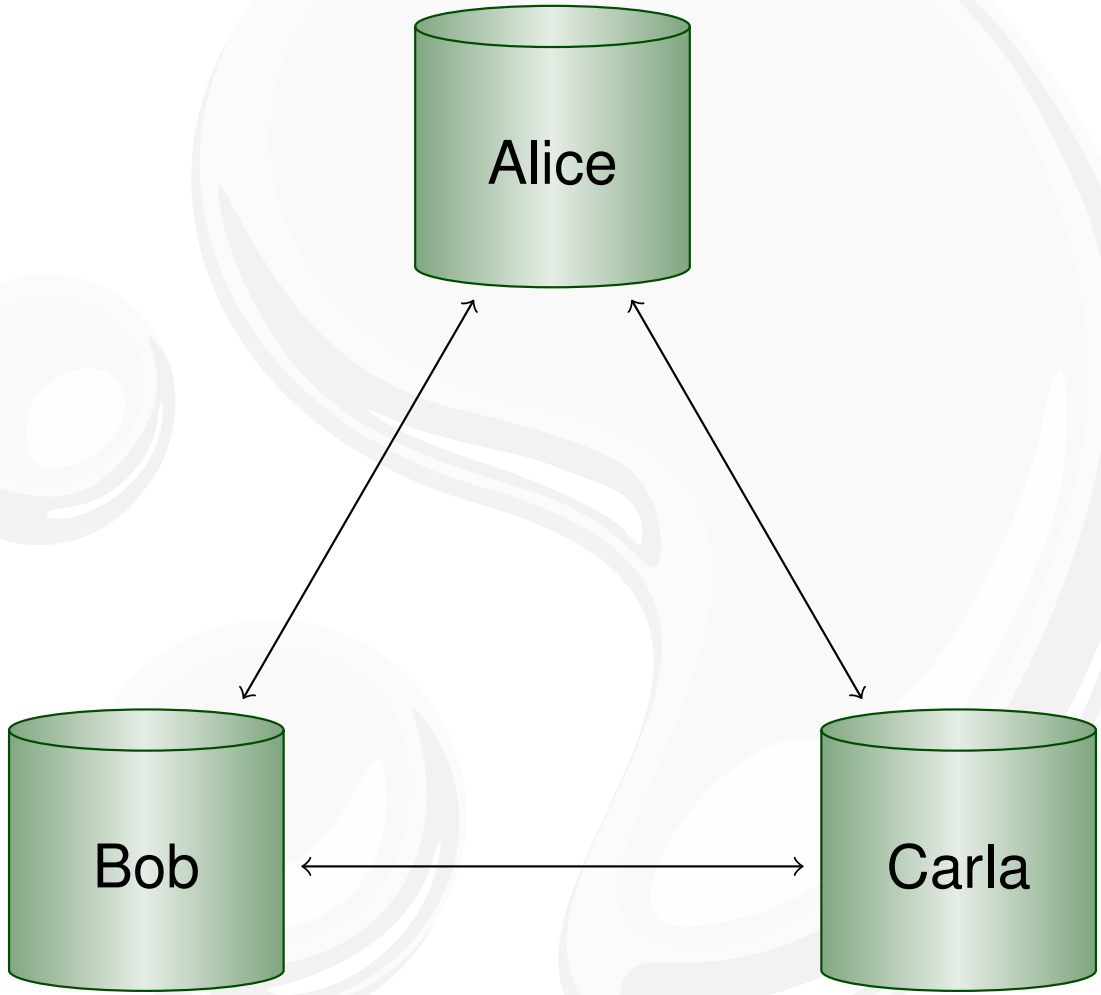


Outline

Outline

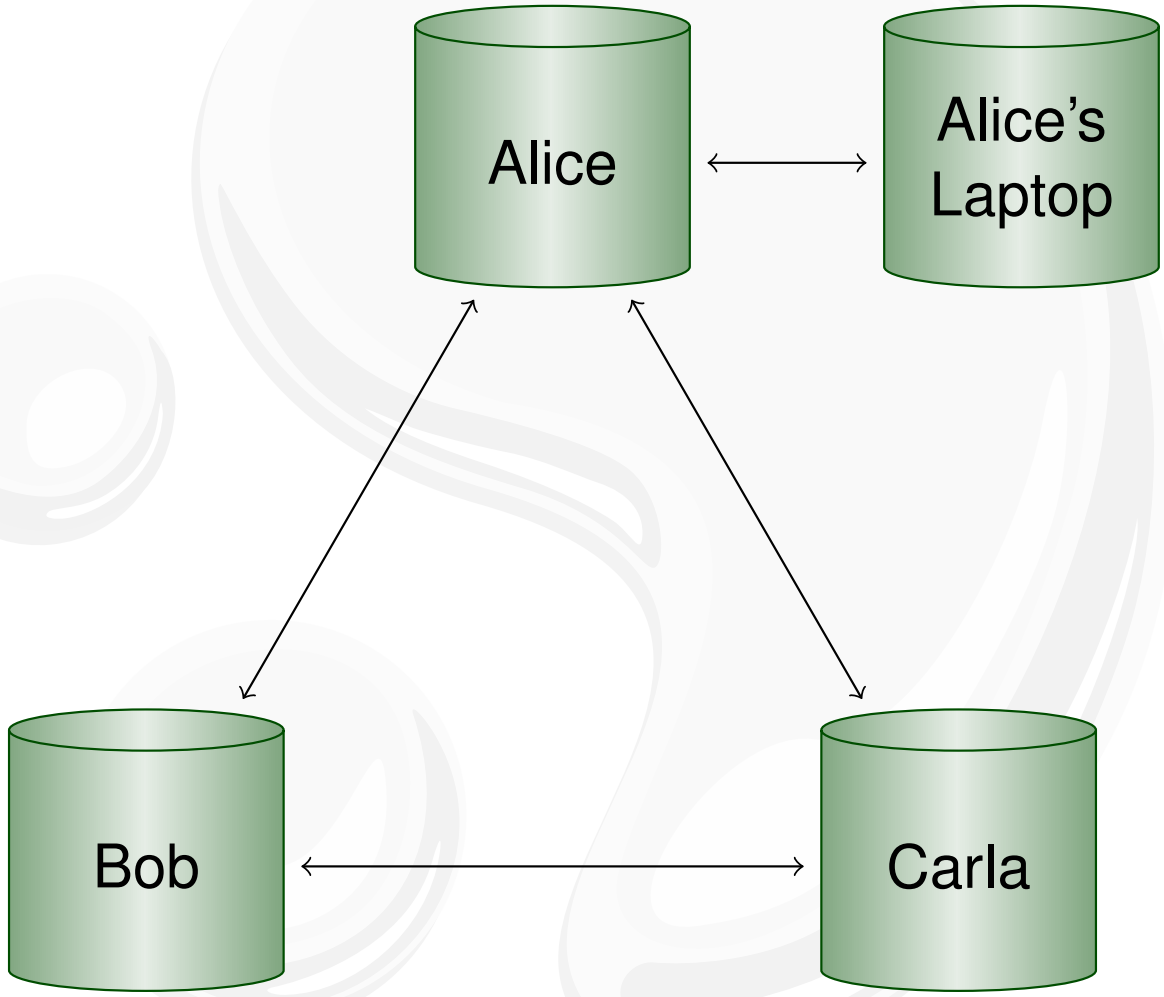
Distributed Revision Control

Mercurial duplicates the history on many servers:



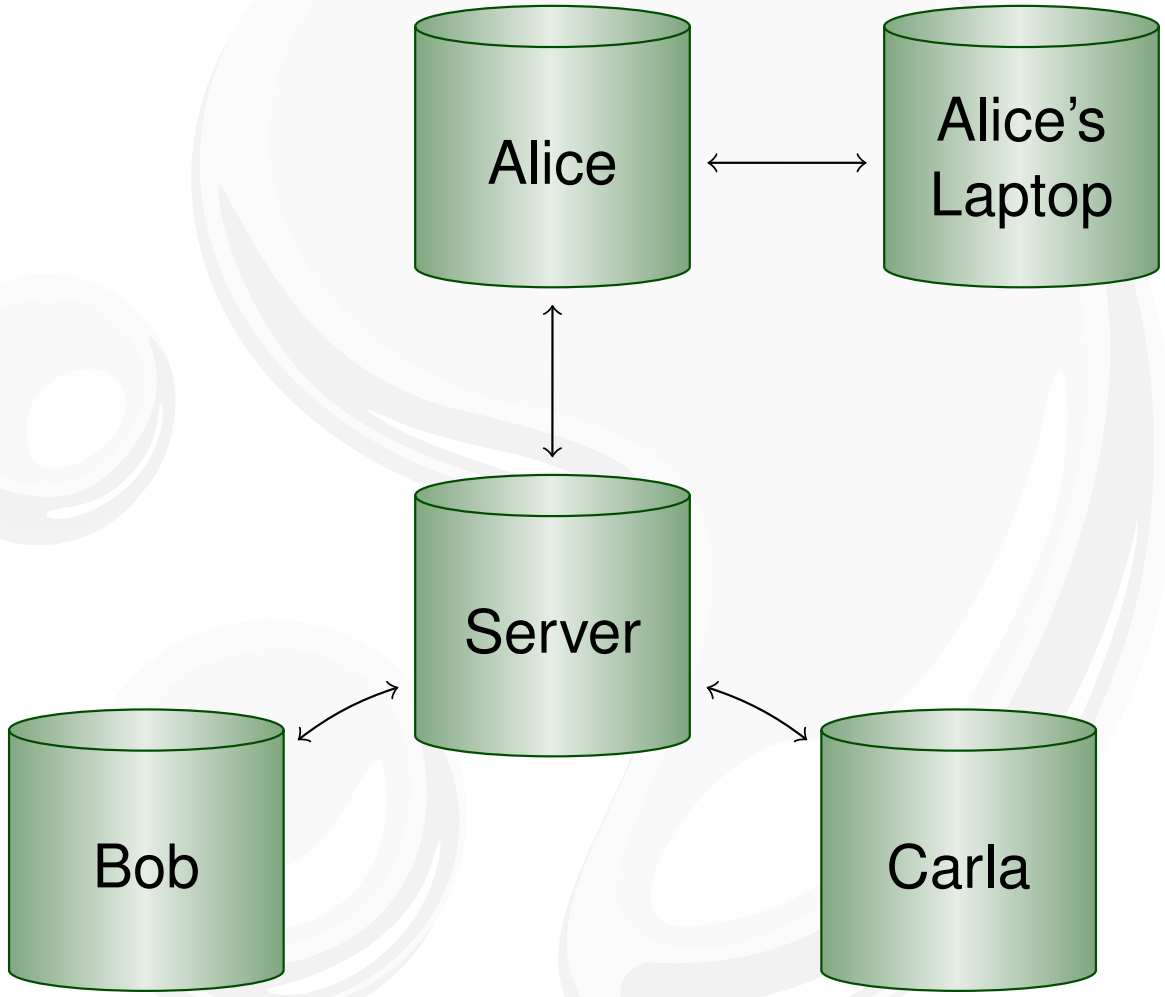
Distributed Revision Control

Mercurial duplicates the history on many servers:



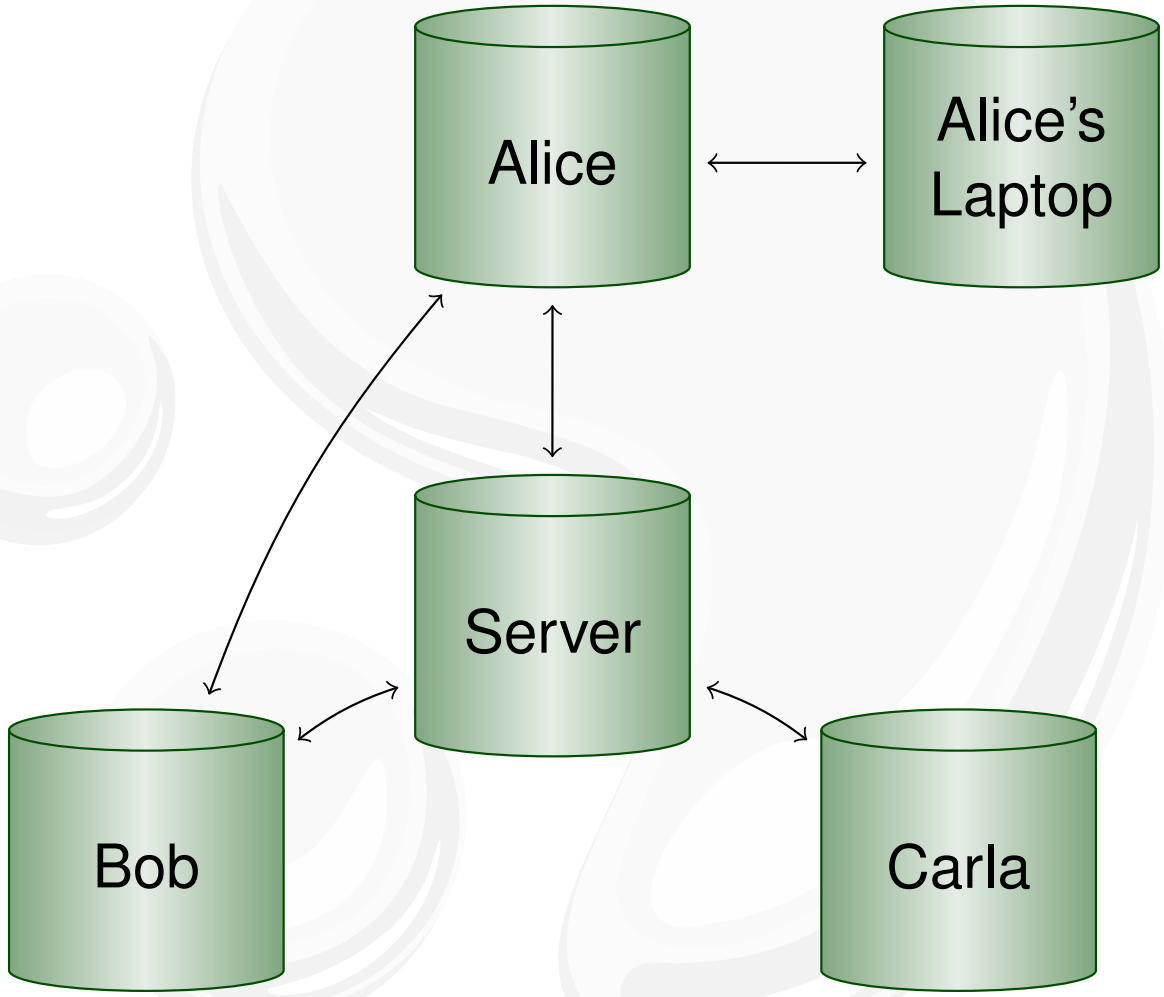
Distributed Revision Control

Mercurial duplicates the history on many servers:



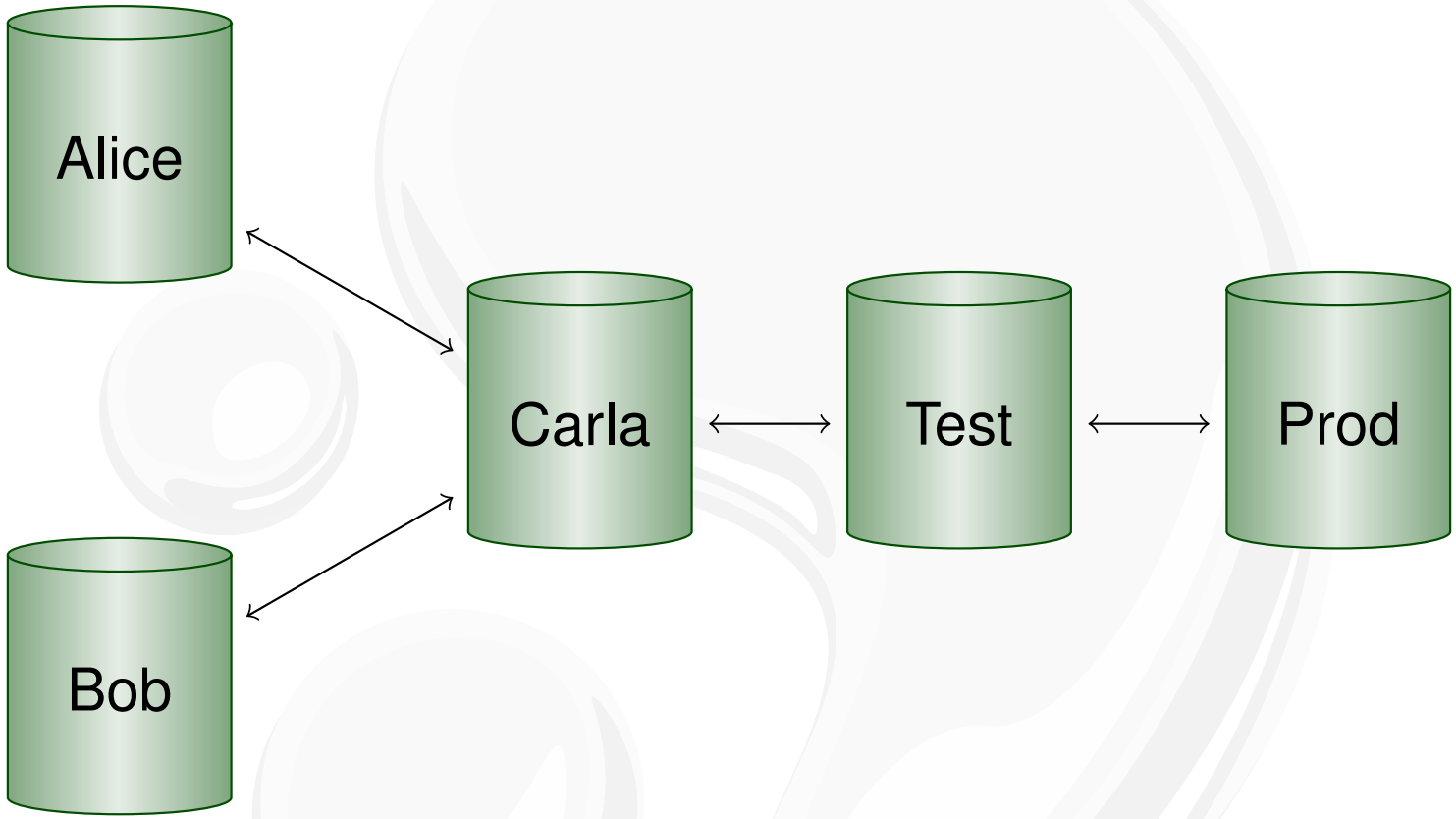
Distributed Revision Control

Mercurial duplicates the history on many servers:



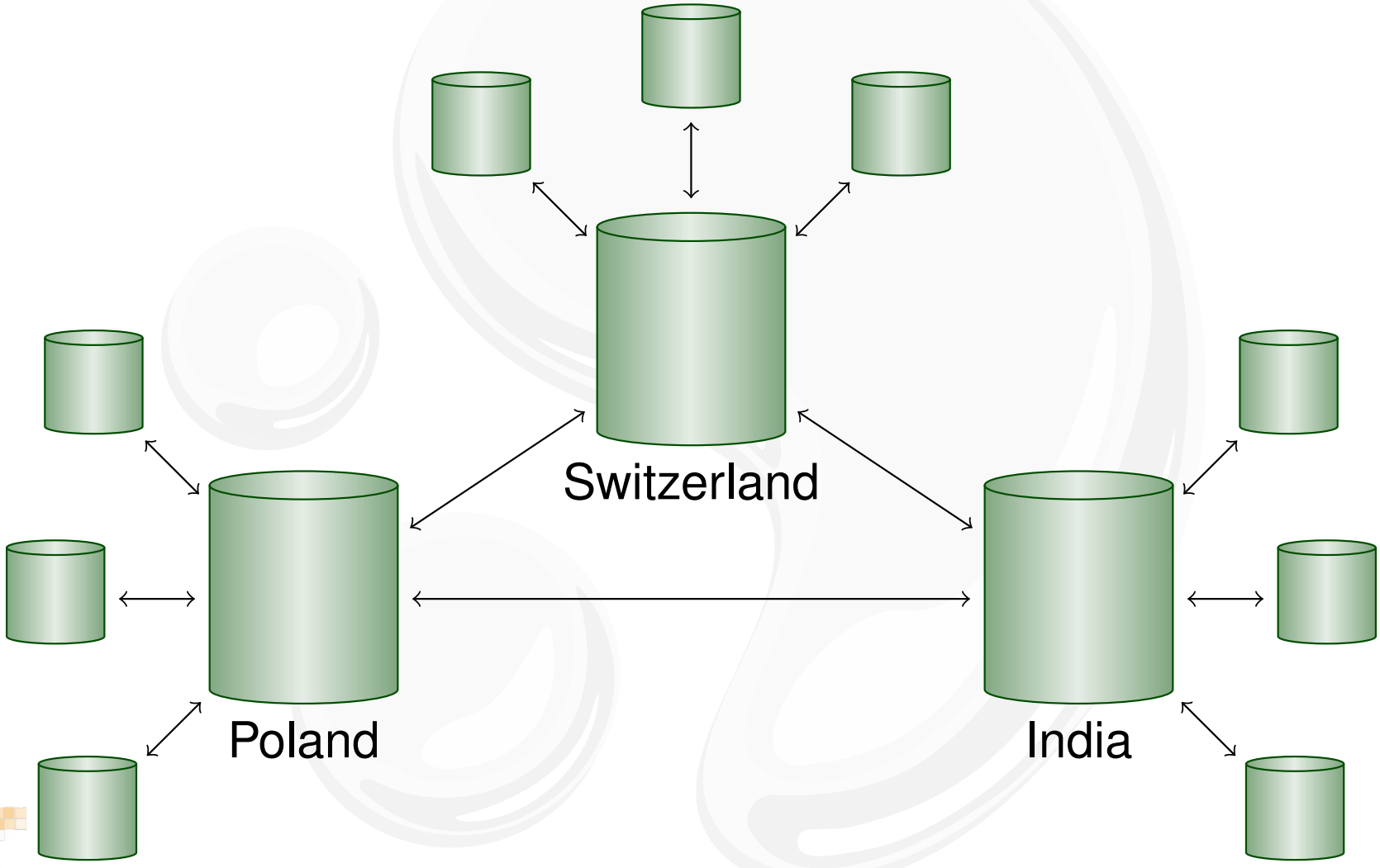
Workflow in a Team

Mercurial scales from a single team. . . :



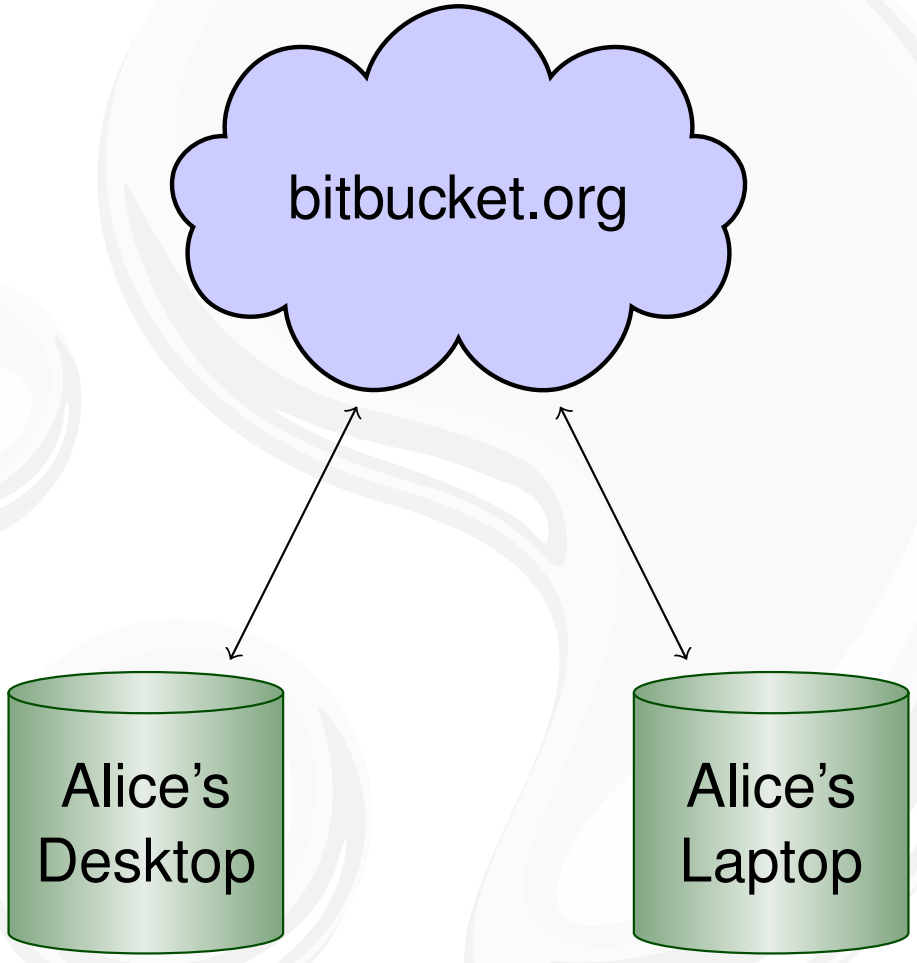
Workflow Between Company Divisions

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



Workflow Between Two Computers

. . . to working with yourself:

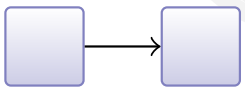


Outline

Branches

A key concept:

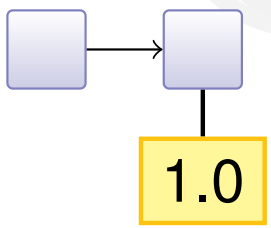
- parallel lines of development
- used to track releases
- used to isolate disruptive changes



Branches

A key concept:

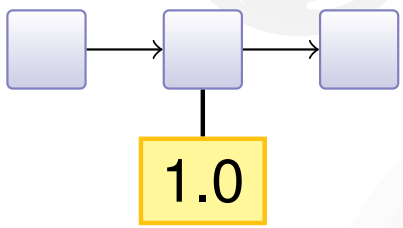
- parallel lines of development
- used to track releases
- used to isolate disruptive changes



Branches

A key concept:

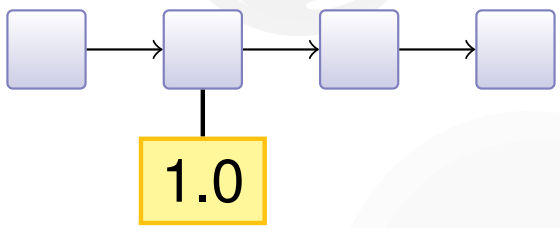
- parallel lines of development
- used to track releases
- used to isolate disruptive changes



Branches

A key concept:

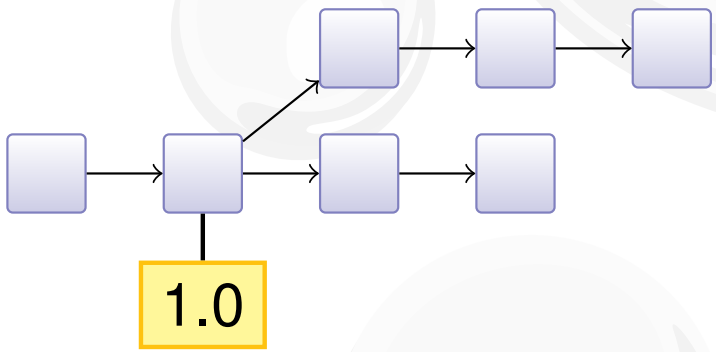
- parallel lines of development
- used to track releases
- used to isolate disruptive changes



Branches

A key concept:

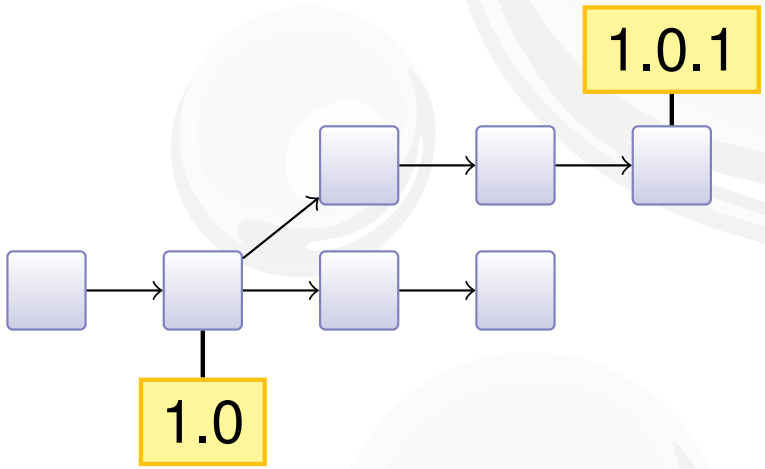
- parallel lines of development
- used to track releases
- used to isolate disruptive changes



Branches

A key concept:

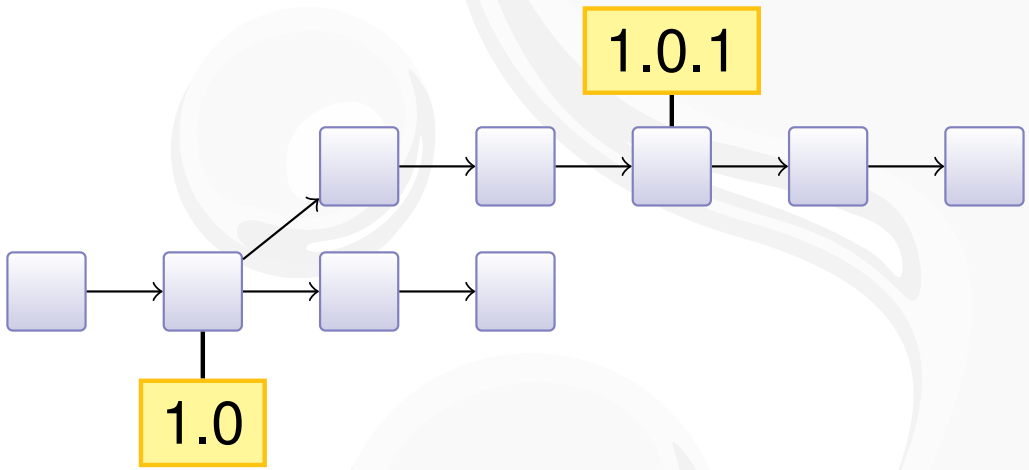
- parallel lines of development
- used to track releases
- used to isolate disruptive changes



Branches

A key concept:

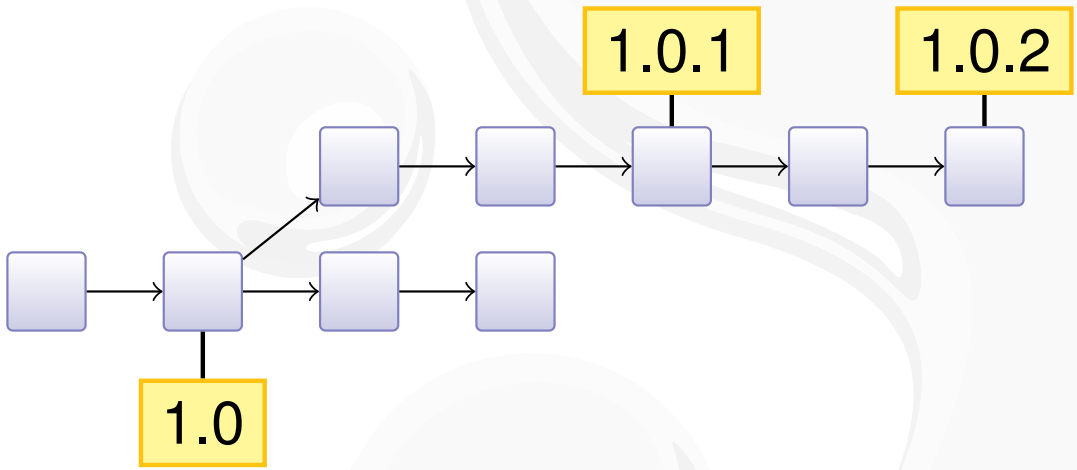
- parallel lines of development
- used to track releases
- used to isolate disruptive changes



Branches

A key concept:

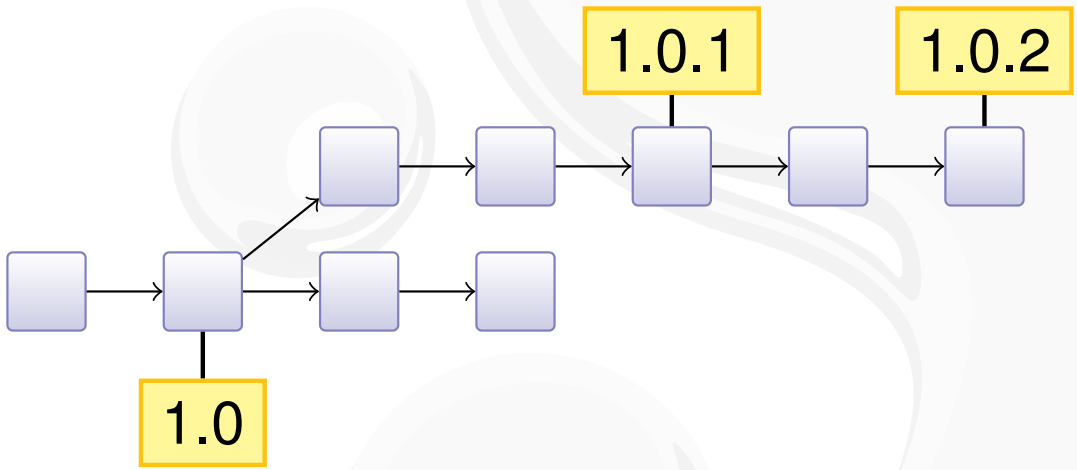
- parallel lines of development
- used to track releases
- used to isolate disruptive changes



Merging

The opposite of branching:

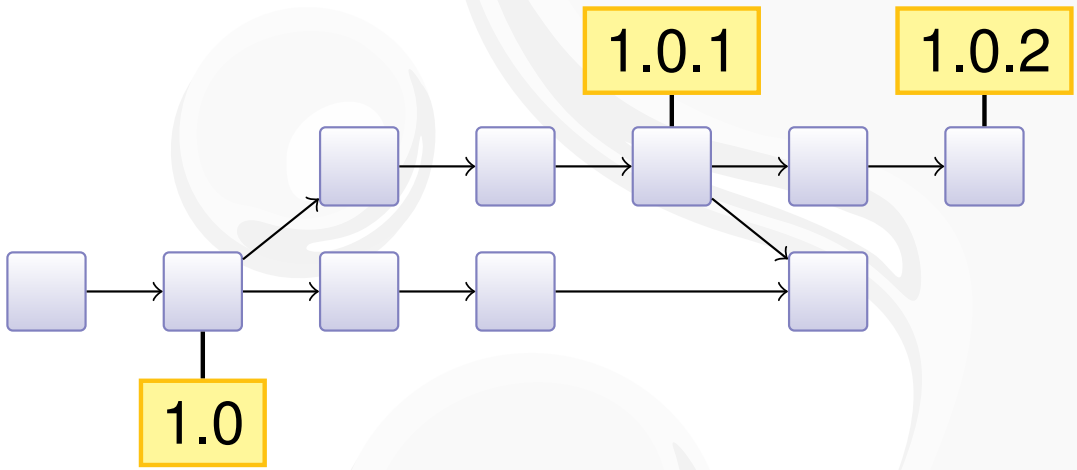
- combines two branches
- used to merge back bugfixes
- used to integrate feature branches



Merging

The opposite of branching:

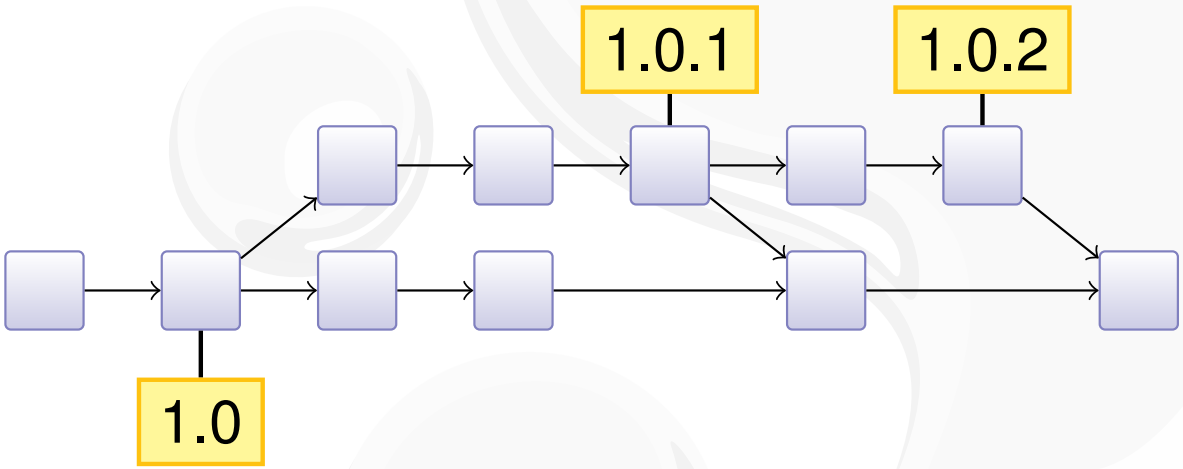
- combines two branches
- used to merge back bugfixes
- used to integrate feature branches



Merging

The opposite of branching:

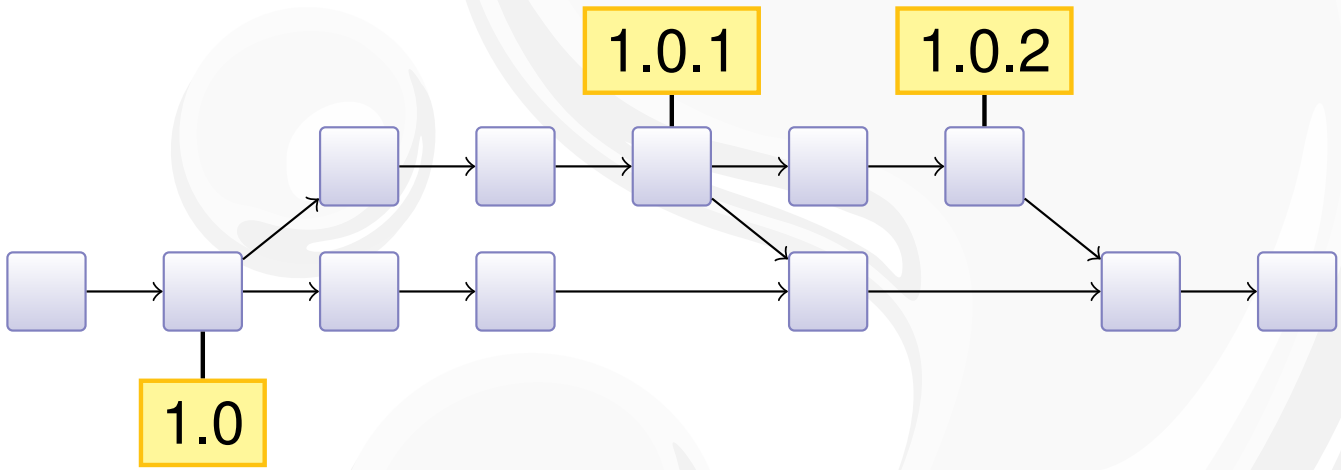
- combines two branches
- used to merge back bugfixes
- used to integrate feature branches



Merging

The opposite of branching:

- combines two branches
- used to merge back bugfixes
- used to integrate feature branches



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

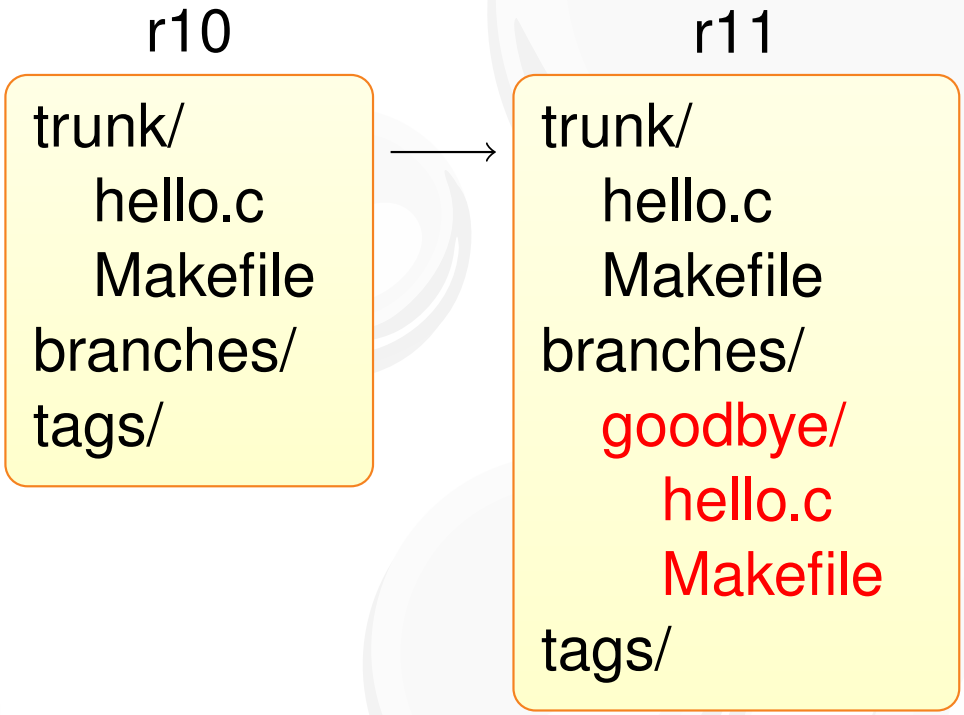
r10

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

Branches in SVN

Subversion knows nothing about branches!

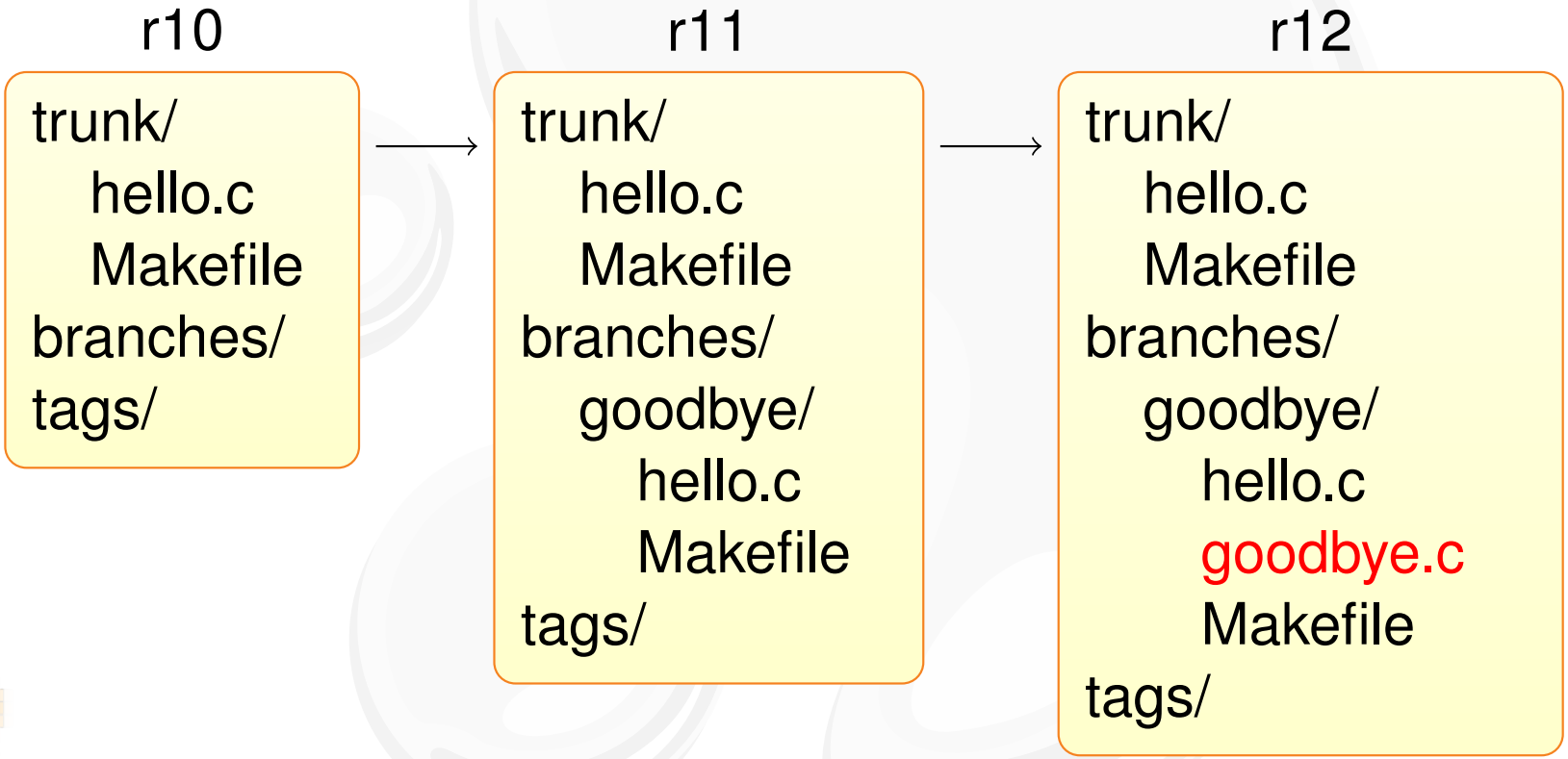
- but SVN has a cheap copy mechanism
- used for tags and branches



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*

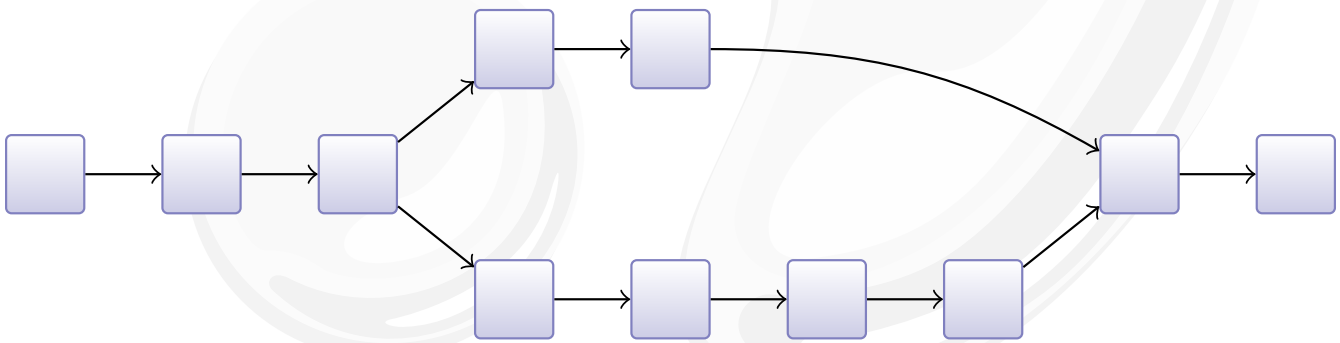
(Mercurial has robust built-in support for merging branches.)

Outline

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



Outline

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?

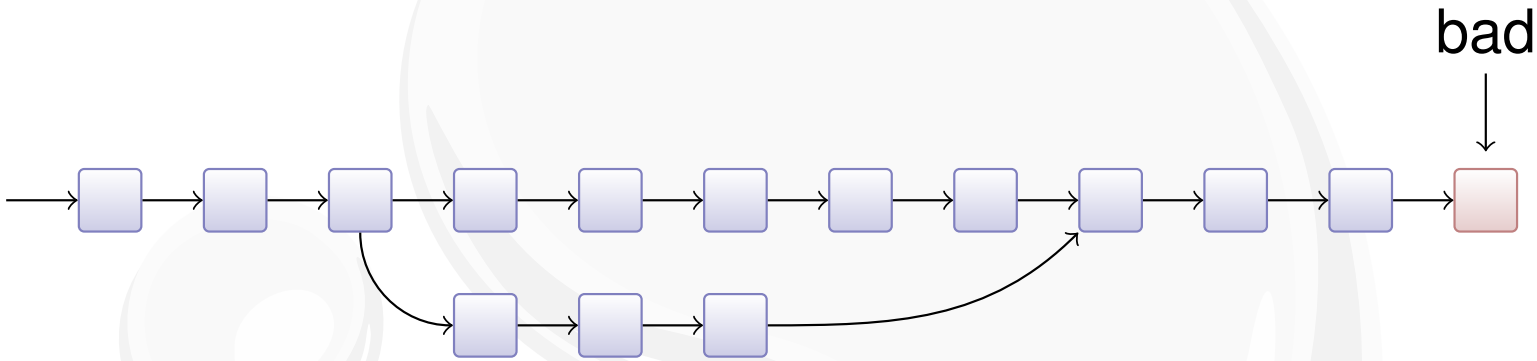
- `hg grep` can help you!

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):
# ...
```

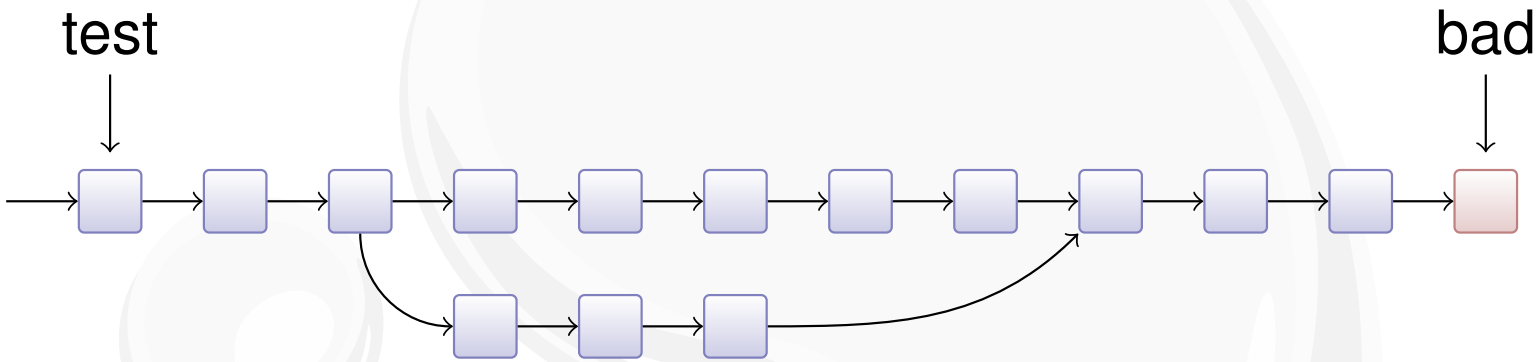
Revision Graph Bisection

You've found a bug! When was it first introduced?
Use `hg bisect` to mark good and bad revisions:



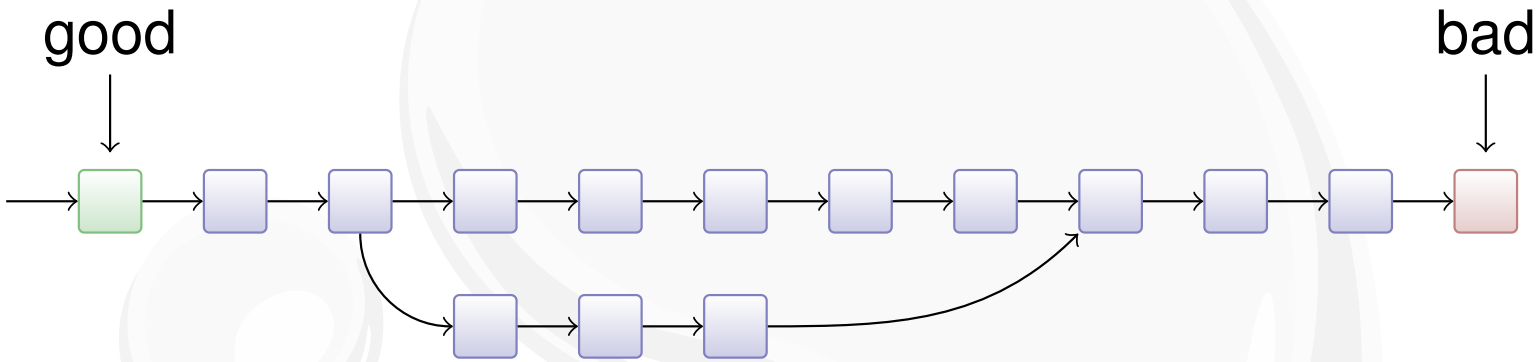
Revision Graph Bisection

You've found a bug! When was it first introduced?
Use `hg bisect` to mark good and bad revisions:



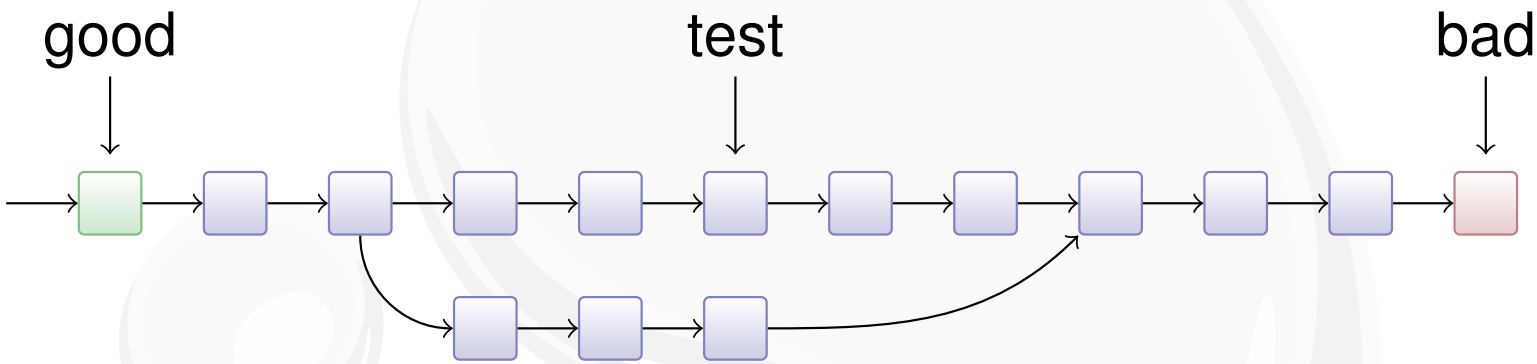
Revision Graph Bisection

You've found a bug! When was it first introduced?
Use `hg bisect` to mark good and bad revisions:



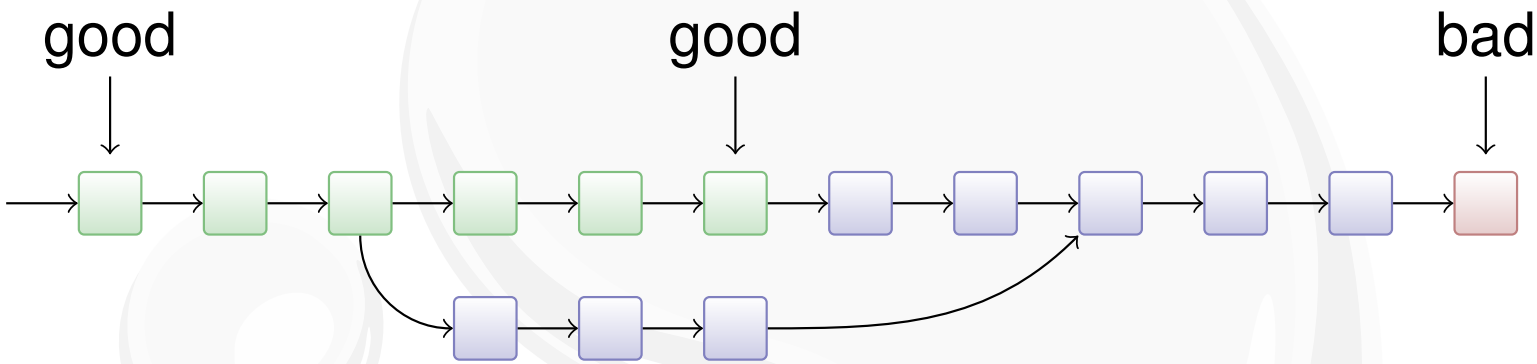
Revision Graph Bisection

You've found a bug! When was it first introduced?
Use `hg bisect` to mark good and bad revisions:



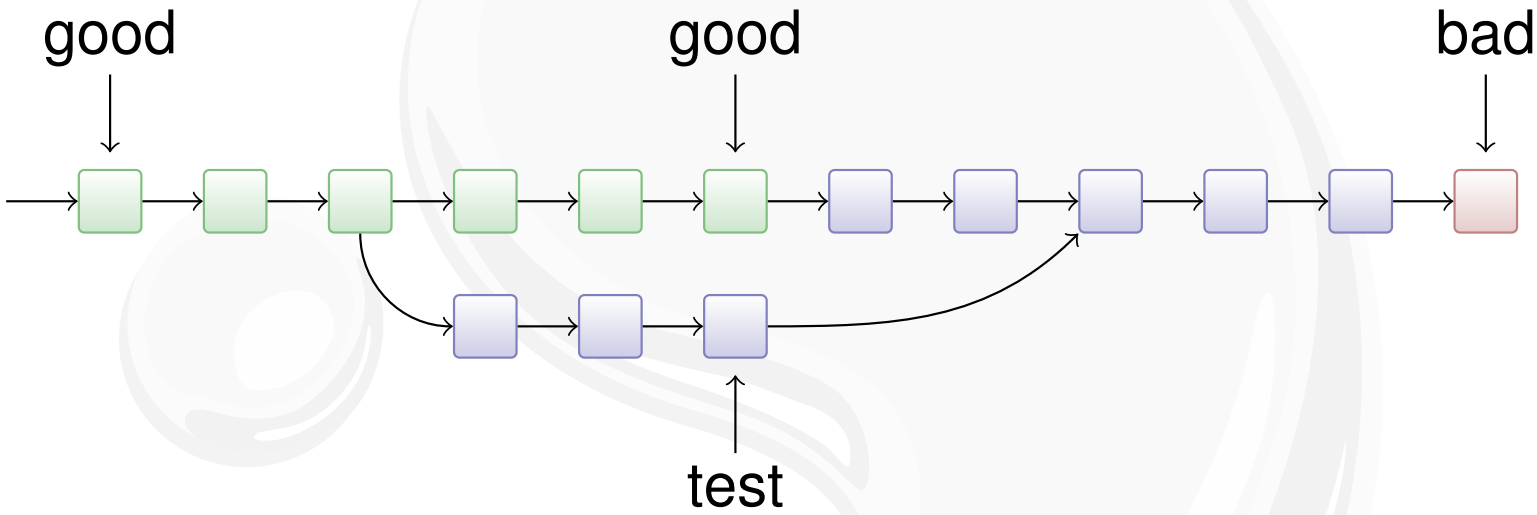
Revision Graph Bisection

You've found a bug! When was it first introduced?
Use `hg bisect` to mark good and bad revisions:



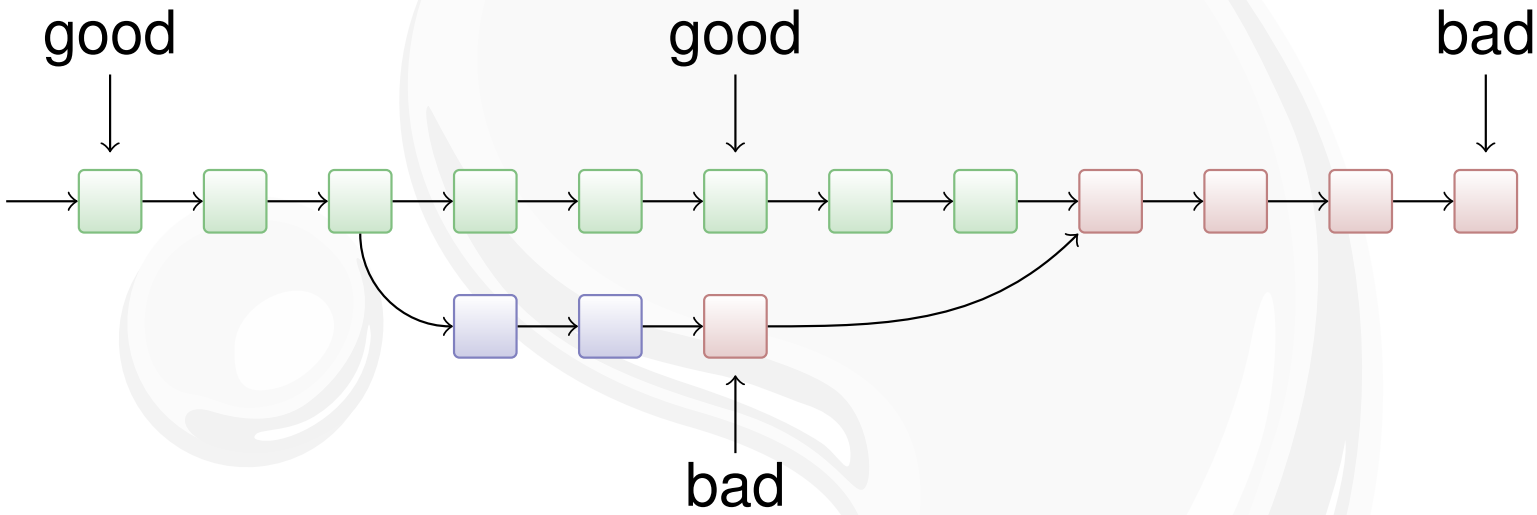
Revision Graph Bisection

You've found a bug! When was it first introduced?
Use `hg bisect` to mark good and bad revisions:



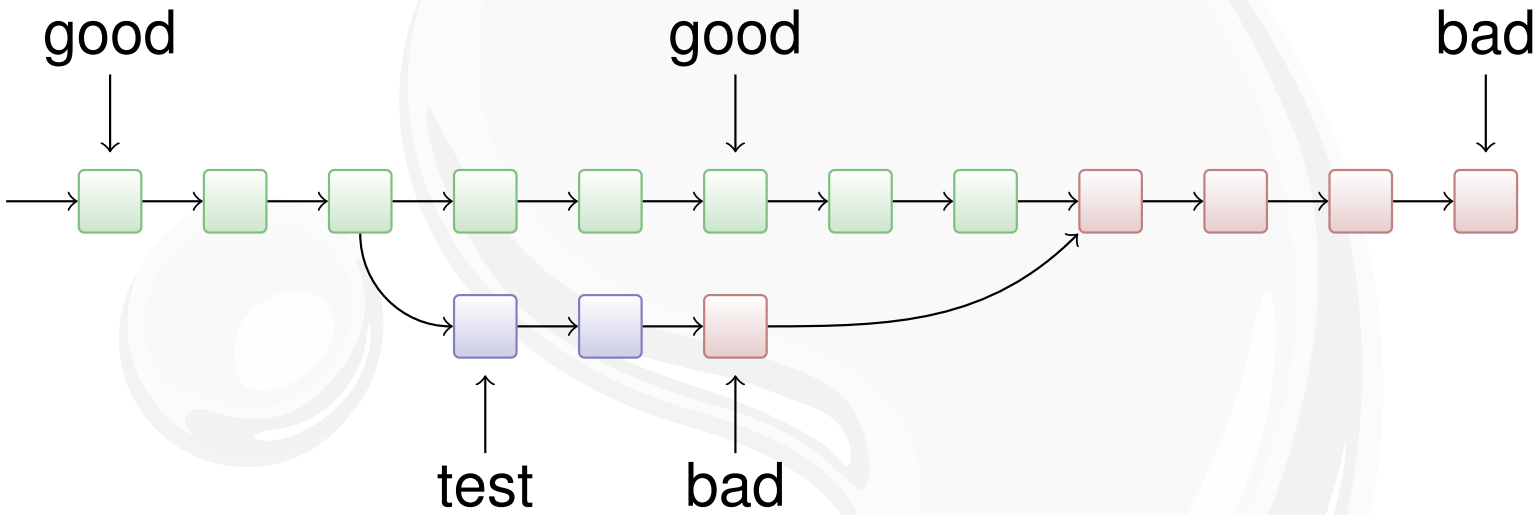
Revision Graph Bisection

You've found a bug! When was it first introduced?
Use `hg bisect` to mark good and bad revisions:



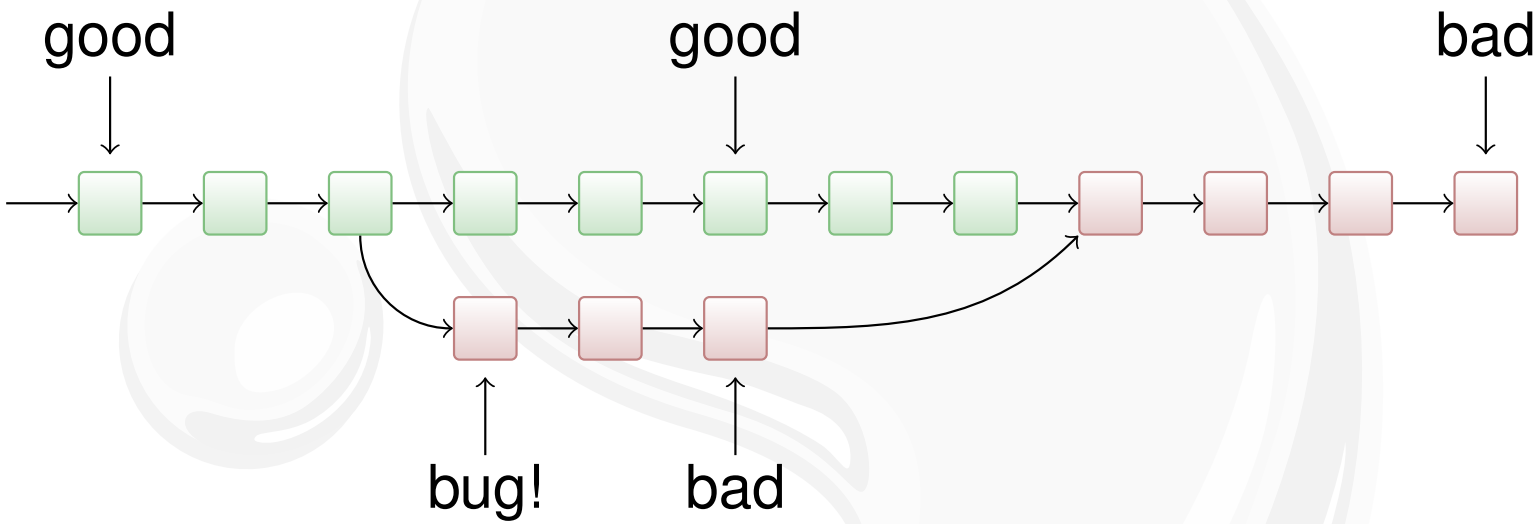
Revision Graph Bisection

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

Mercurial is Extensible

You can add new functionality to Mercurial:

- ships with 30+ extensions
- wiki lists 75+ extensions
- extensions can change basically everything
- helps to keep the core small and focused

Outline

Moving Changesets Around

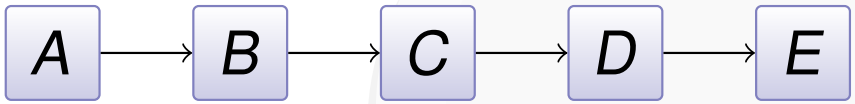
Tired of all those merges? Use the `rebase` extension!

- Revision graph:

Moving Changesets Around

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

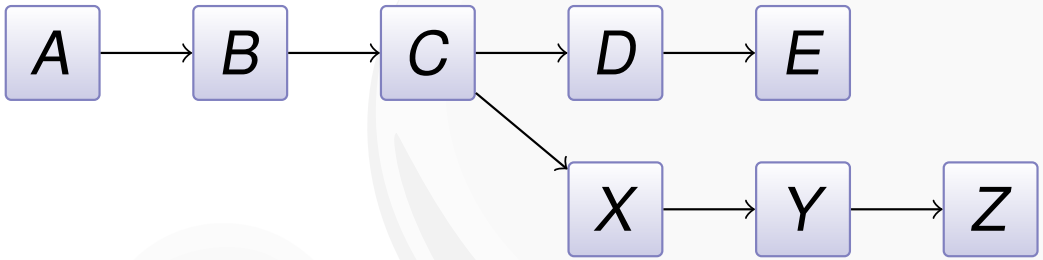
- Revision graph:



Moving Changesets Around

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

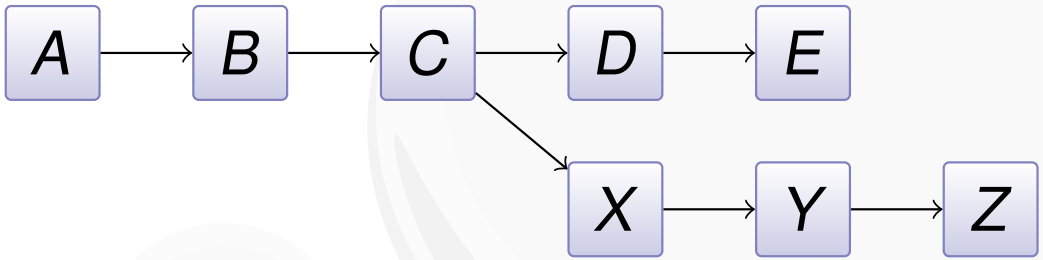
- Revision graph:



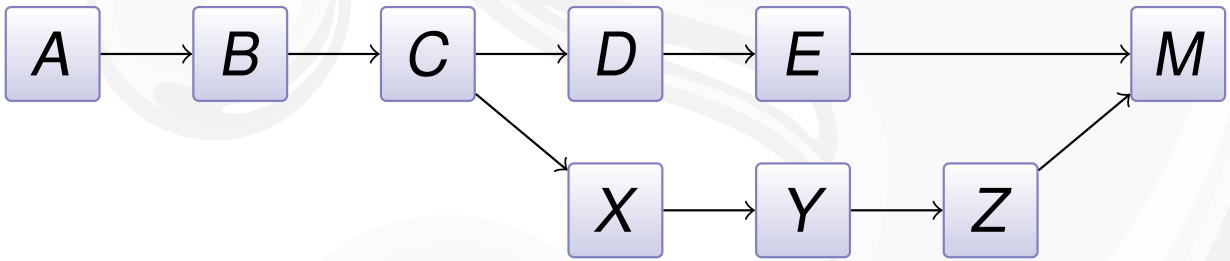
Moving Changesets Around

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

- Revision graph:



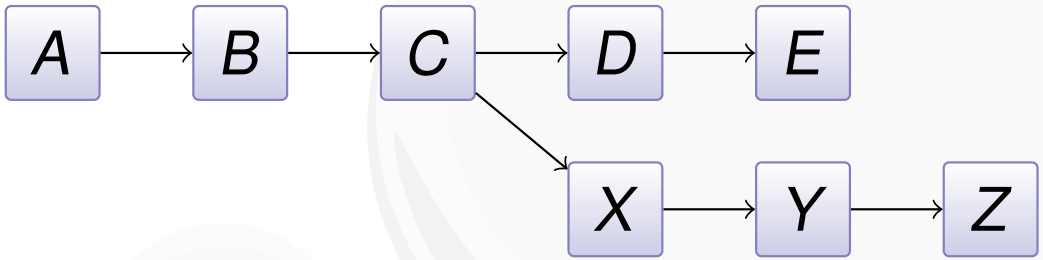
- Merge:



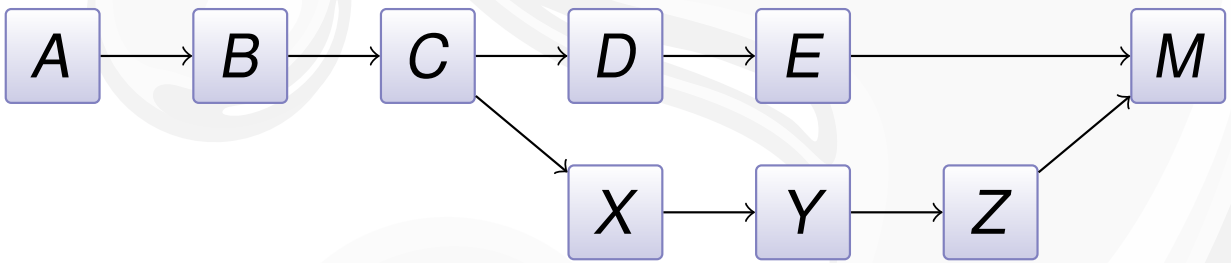
Moving Changesets Around

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

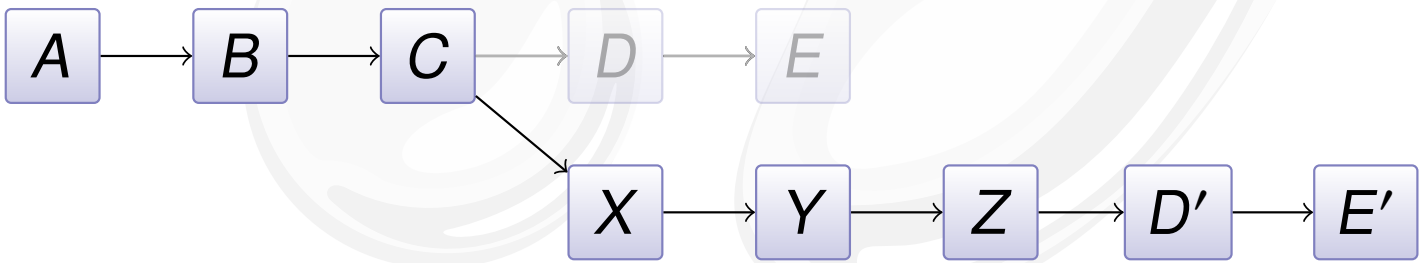
- Revision graph:



- Merge:



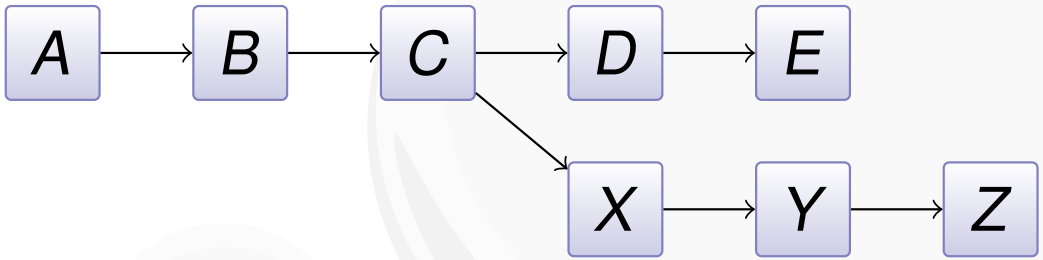
- Rebase:



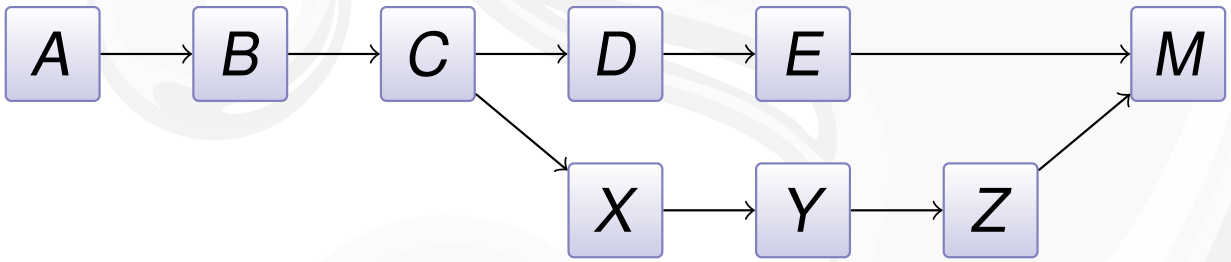
Moving Changesets Around

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

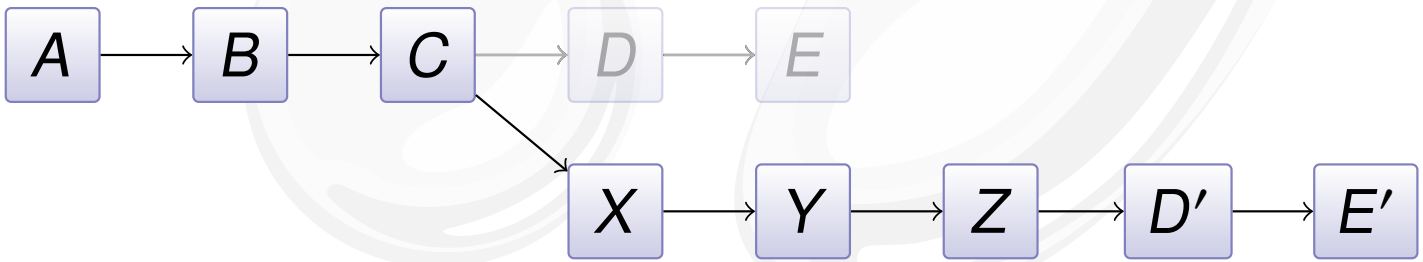
- Revision graph:



- Merge:



- Rebase:



- Beware: public changes should never be rebased.

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

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`!

Outline

Third-Party Tools

Tools with Mercurial support:

- Shell integration: TortoiseHg (Windows, Mac, Linux)
- IDEs: Eclipse, NetBeans, IntelliJ, Visual Studio, Emacs...
- Project Support: Trac, JIRA, Maven, Hudson, BuildBot...

Outline

Live Demo!

Outline

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/ (Microsoft)

Contact

Please get in touch if you have more questions or have considered using Mercurial in your organization:

- Email: `mg@aragost.com`
- IRC: `mg` in `#mercurial` on `irc.freenode.net`

Contact

Please get in touch if you have more questions or have considered using Mercurial in your organization:

- Email: `mg@aragost.com`
- IRC: `mg` in `#mercurial` on `irc.freenode.net`

Thank you!