MHPC
Master in High Performance Computing

ICTP
The Abdus Salam
International Centre
for Theoretical Physics

ma per seguir virtute e canoscenza
SISSA

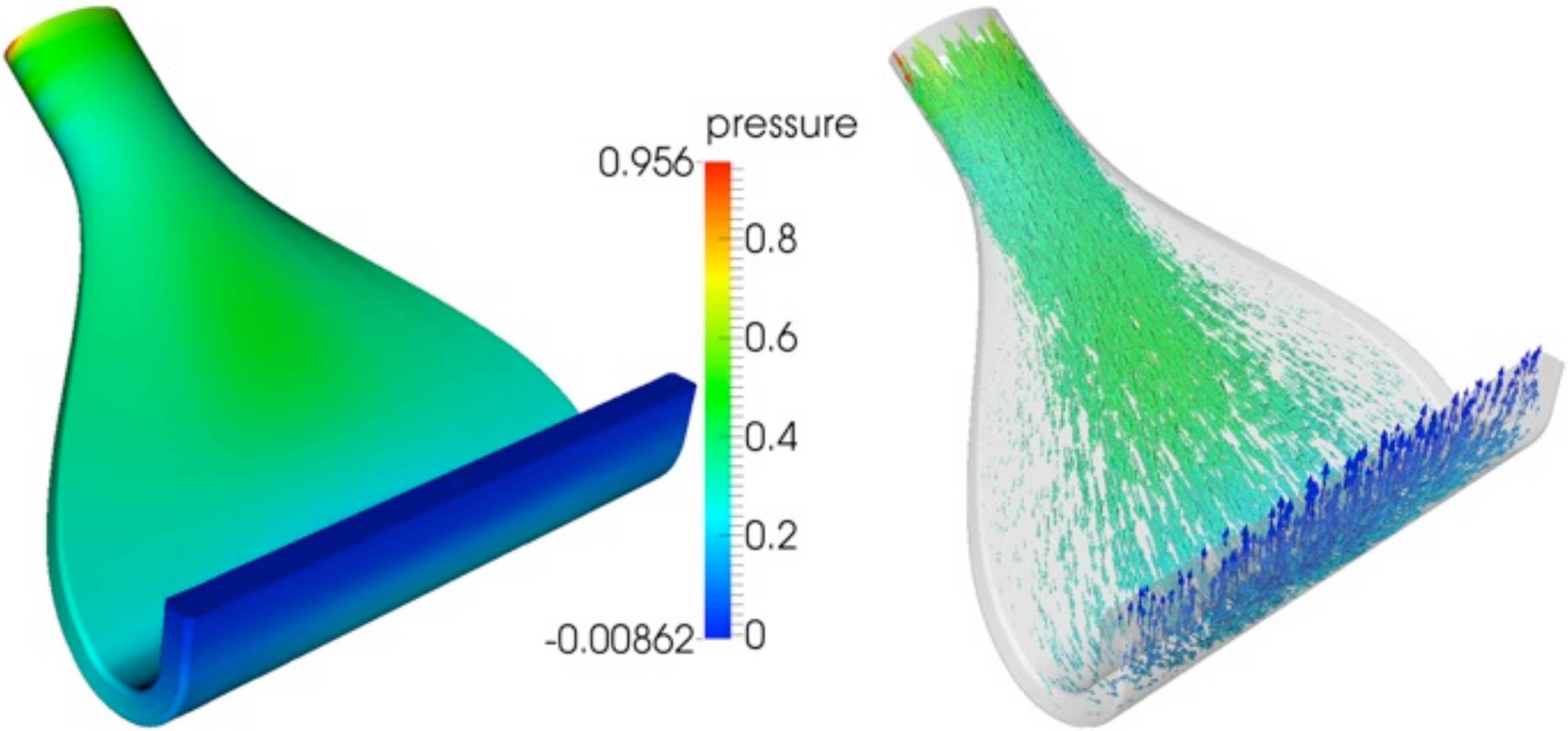
Nicola Cavallini

Basic Git
(adapted from Luca Heltai,
adapted from Patrick Hogan)

ma per seguir virtute e canoscenza
SISSA
International School for Advanced Studies

Fluid Dynamics

- Industrial Test Case



TERRIFIC
Enhancing Interoperability

SEVENTH FRAMEWORK
PROGRAMME

$$N_{ij} = \int_\Omega (\nabla \mathbf{u}_j)\,\mathbf{u} \cdot \mathbf{v}_i =$$

$$\int_\Omega \begin{pmatrix} \partial_x u_x & \partial_y u_x \\ \partial_x u_y & \partial_y u_y \end{pmatrix}_j \begin{pmatrix} u_x \\ u_y \end{pmatrix} \cdot \begin{pmatrix} v_x \\ v_y \end{pmatrix}_i$$

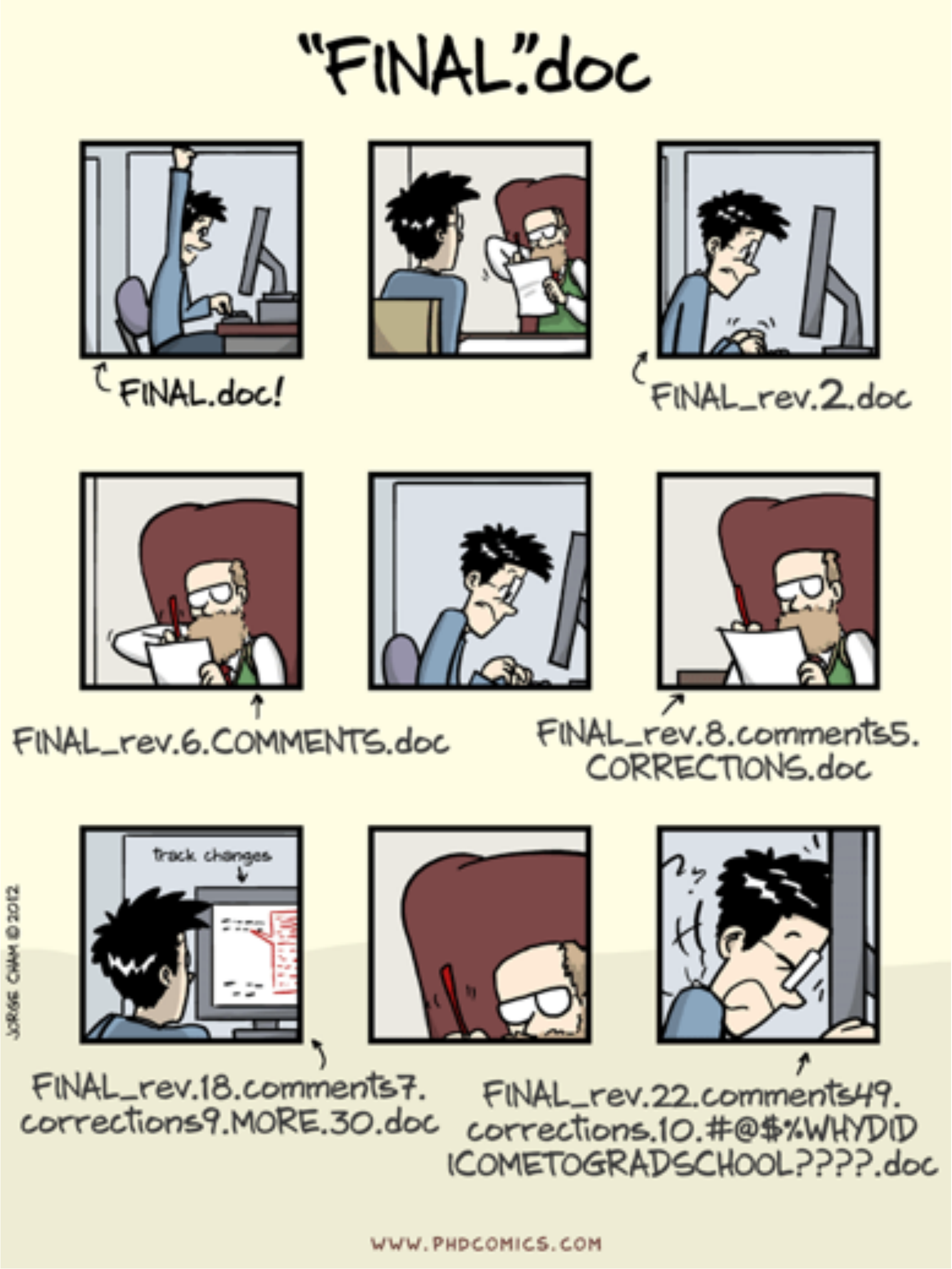
```
for (dof_index q = 0; q < quad.get_num_points(); q++)
{
   Tensor<dim_phys, rank, contravariant, double> vel_q;

   for (dof_index i = 0; i <local_ndofs; i ++){
      auto phi = element->get_value(i,q);
      vel_q = vel_q+vel[local_dofs[i]]*phi;}

   for (dof_index i = 0; i <local_ndofs; i ++){
      for (dof_index j = 0; j <local_ndofs; j ++){
         adv_ij = scalar_product(
                       action(element->get_gradient(j,q),vel_q),
                                 element->get_value(i,q) )*
                                 element->get_w_measures()[q];]}}
}
```

Listing 1: Advection assemble.

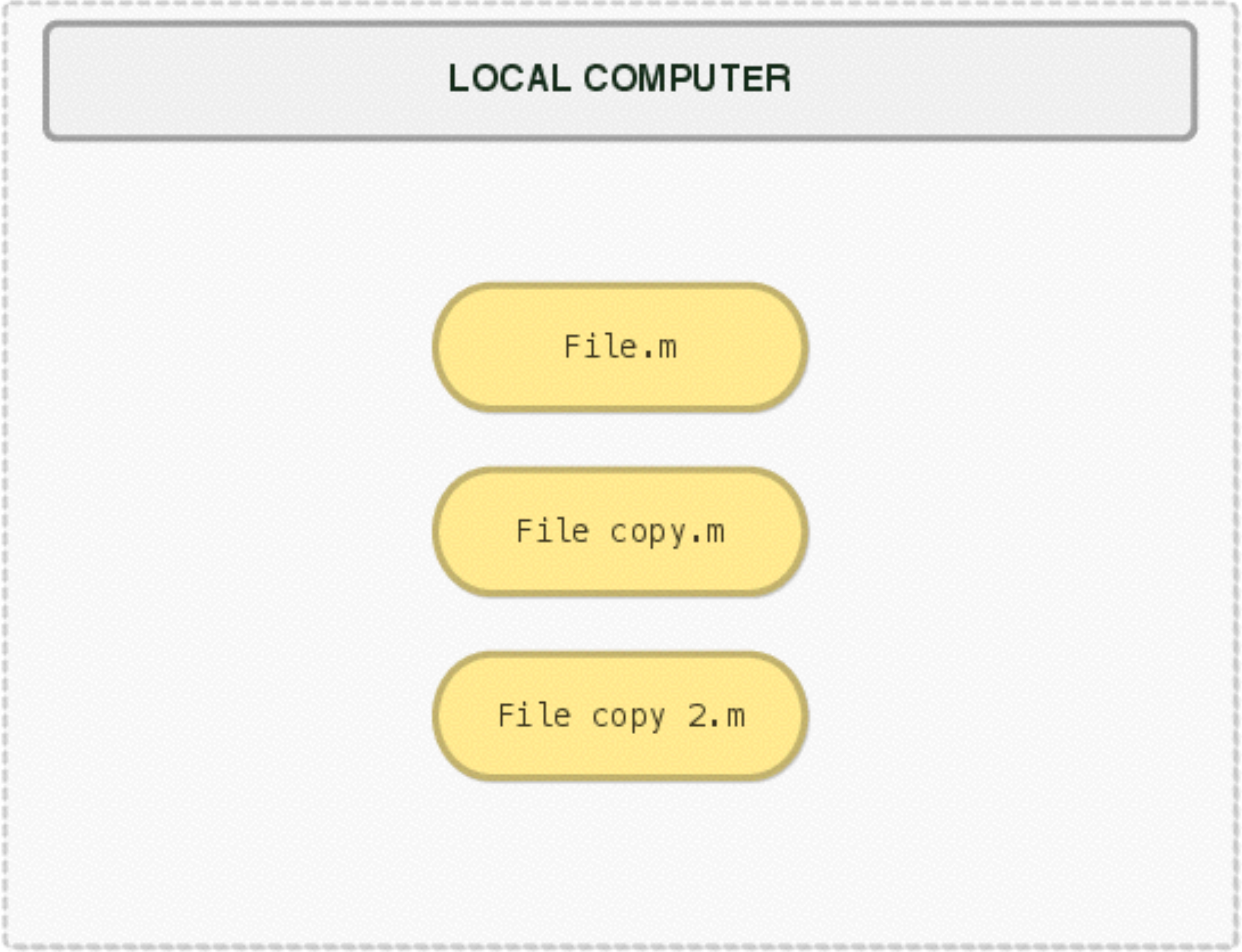
TERRIFIC
Enhancing Interoperability

SEVENTH FRAMEWORK PROGRAMME

Why Version Control?



Local Filesystem



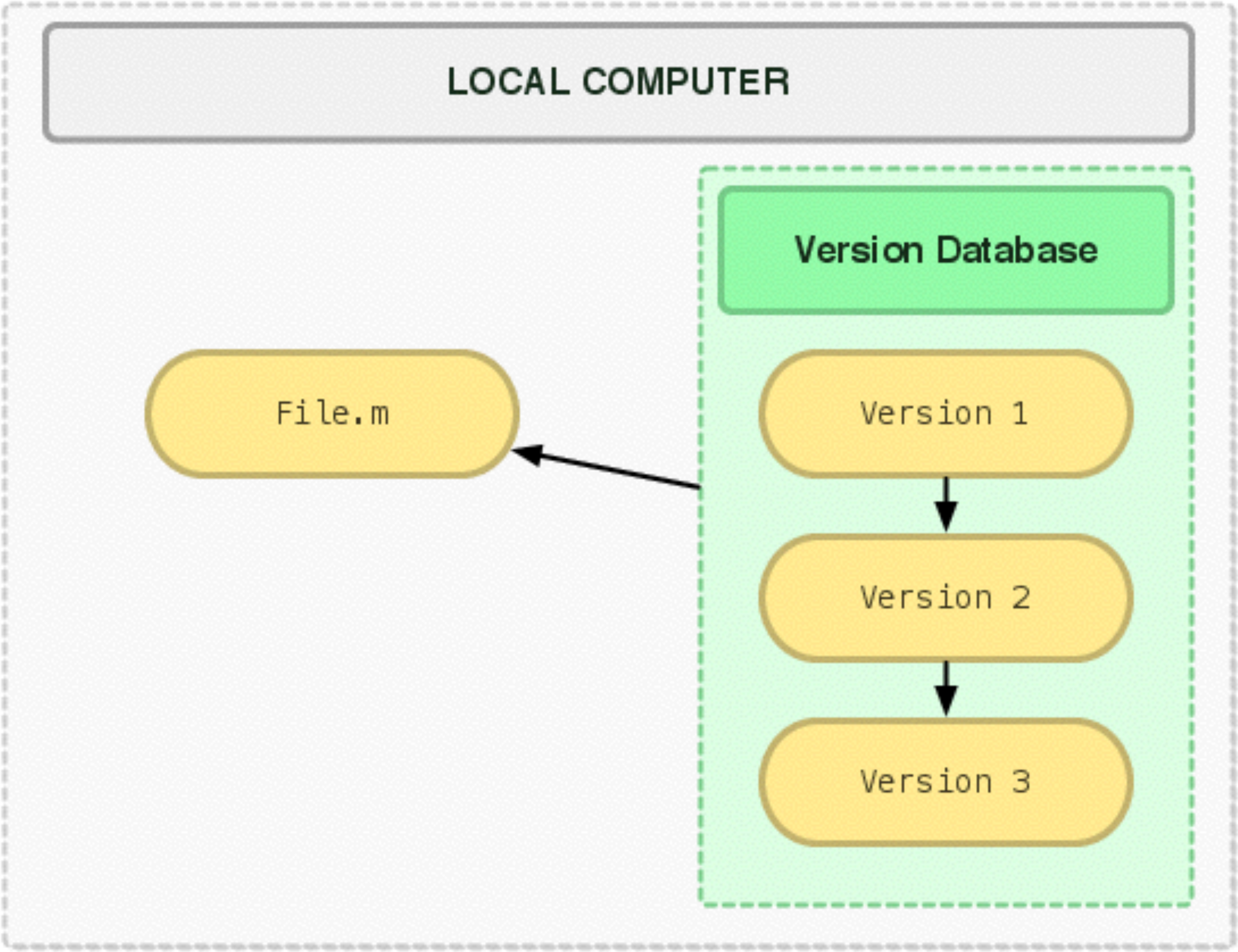
LOCAL COMPUTER

File.m

File copy.m

File copy 2.m

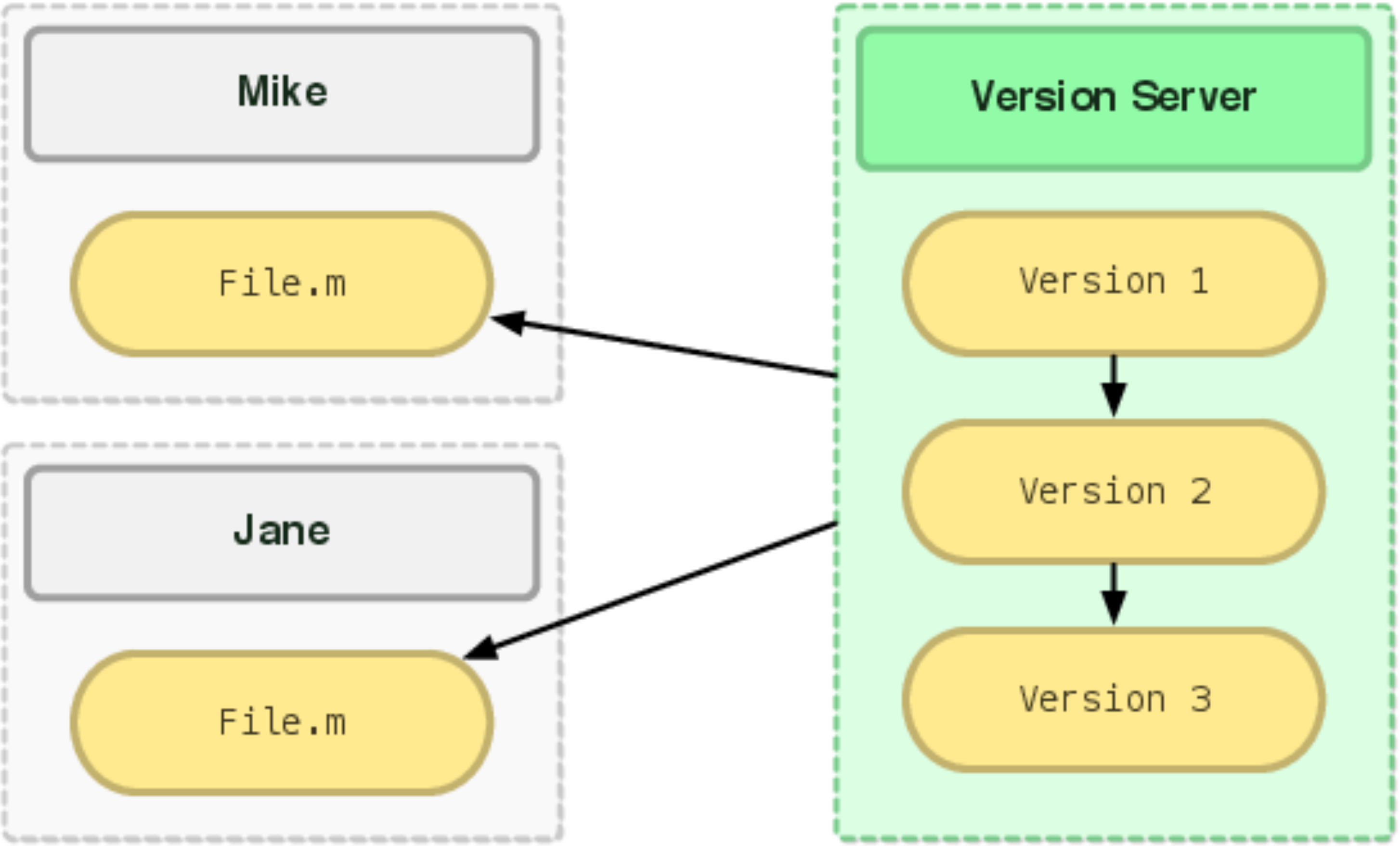
LOCAL
Version Control System



RCS: one image for each file or directory, each one with its own history.

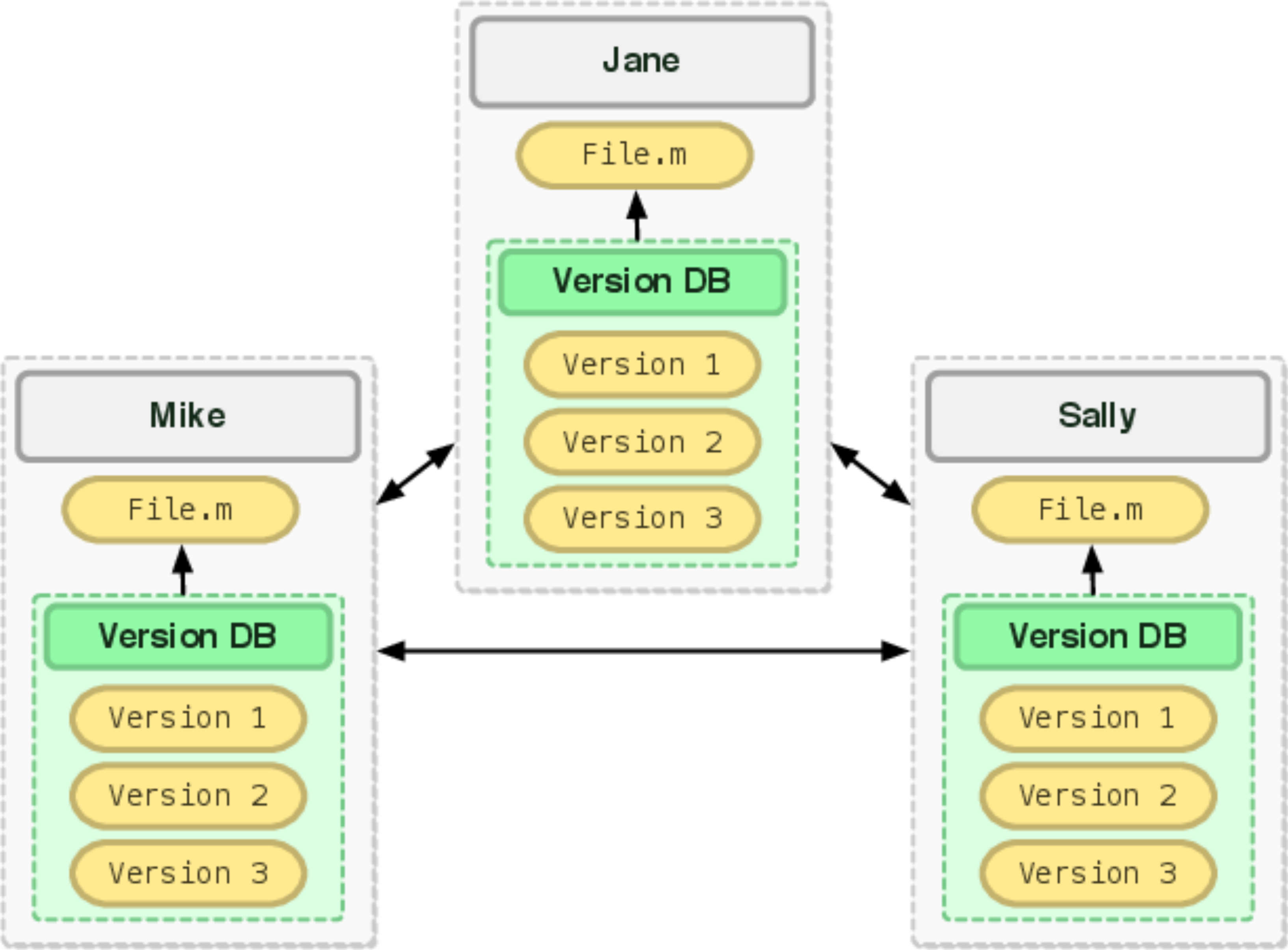
Master in High Performance Computing

CENTRALIZED Version Control System



CVS: centralized version of RCS,
with tags that address single version for each file or dir.

DISTRIBUTED Version Control System



Everything is Local

(Almost)

No Network Required

Create Repo

Commit

Merge

Branch

Rebase

Tag

Status

Revisions

Diff

History

Bisect

Local Sync

Advantages

Everything is Fast

Everything is Transparent

Every Clone is a Backup

You Can Work Offline

Storage

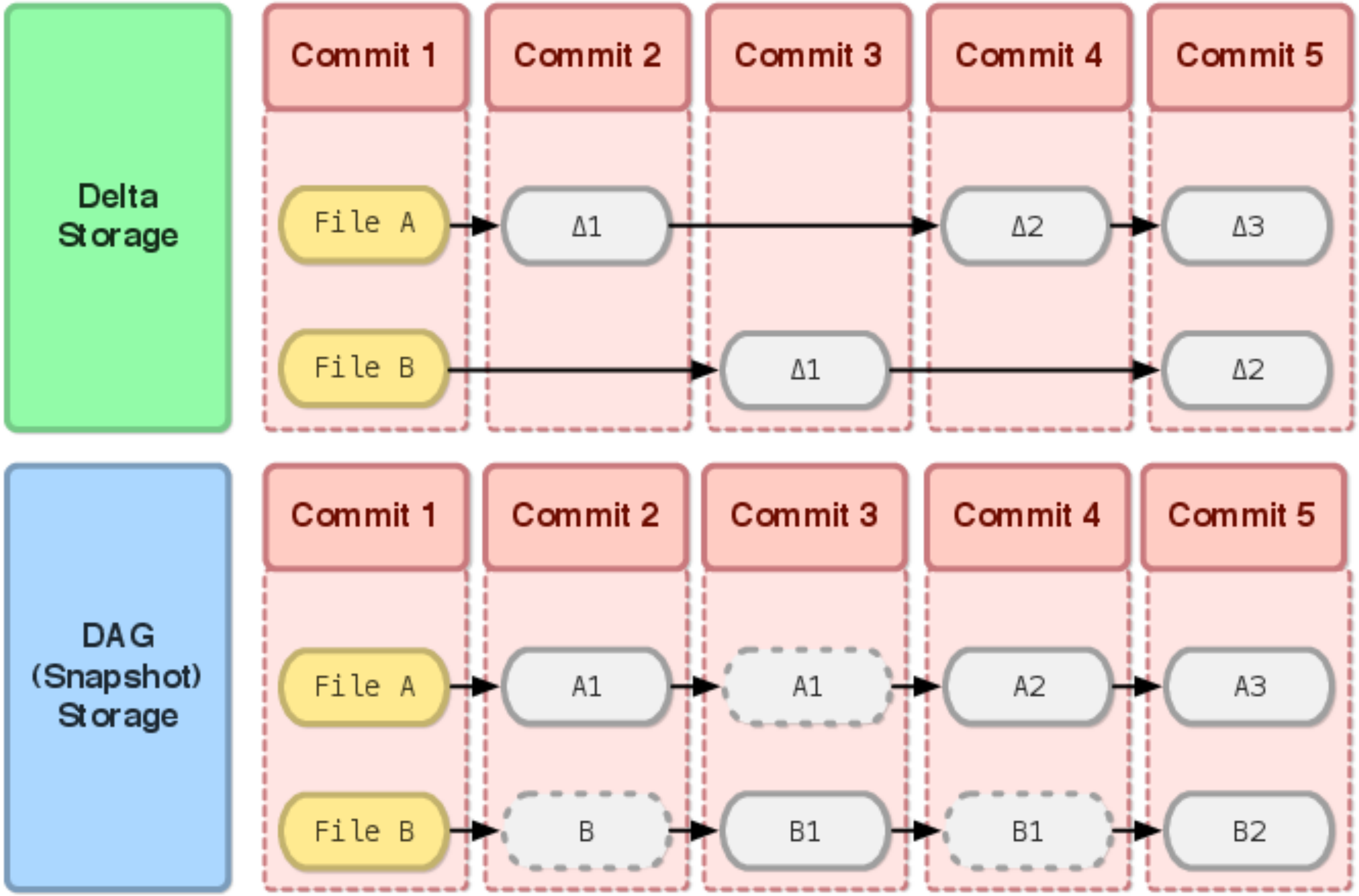
Delta Storage

Snapshot Storage

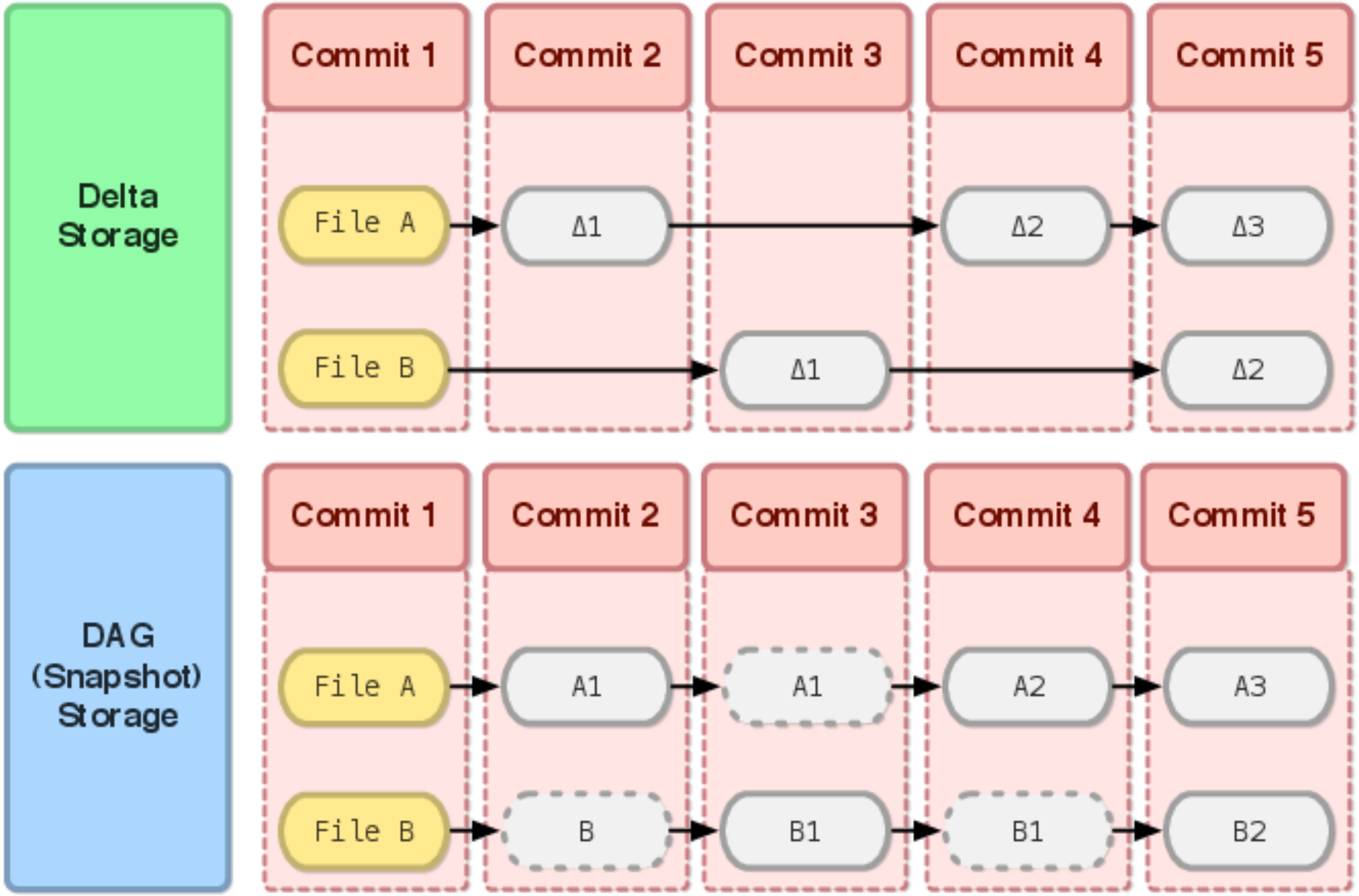
(a.k.a. Direct Acyclic Graph)

Delta vs. Snapshot

MHPC Master in High Performance Computing

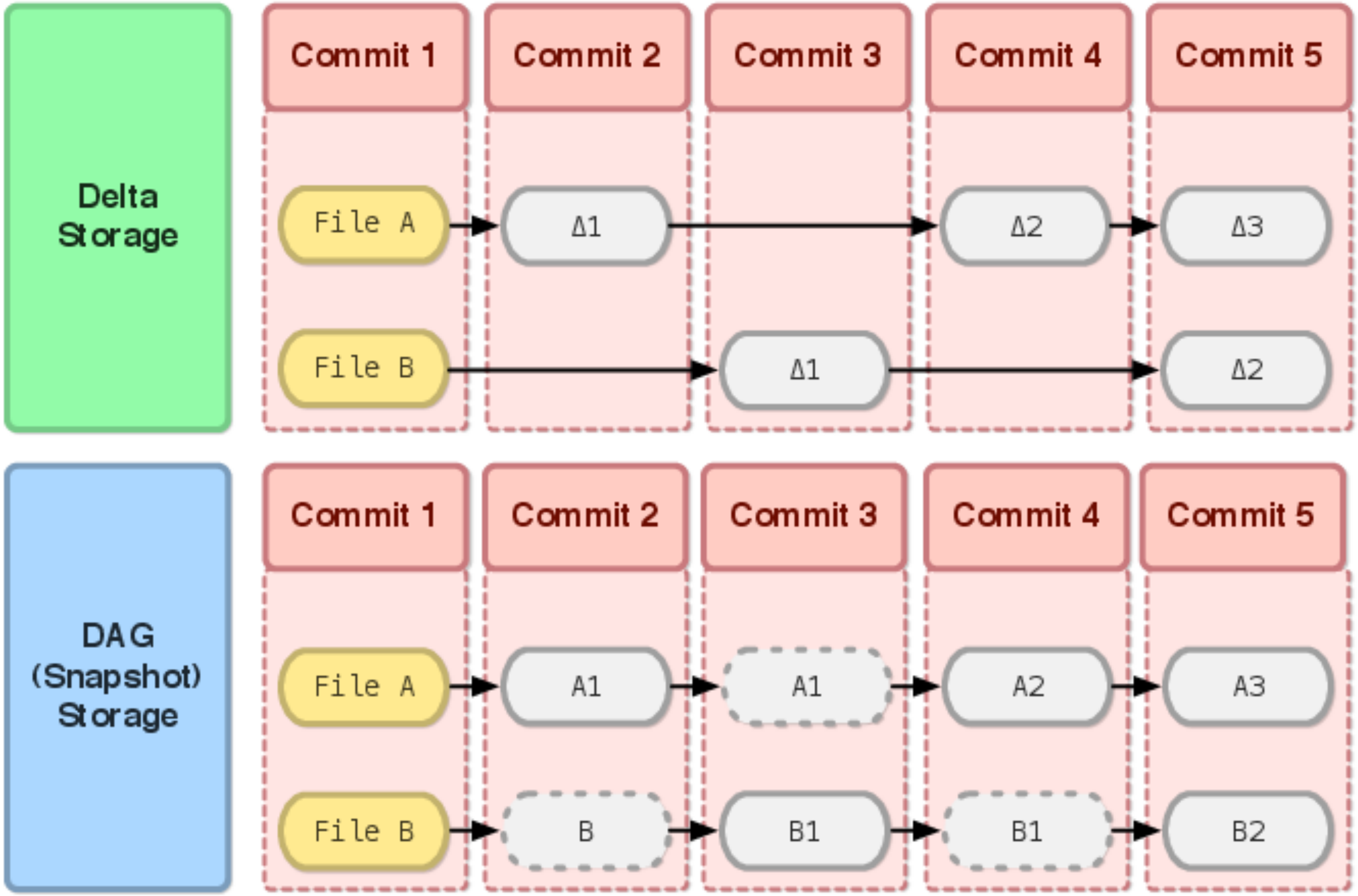


Delta vs. Snapshot

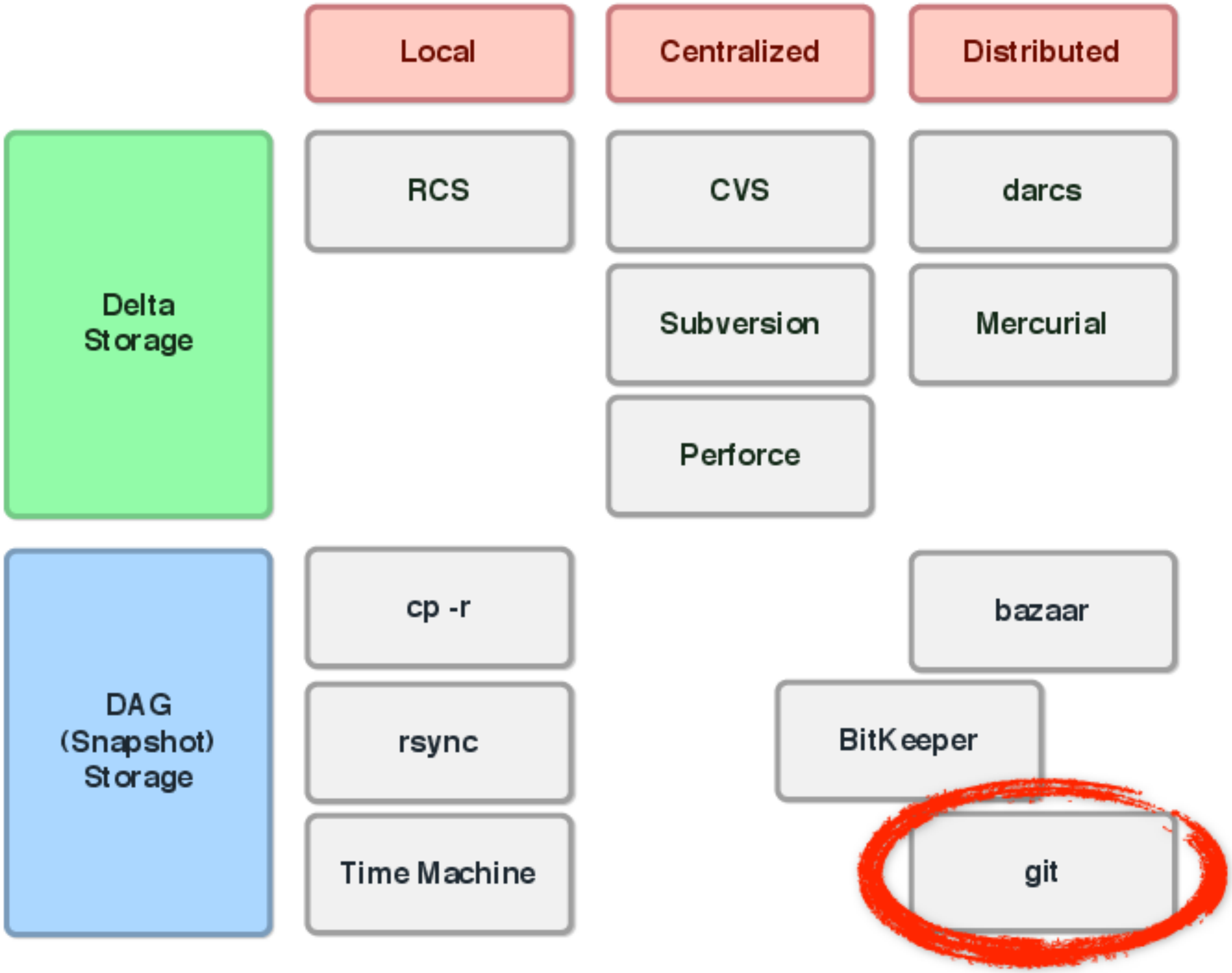


DELTA STORAGE, commit 3 and 4 are:
useless chunks of data.

Delta vs. Snapshot



SNAPSHOTS STORAGE, commit 3 and 4 are:
independent repository versions.



Local

Centralized

Distributed

Delta Storage

RCS

CVS

darcs

Subversion

Mercurial

Perforce

DAG (Snapshot) Storage

cp -r

bazaar

rsync

BitKeeper

Time Machine

git

About Git

Git in a Nutshell

Free and Open Source

Distributed Version Control System

Designed to Handle Large Projects

Fast and Efficient

Excellent Branching and Merging

Projects Using Git

Git

Linux

Perl

Eclipse

Qt

Rails

Android

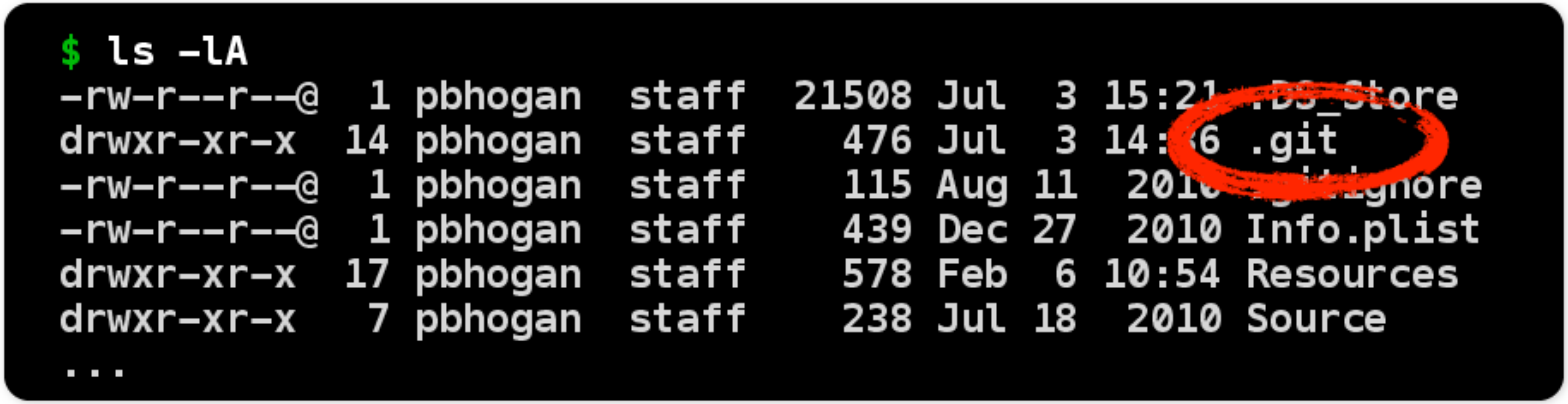
PostgreSQL

KDE

Gnome

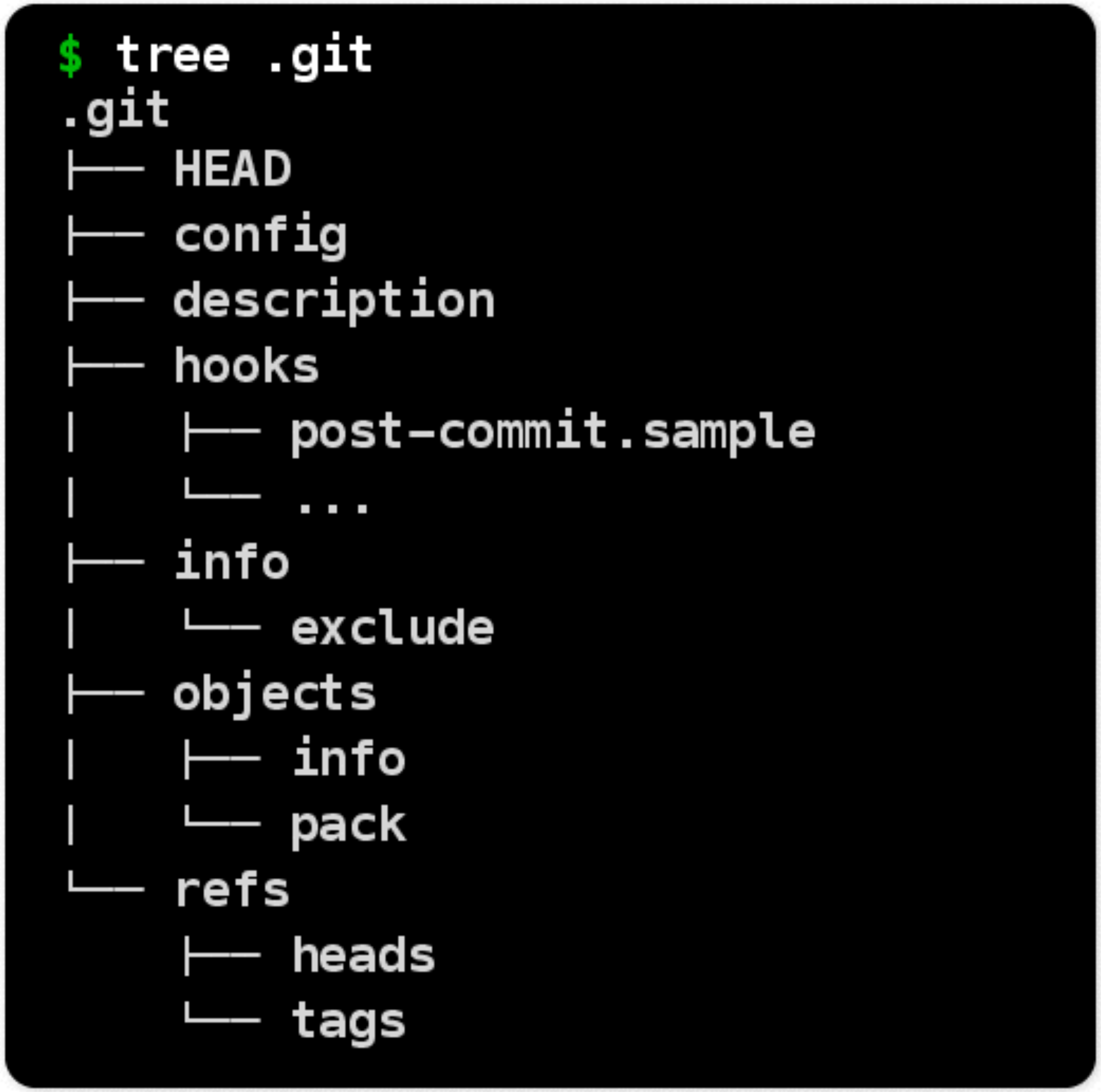
Under The Hood

Git Directory



```
$ ls -lA
-rw-r--r--@  1 pbhogan  staff  21508 Jul  3 15:21 .DS_Store
drwxr-xr-x  14 pbhogan  staff    476 Jul  3 14:36 .git
-rw-r--r--@  1 pbhogan  staff    115 Aug 11  2010 .gitignore
-rw-r--r--@  1 pbhogan  staff    439 Dec 27  2010 Info.plist
drwxr-xr-x  17 pbhogan  staff    578 Feb  6 10:54 Resources
drwxr-xr-x   7 pbhogan  staff    238 Jul 18  2010 Source
...
```

Git Directory



```
$ tree .git
.git
├── HEAD
├── config
├── description
├── hooks
|    ├── post-commit.sample
|    └── ...
├── info
|    └── exclude
├── objects
|    ├── info
|    └── pack
└── refs
     ├── heads
     └── tags
```

.git only in root of Working Directory

(unlike Subversion)

Master in High Performance Computing

Git Directory

Configuration File

Hooks

Object Database

References

Index

Object Database

Four Object Types

Object Database

blob

tree

commit

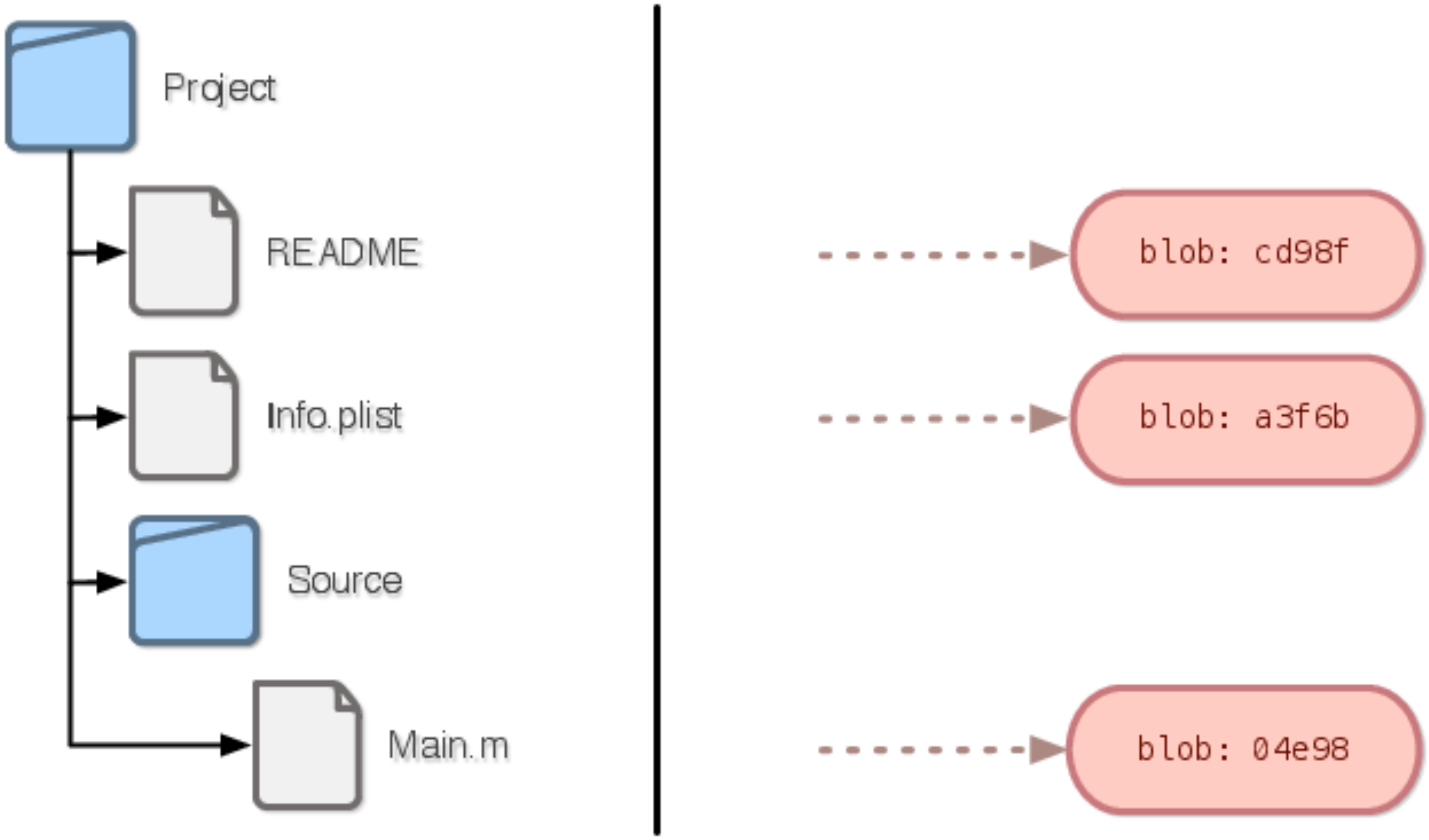
tag

Object Database

blob

Working Directory

Git Directory



Project

README

Info.plist

Source

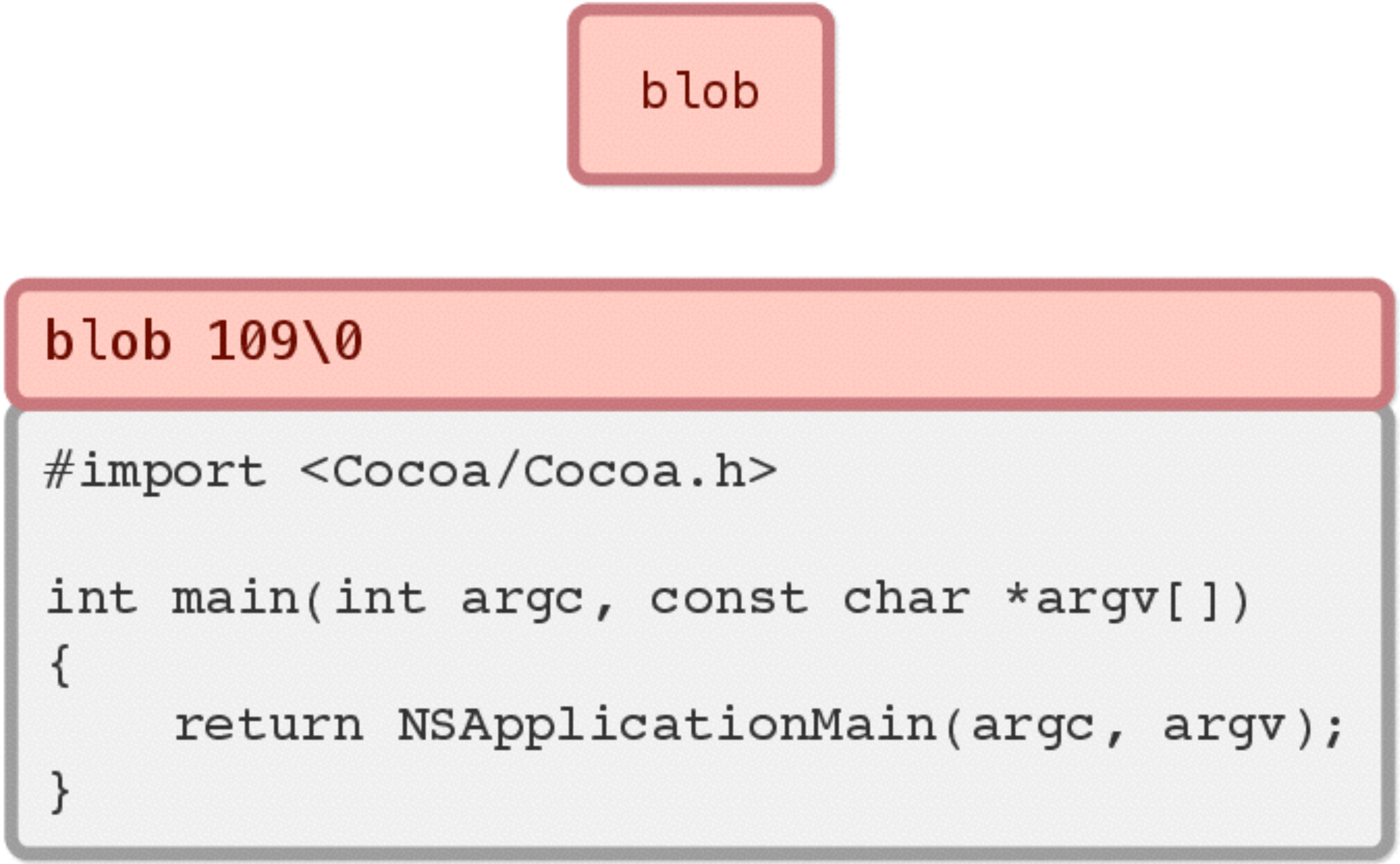
Main.m

blob: cd98f

blob: a3f6b

blob: 04e98

Object Database



blob

```
blob 109\0
```

```
#import <Cocoa/Cocoa.h>

int main(int argc, const char *argv[])
{
    return NSApplicationMain(argc, argv);
}
```

Object Database

blob

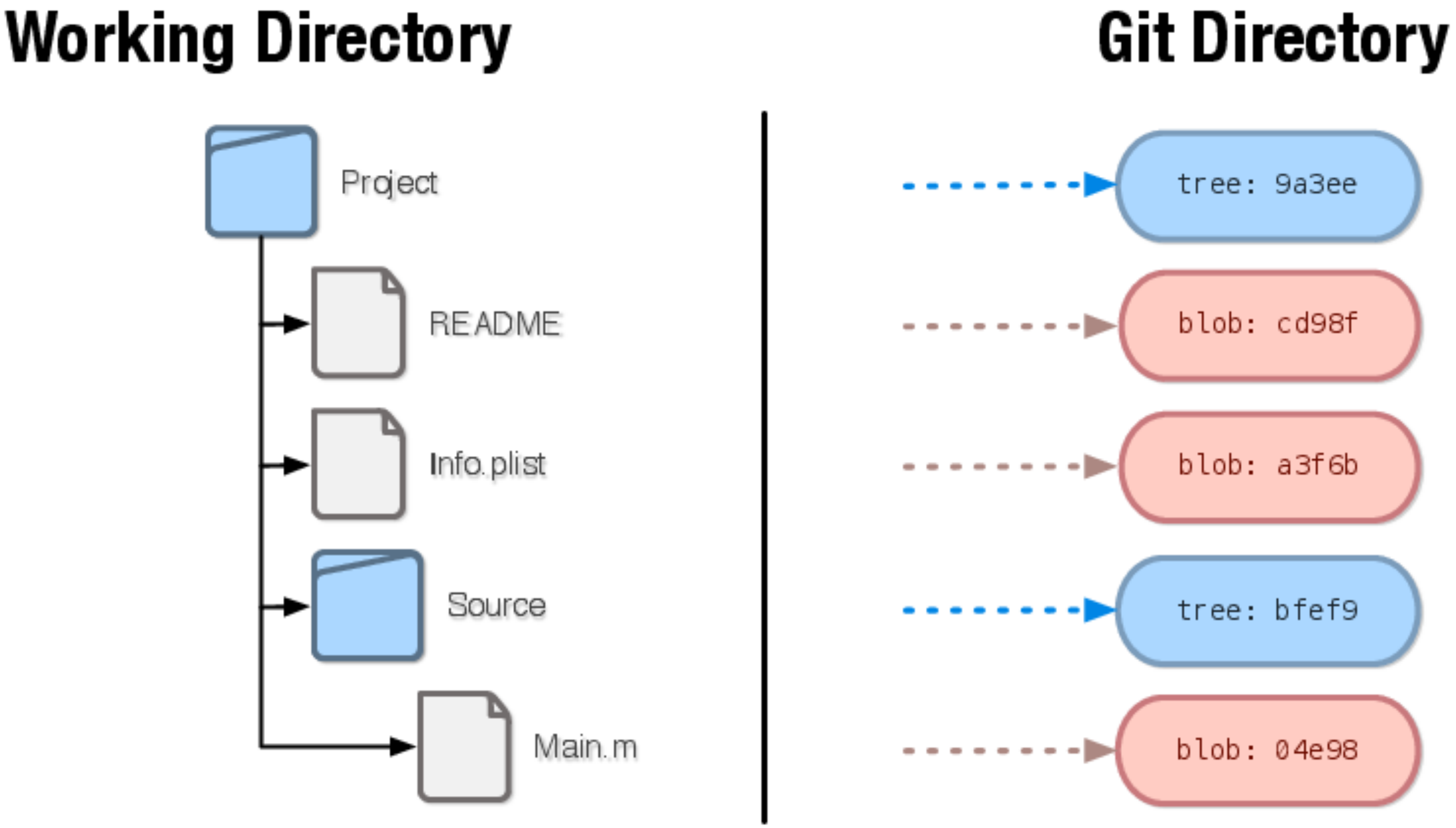
tree

commit

tag

Object Database

tree



Object Database

tree

```
tree 84\0
```

```
100644 blob cd98f   README
100644 blob a3f6b   Info.plist
040000 tree bfef9   Source
```

Object Database

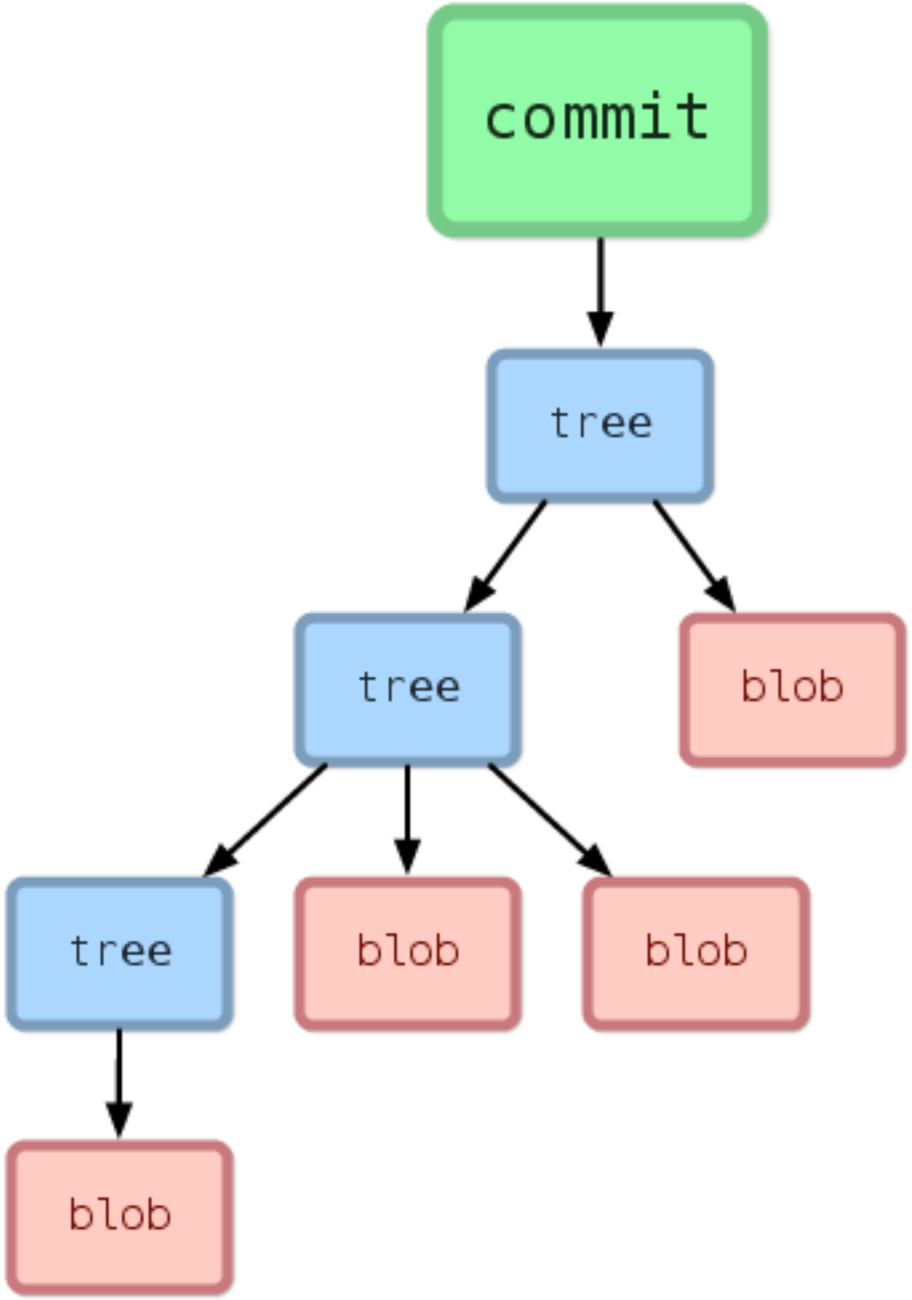
blob

tree

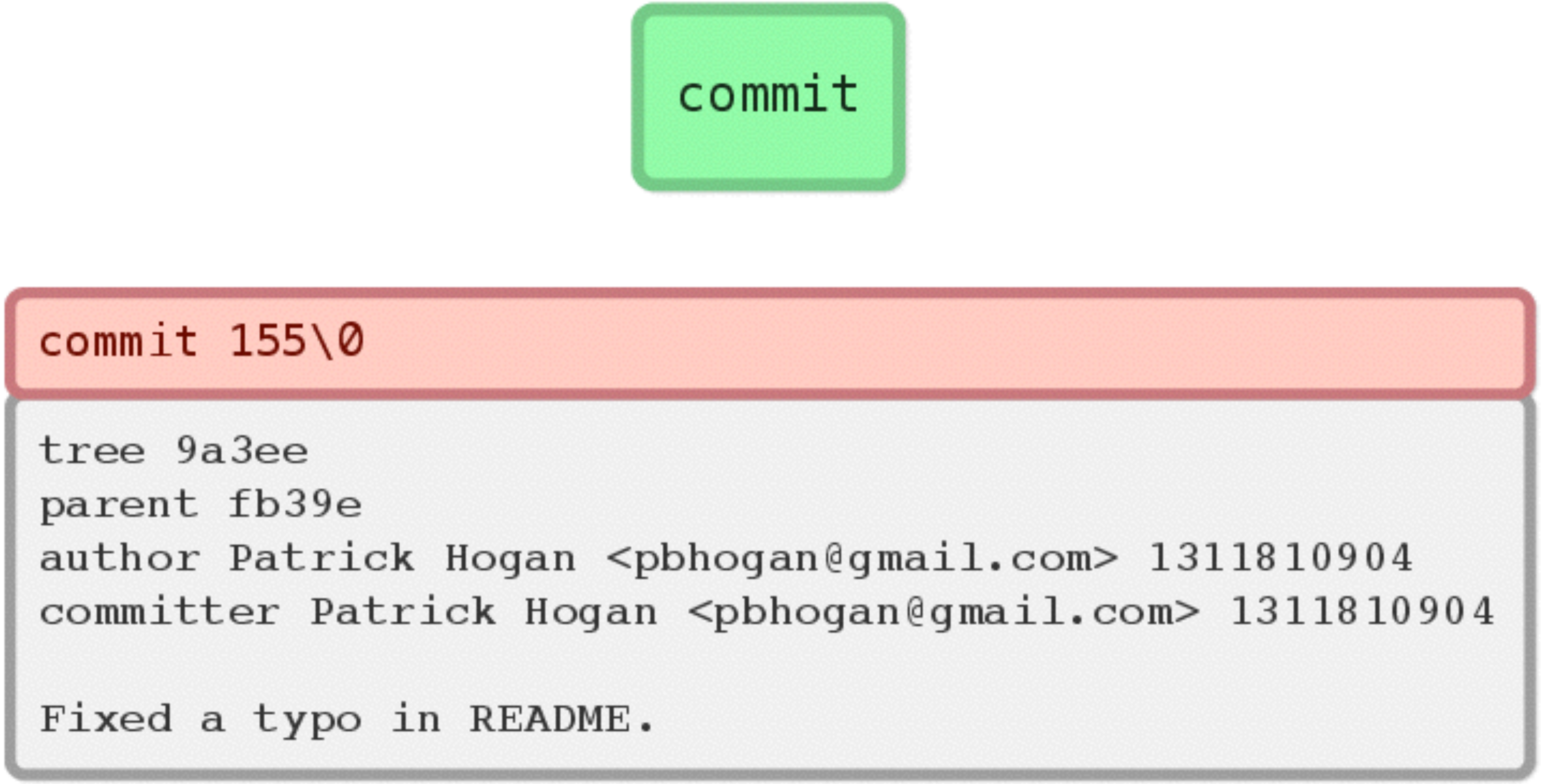
commit

tag

Object Database



Object Database

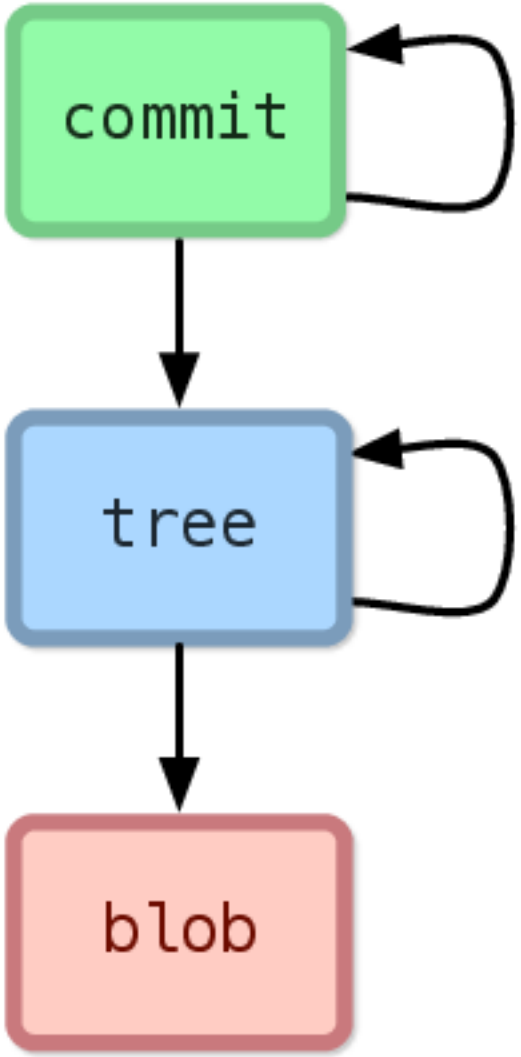


commit

```
commit 155\0
```

```
tree 9a3ee
parent fb39e
author Patrick Hogan <pbhogan@gmail.com> 1311810904
committer Patrick Hogan <pbhogan@gmail.com> 1311810904

Fixed a typo in README.
```

Object Database



Object Database

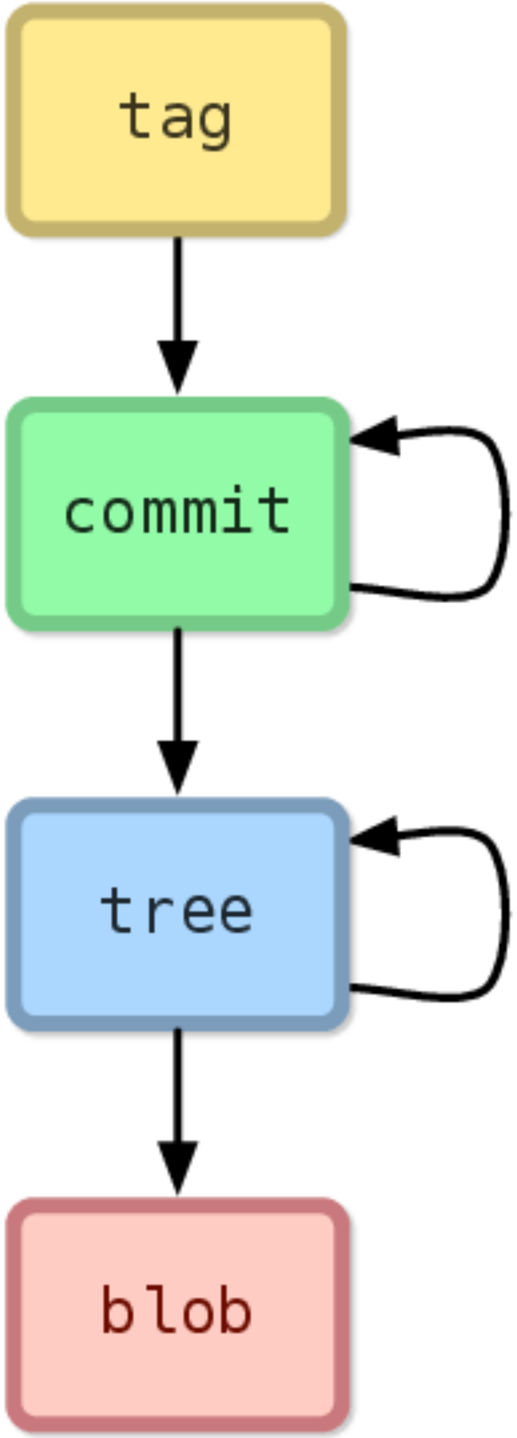
blob

tree

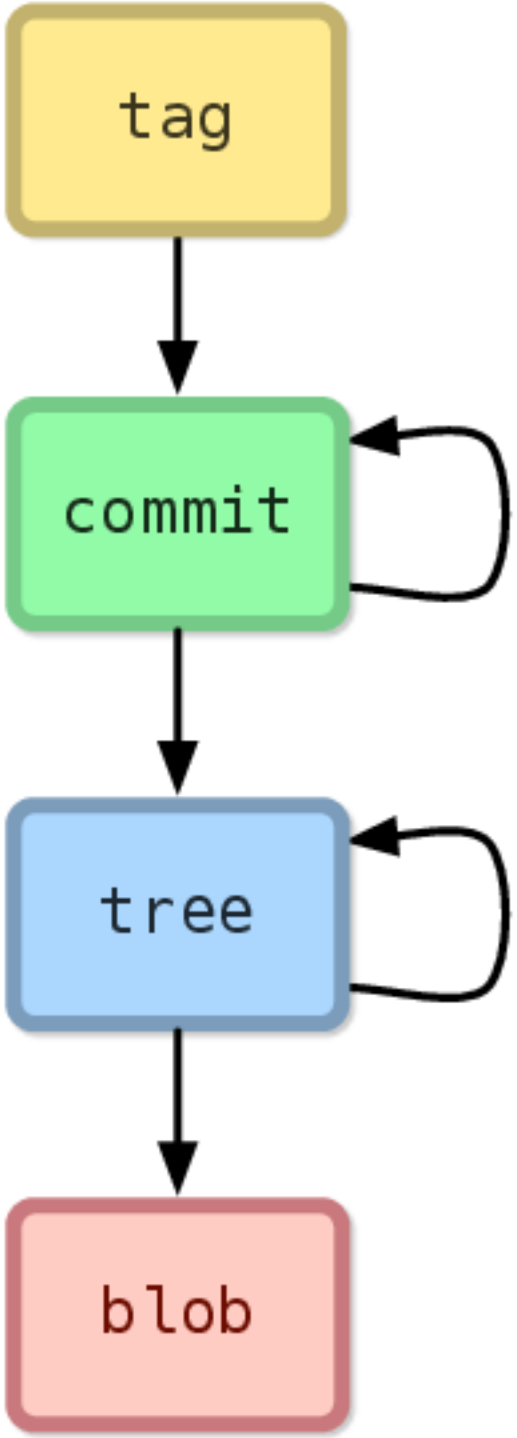
commit

tag

Object Database

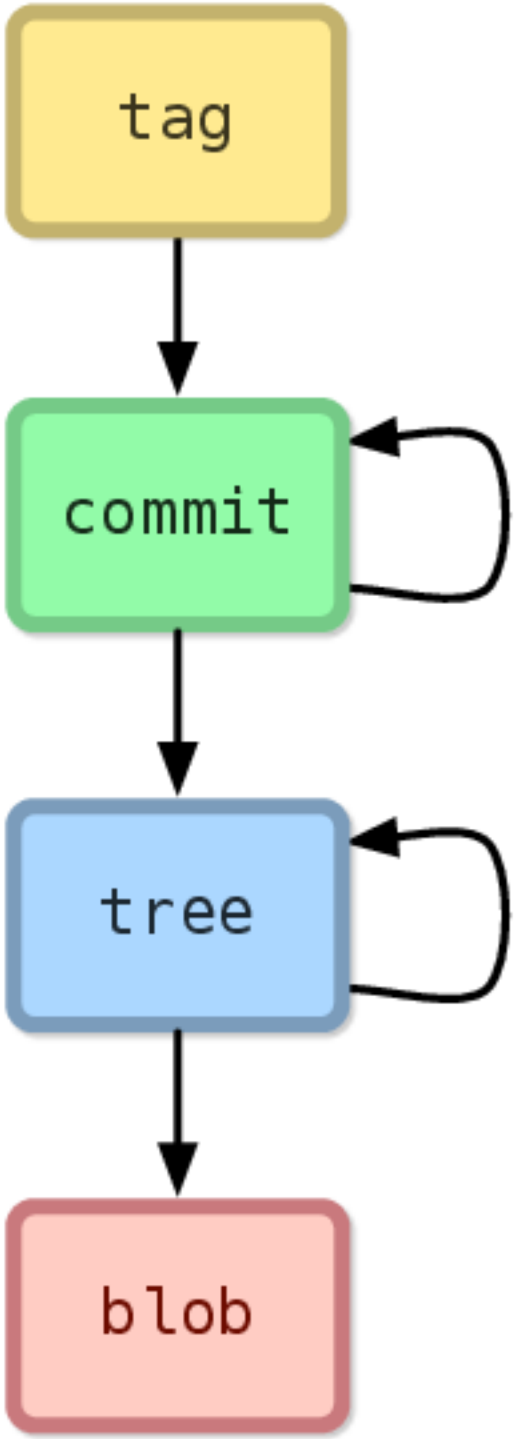


Object Database



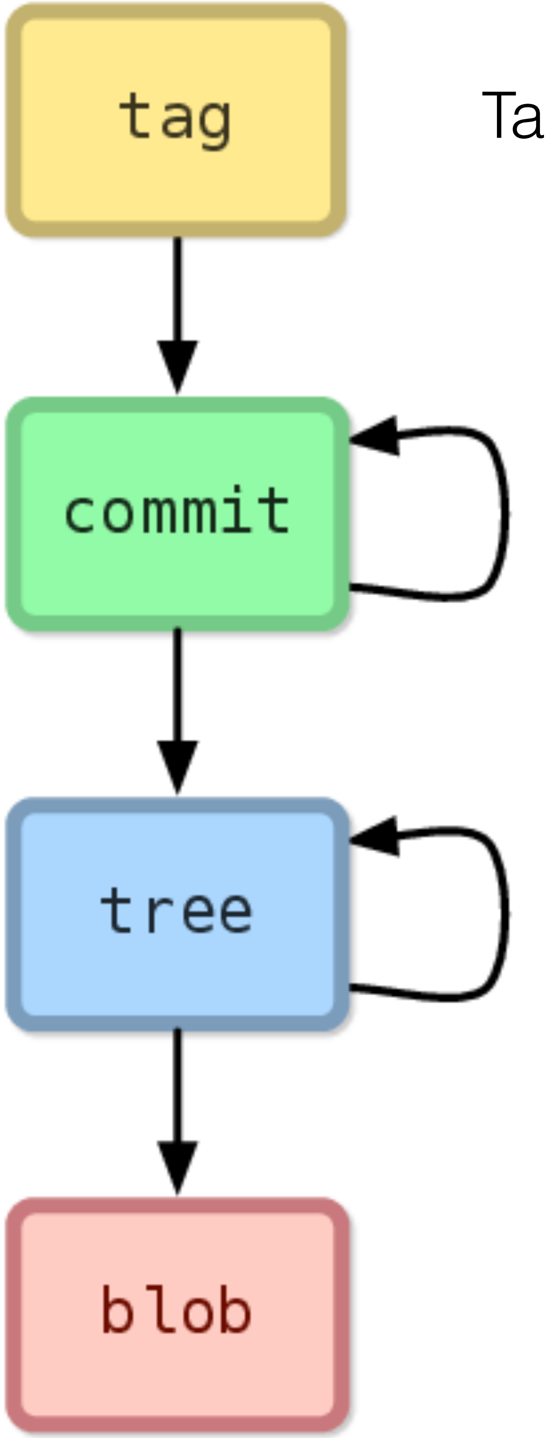
We saw that tree point
to each other.

Object Database



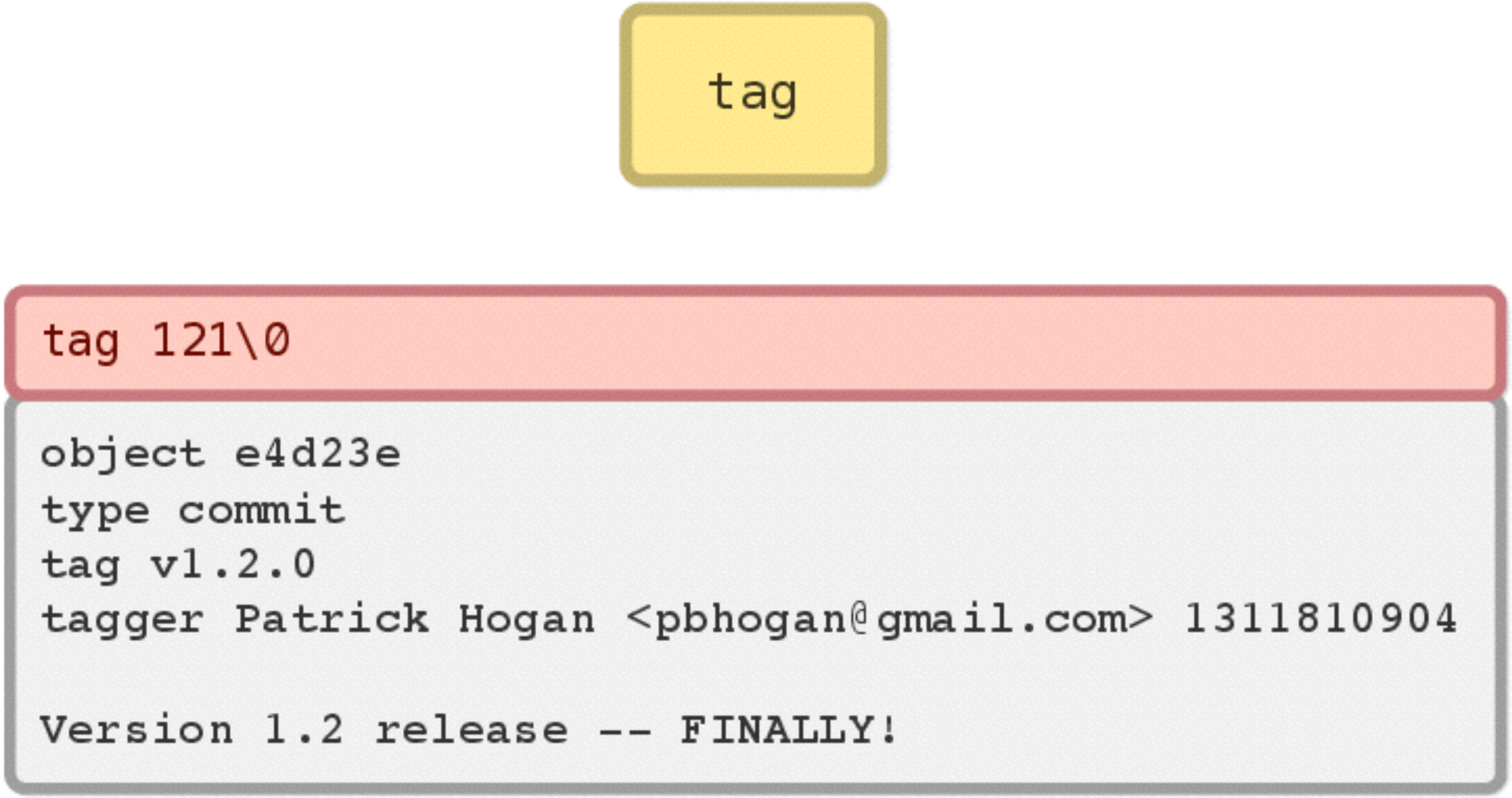
We will see that commit point to each other.

Object Database



Tag is a pointer to a commit.

Object Database



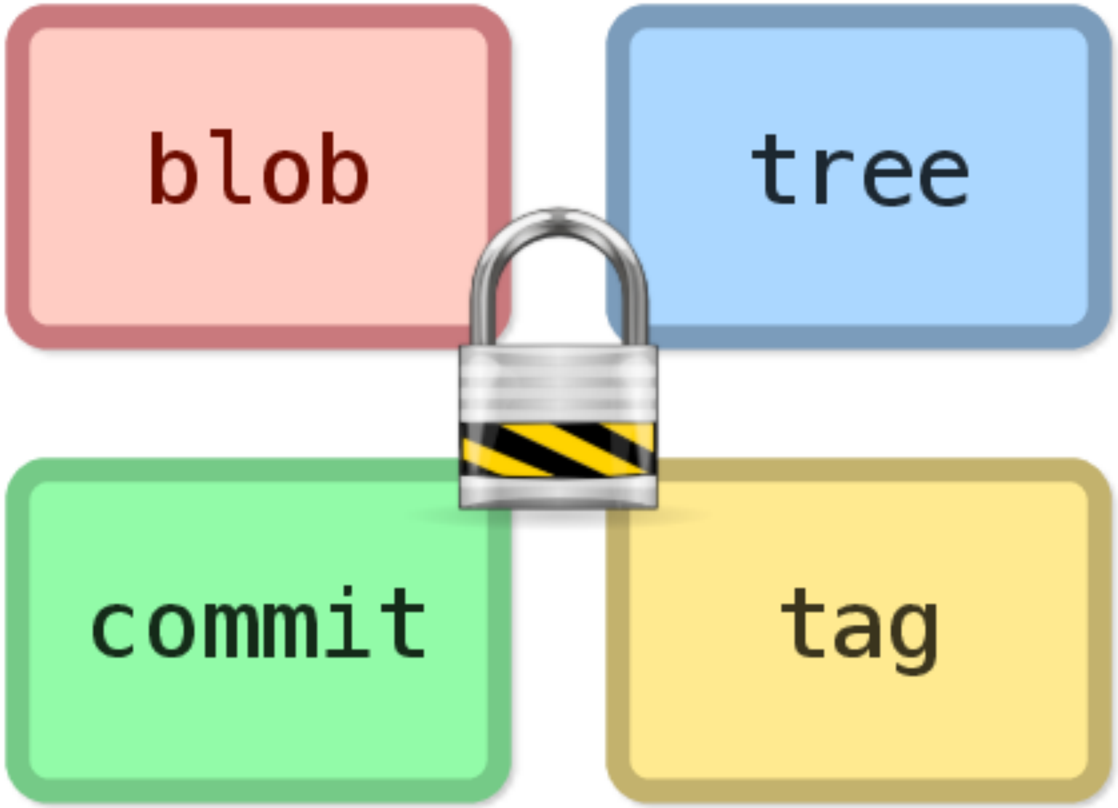
```
tag 121\0
```

```
object e4d23e
type commit
tag v1.2.0
tagger Patrick Hogan <pbhogan@gmail.com> 1311810904

Version 1.2 release -- FINALLY!
```

`.git/objects/20/c71174453dc760692cd1461275bf0cffeb772f`

`.git/refs/tags/v1.2.0`

Object Database



Immutable!

Never Removes Data

(Almost)

"Rewriting History" Writes Alternate History

Git Directory

Configuration File

Hooks

Object Database

References

Index

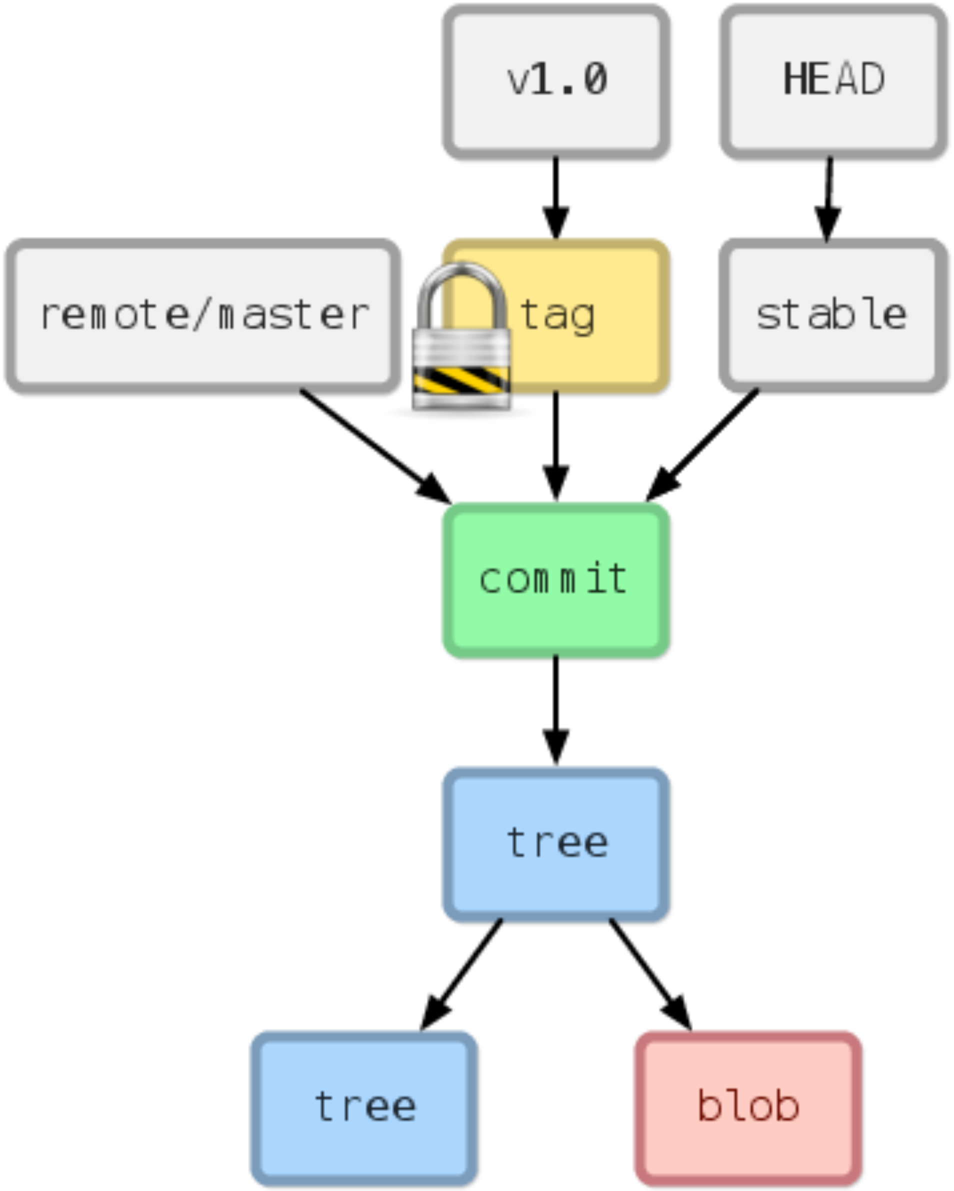
References

References

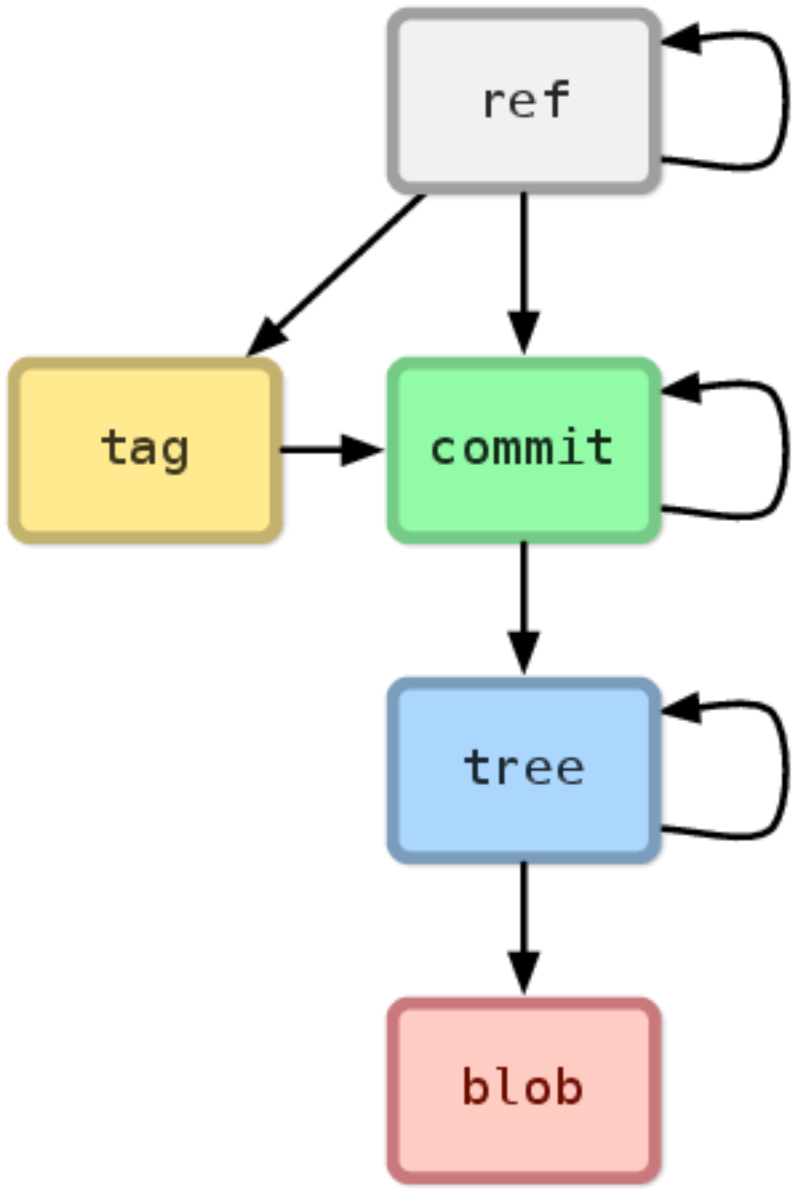
Lightweight, Movable Pointers to Commits

(and other things)

References

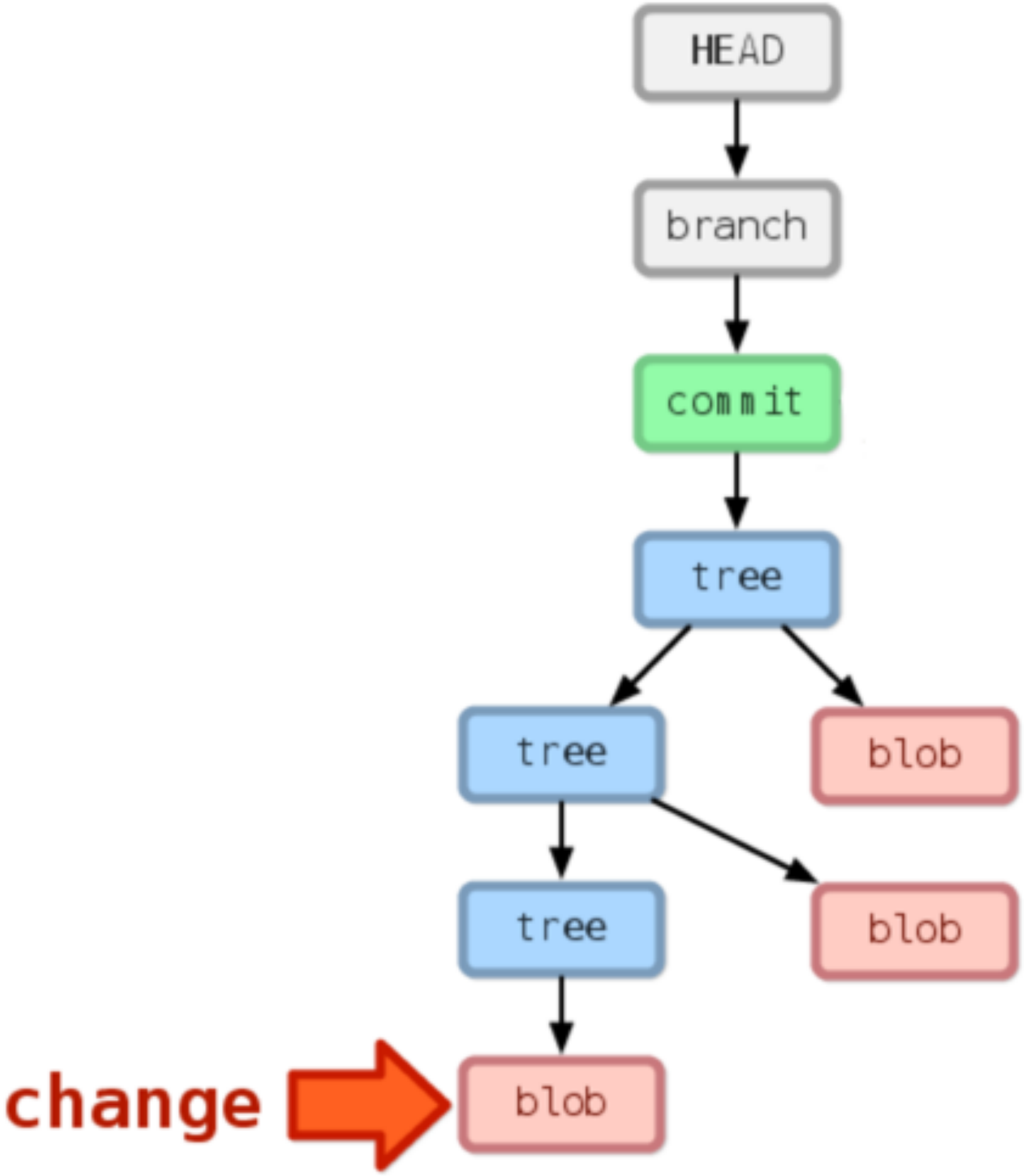


References

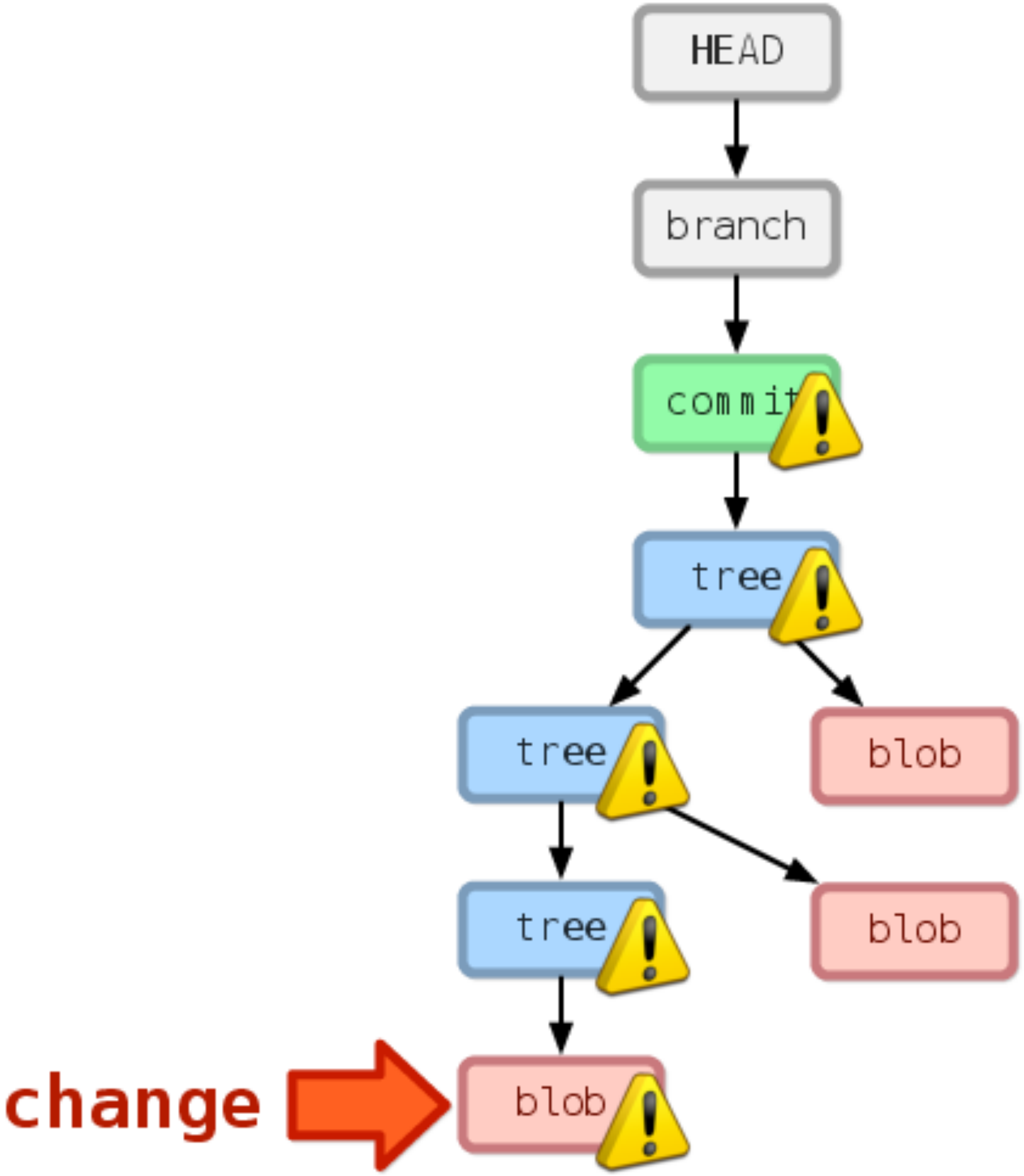


Scenario

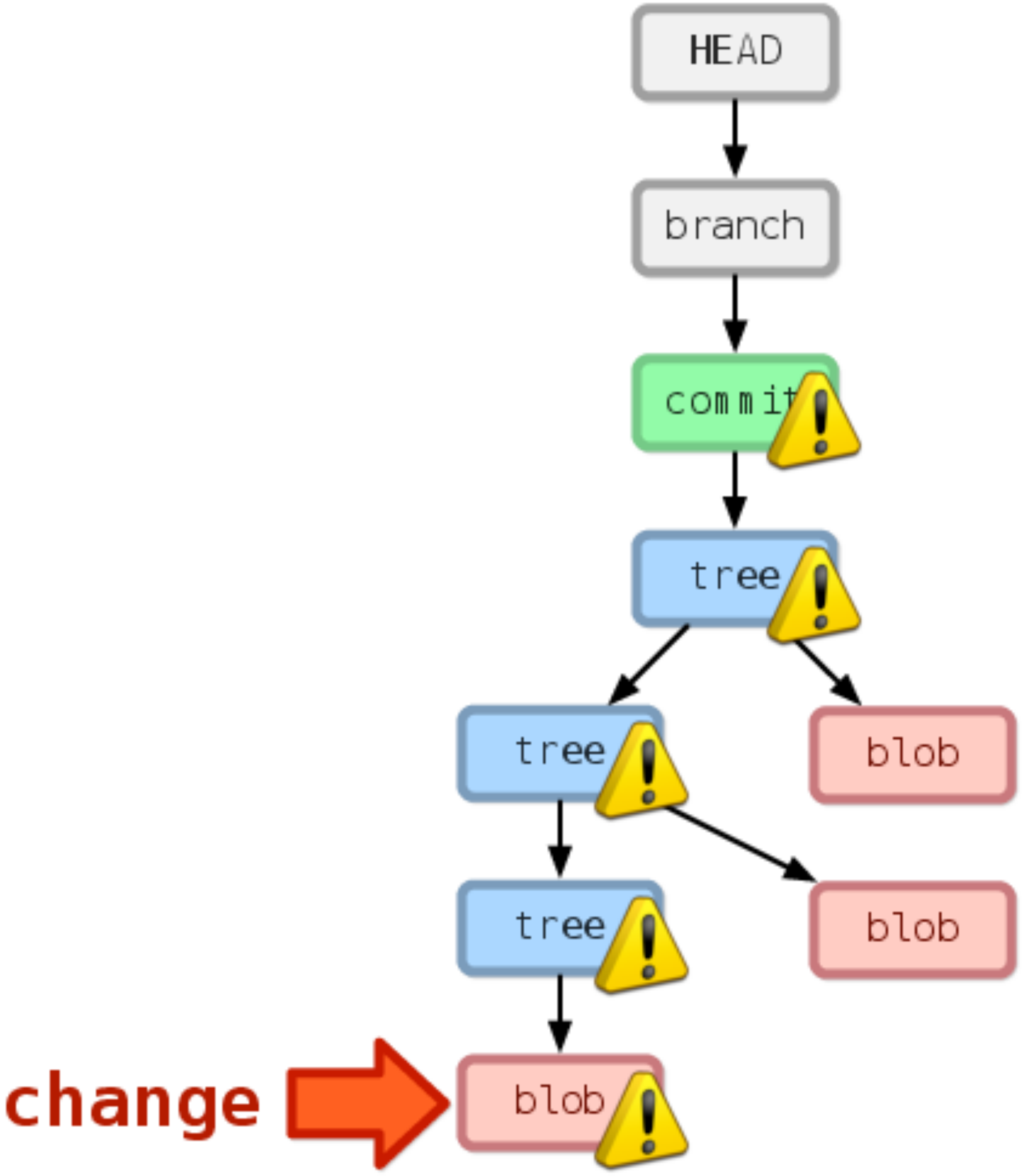
Scenario



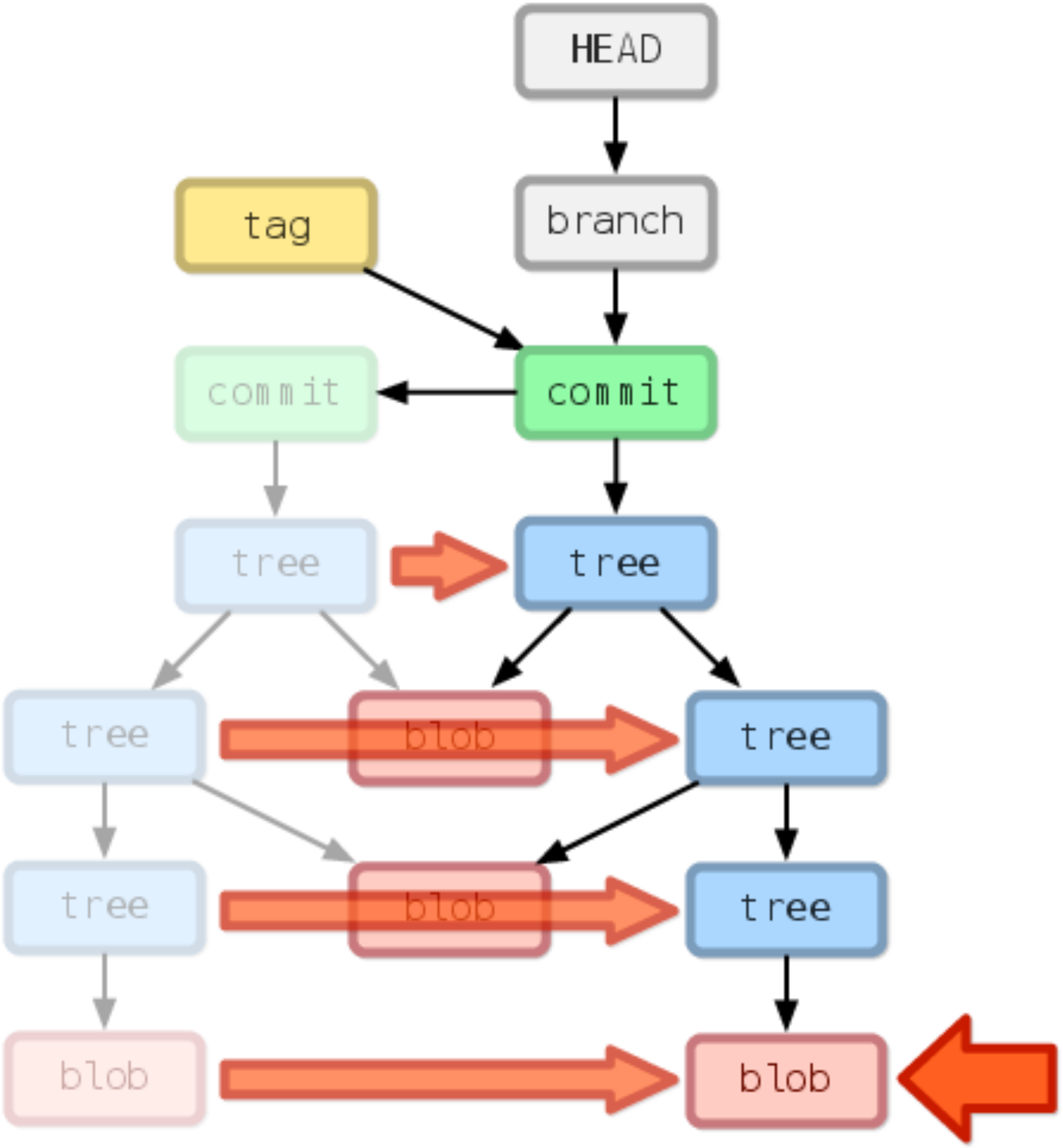
Scenario



Scenario

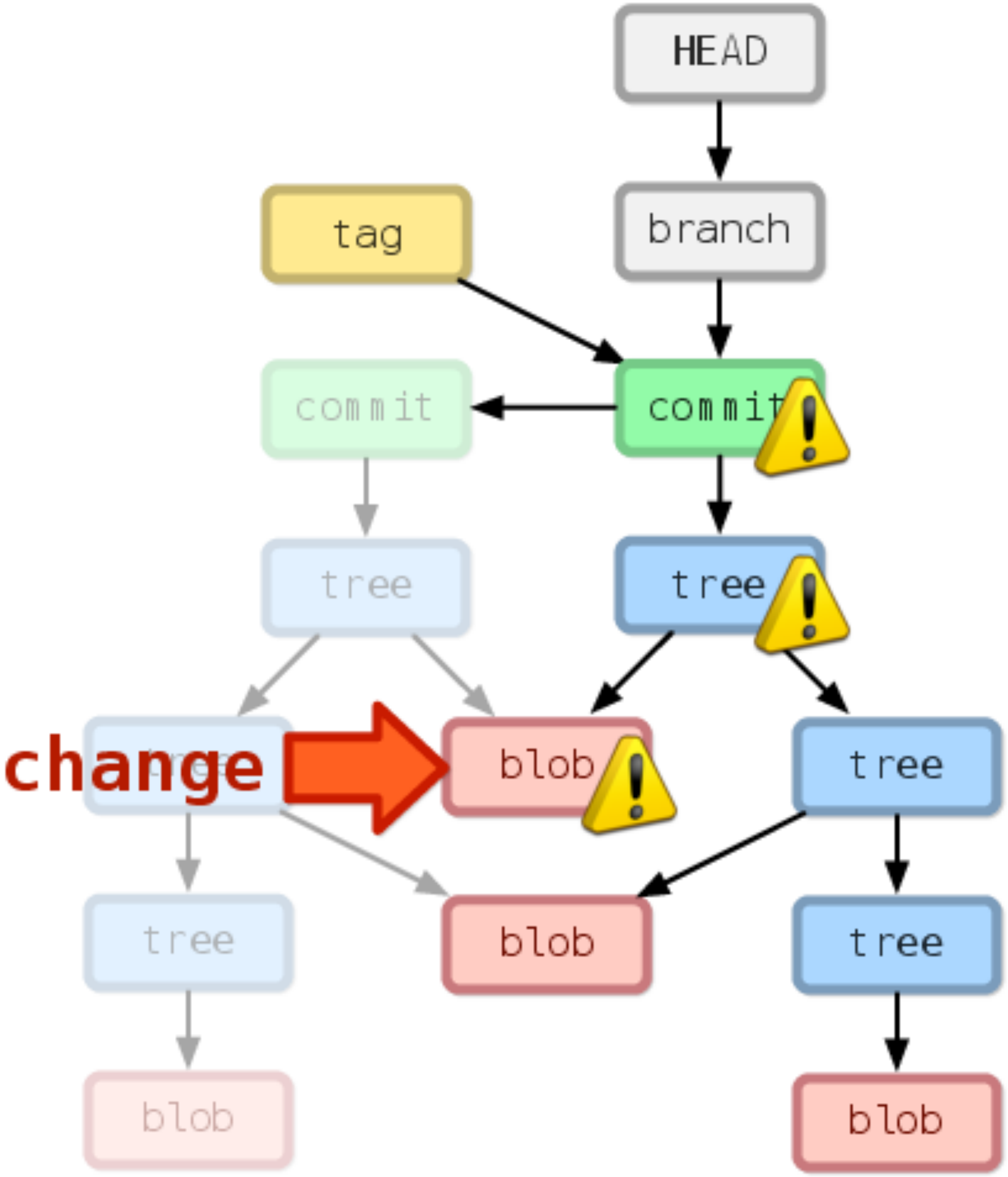


Scenario



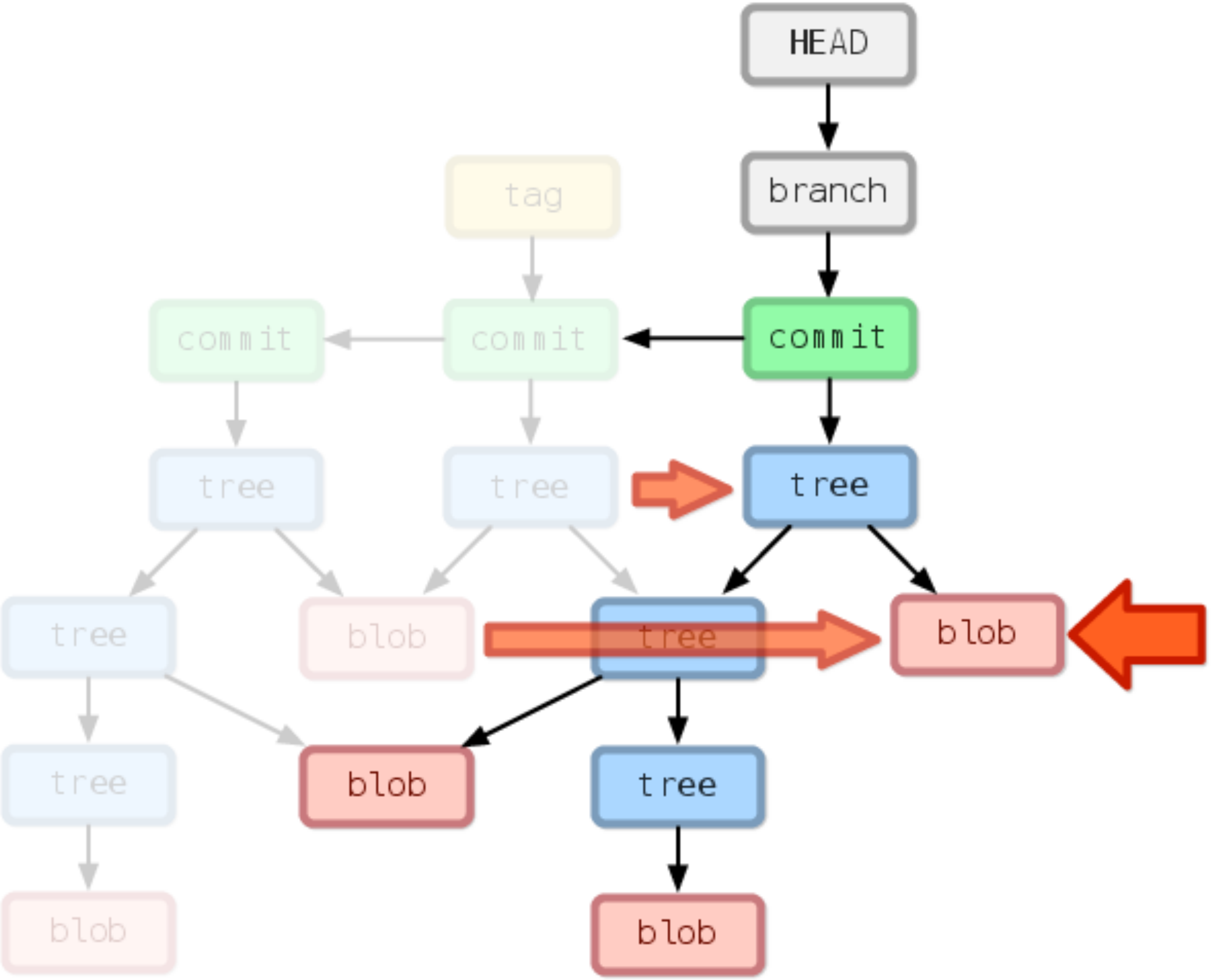
new objects

Scenario

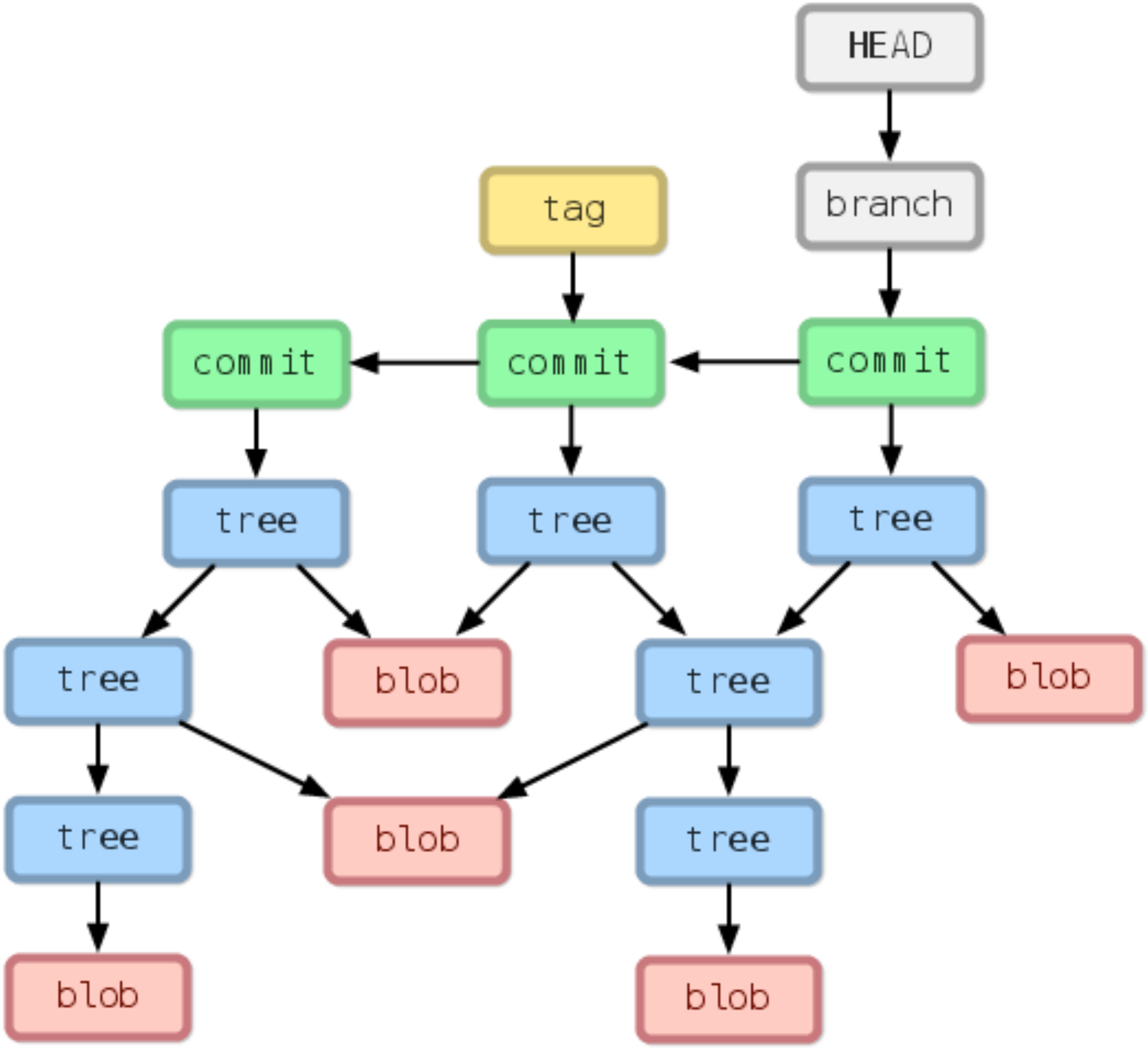


HEAD
tag
branch
commit
commit
tree
tree
tree
change
blob
tree
tree
blob
tree
blob
blob

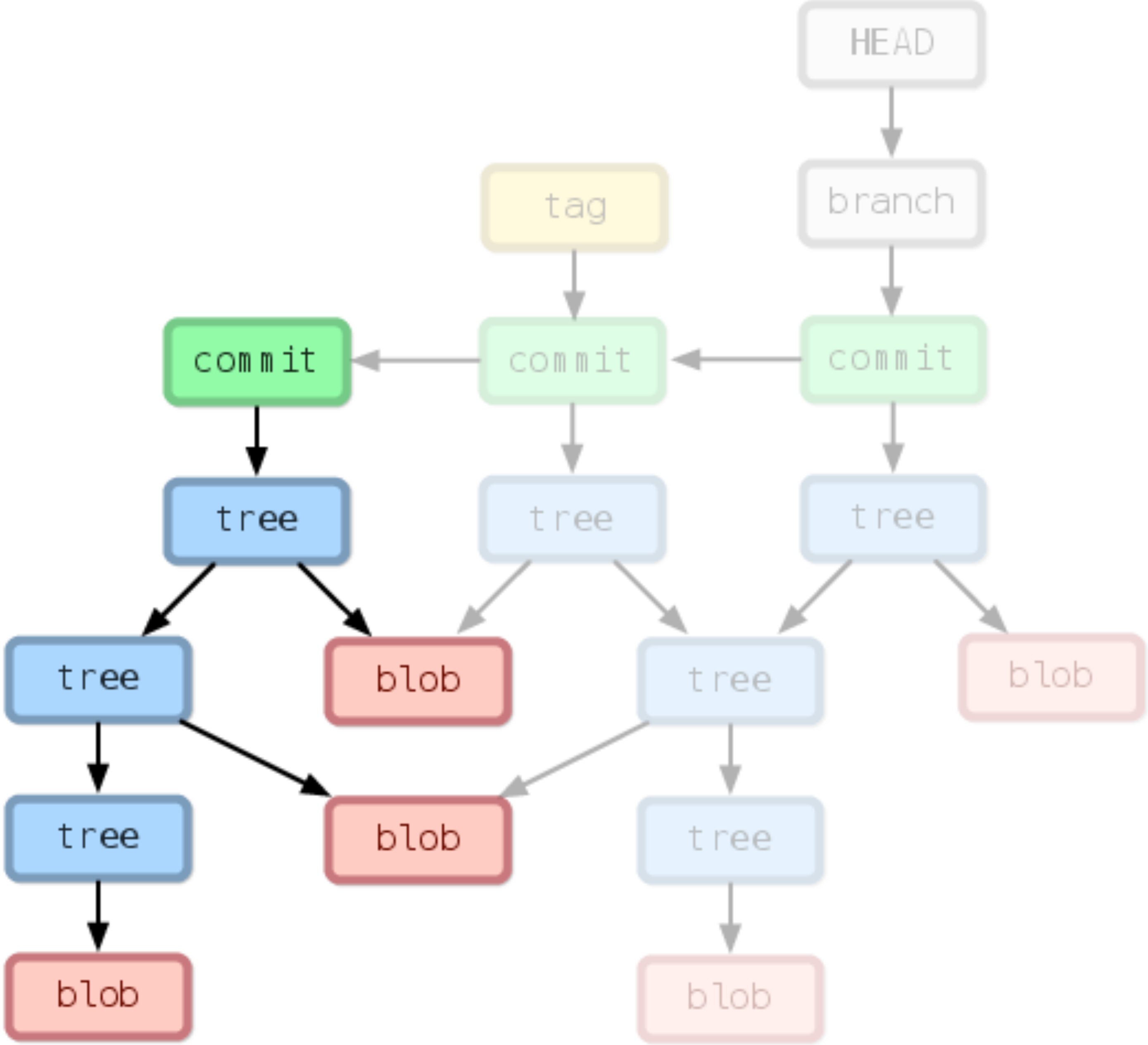
Scenario


HEAD
tag
branch
commit
commit
commit
tree
tree
tree
tree
blob
tree
blob
tree
blob
tree
blob
blob

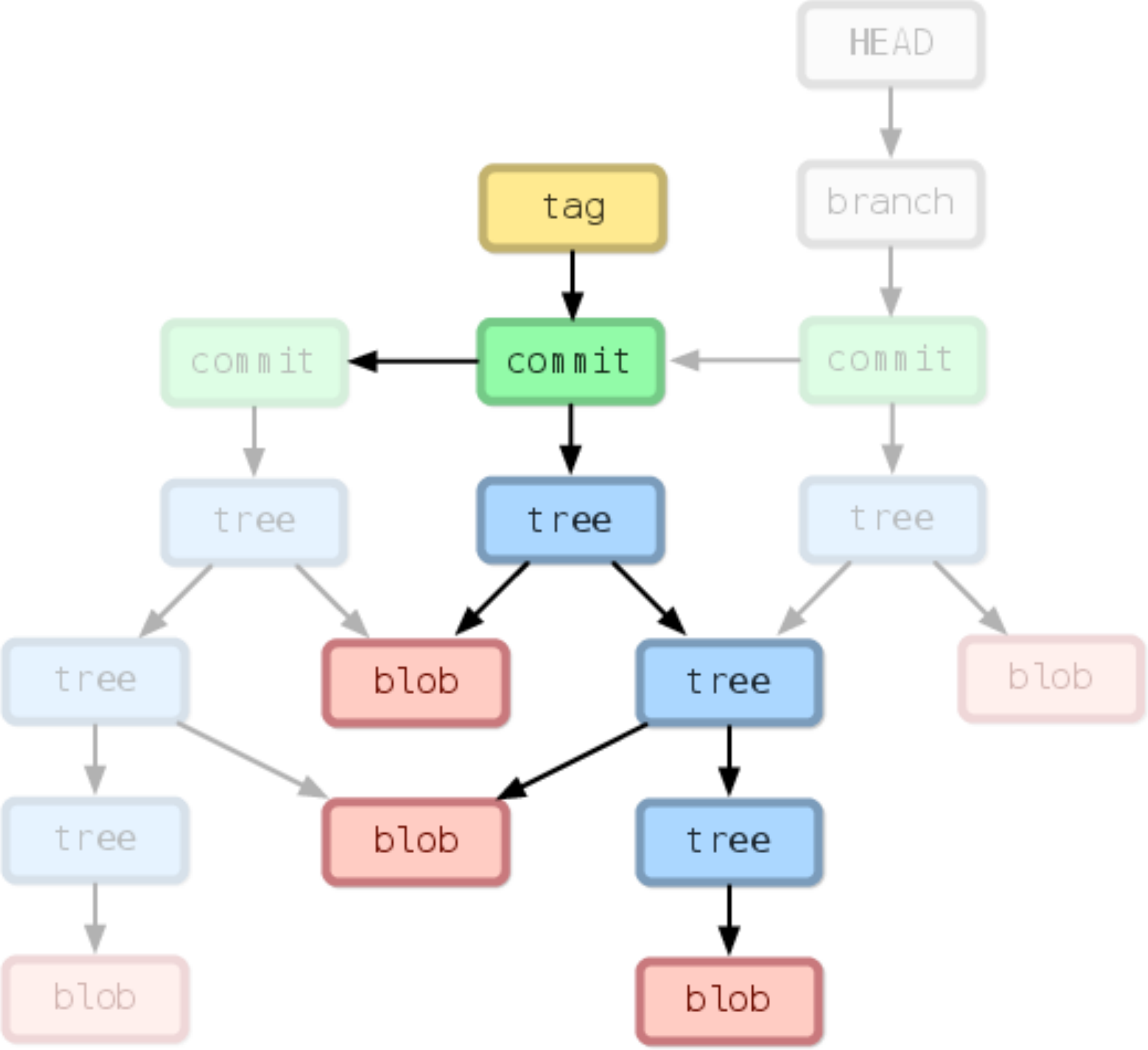
Scenario


HEAD
tag
branch
commit
commit
commit
tree
tree
tree
tree
blob
tree
blob
tree
blob
tree
blob
blob

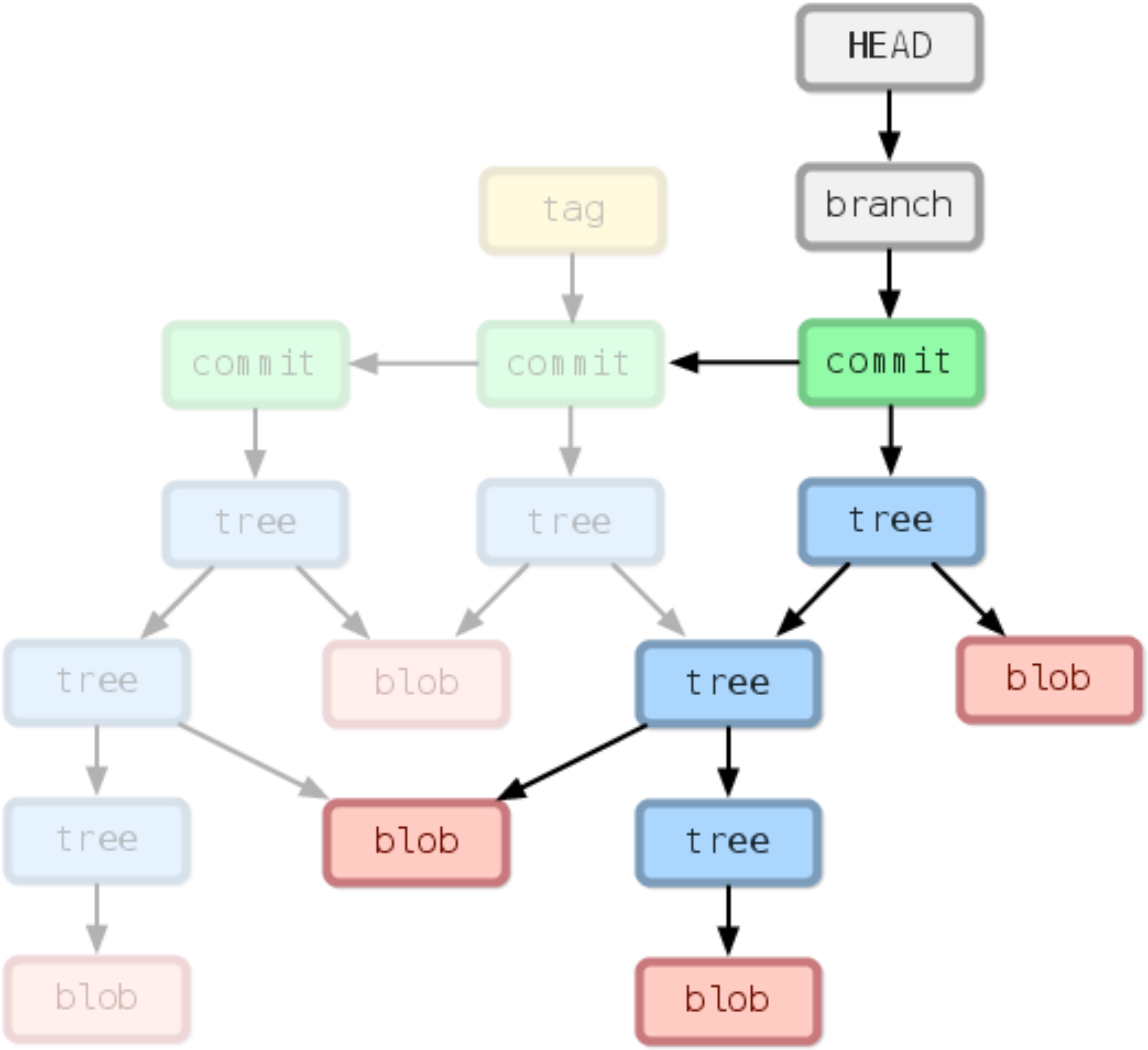
Scenario


HEAD
tag
branch
commit
commit
commit
tree
tree
tree
tree
blob
tree
blob
tree
blob
tree
blob
blob

Scenario


HEAD
tag
branch
commit
commit
commit
tree
tree
tree
tree
blob
tree
blob
tree
blob
tree
blob
blob
blob

Scenario


HEAD
branch
tag
commit
commit
commit
tree
tree
tree
tree
blob
tree
blob
tree
blob
tree
blob
blob

Git Directory

Configuration File

Hooks

Object Database

References

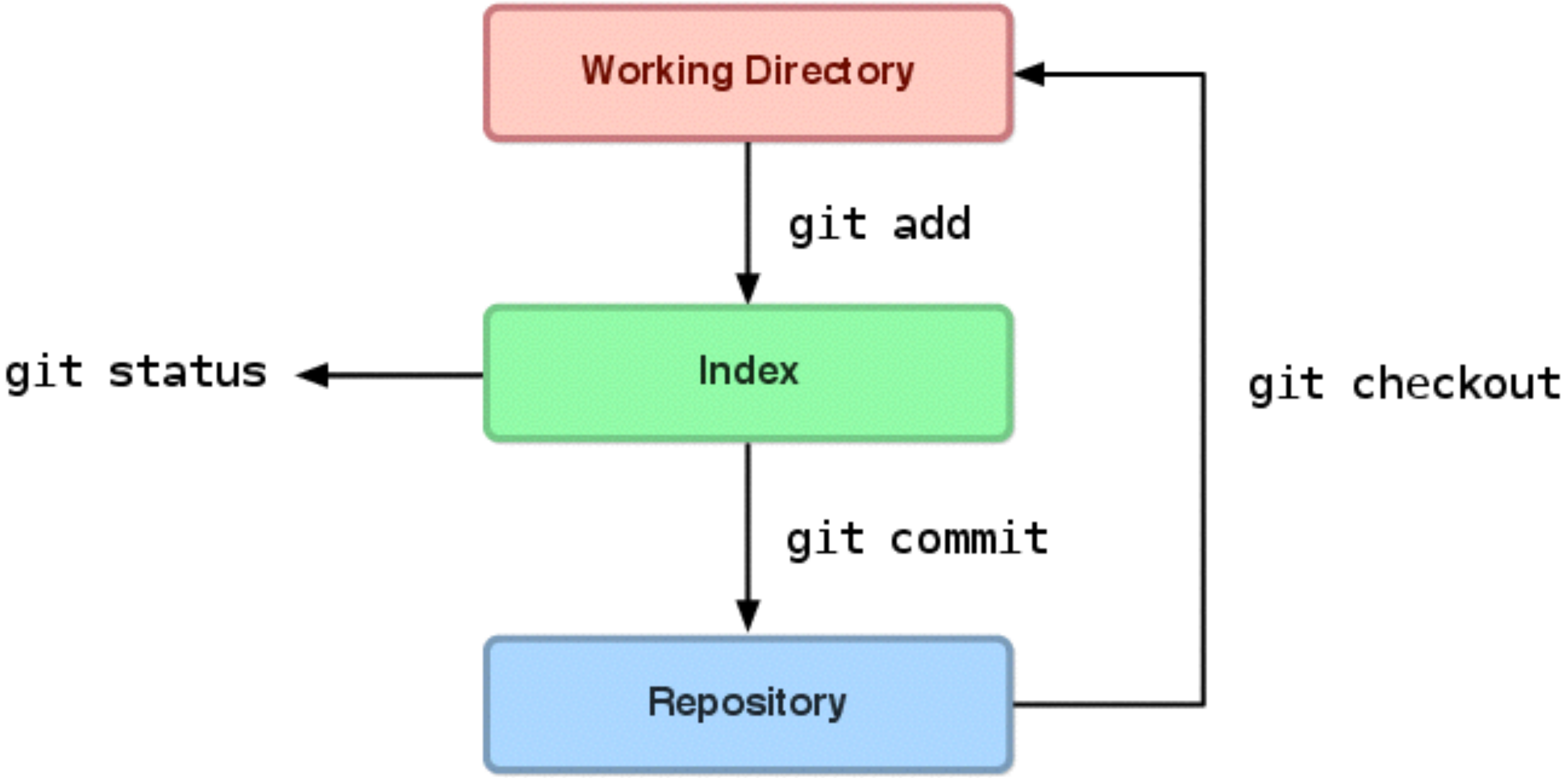
Index

Index

Index

```
== Staging Area
```

Index


Working Directory
git add
git status
Index
git checkout
git commit
Repository

Index FTW

No Need To Commit All At Once

Pick (Stage) Logical Units to Commit

Helps You Review Your Changes

Lets You Write Your History Cleanly