EGit

Eclipse plug-in for git

Tomasz Zarna

Eclipse Platform Workspace committer
IBM Poland

2009-11-28 / Eclipse DemoCamp

Outline

Outline

What is git?
Characteristic, pros and cons

Git is a free distributed revision control

What is git?
Characteristic, pros and cons

Git is a free distributed revision control

- distributed
- high performance
- reliable
- simple design

What is git?
Characteristic, pros and cons

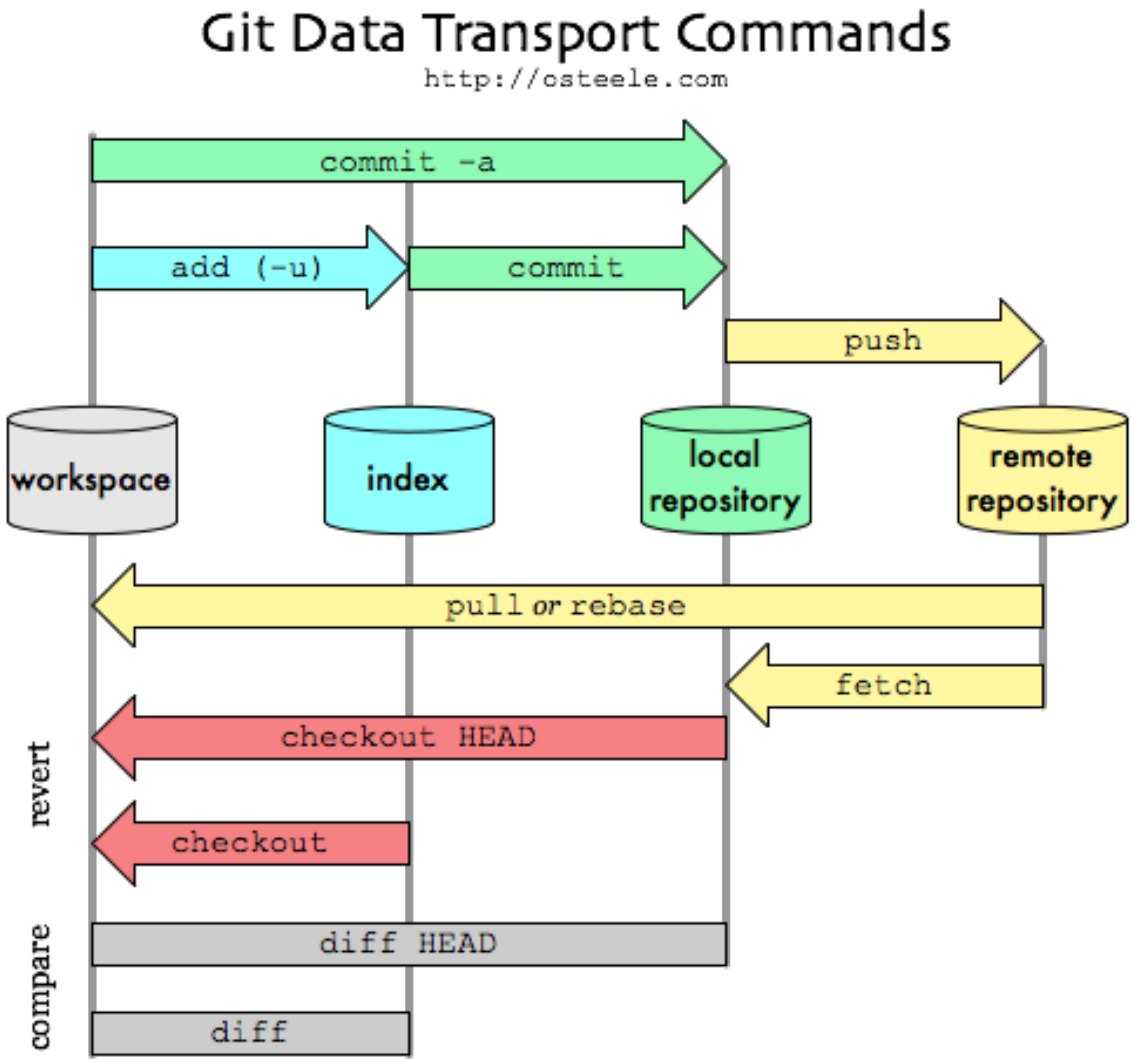
Git is a free distributed revision control

- distributed
- high performance
- reliable
- simple design

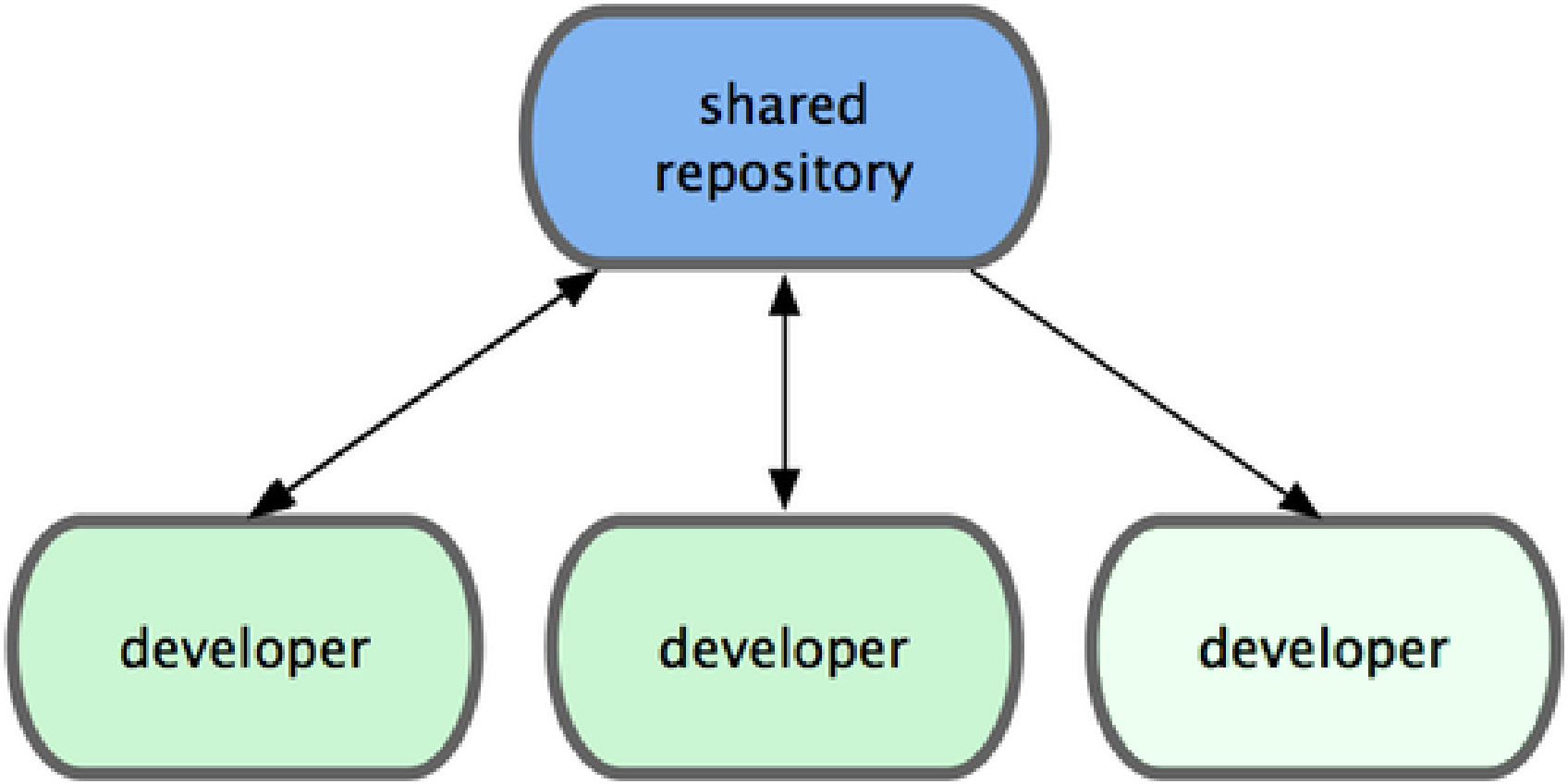
- difficult concepts
- no full control over the history
- bypassing official channels
- no dedicated versions at one time

Transport commands

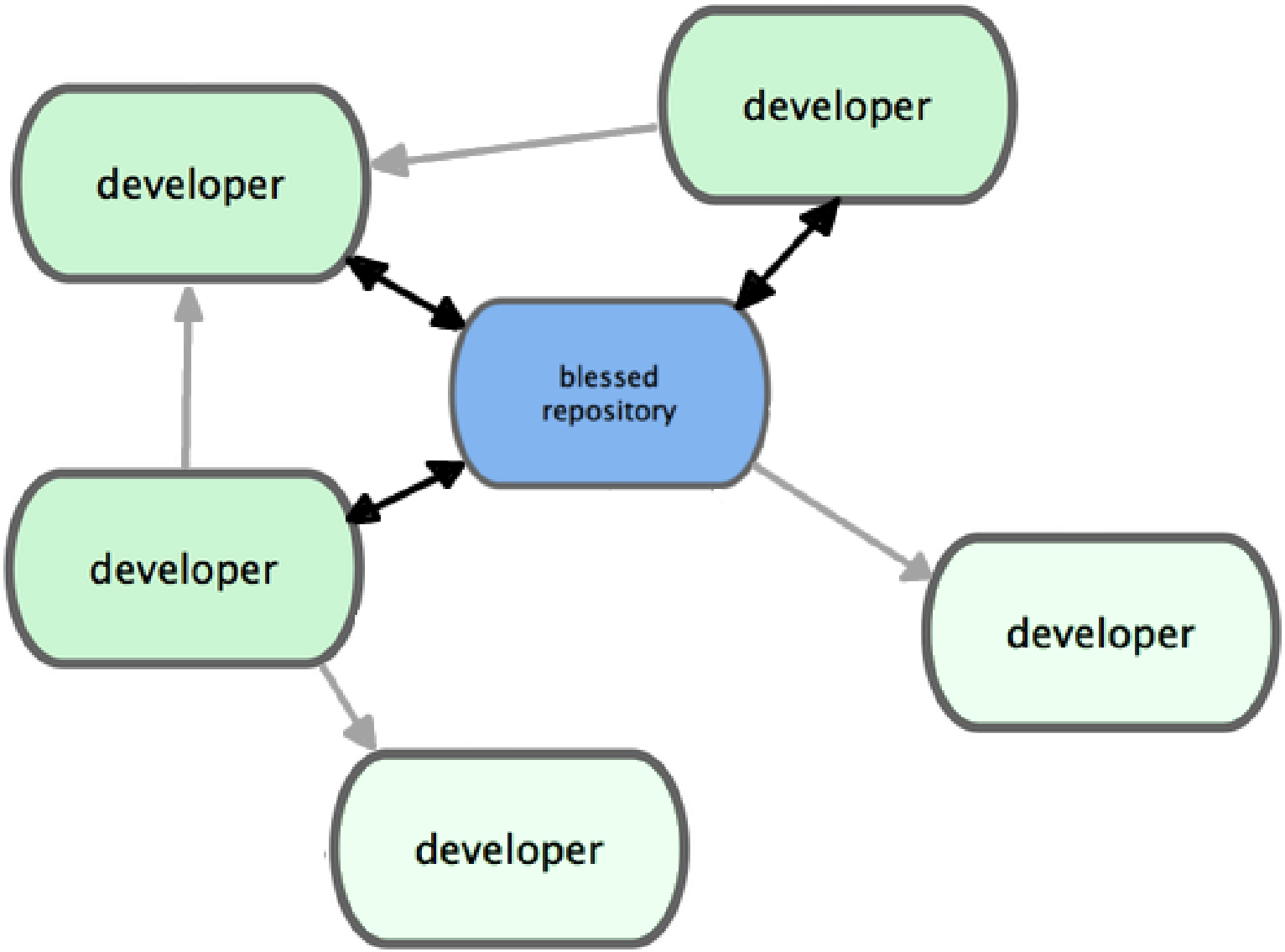
Add, push, pull, fetch. . . WTF?



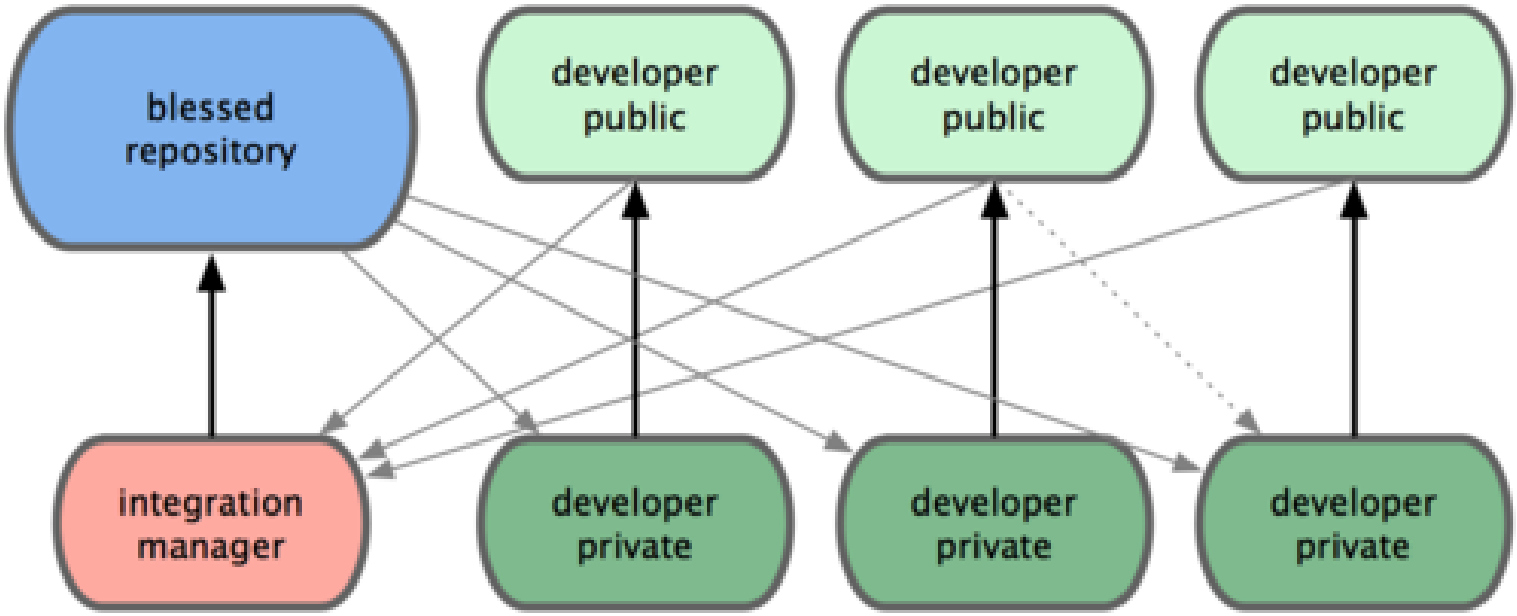
Centralized



Distributied

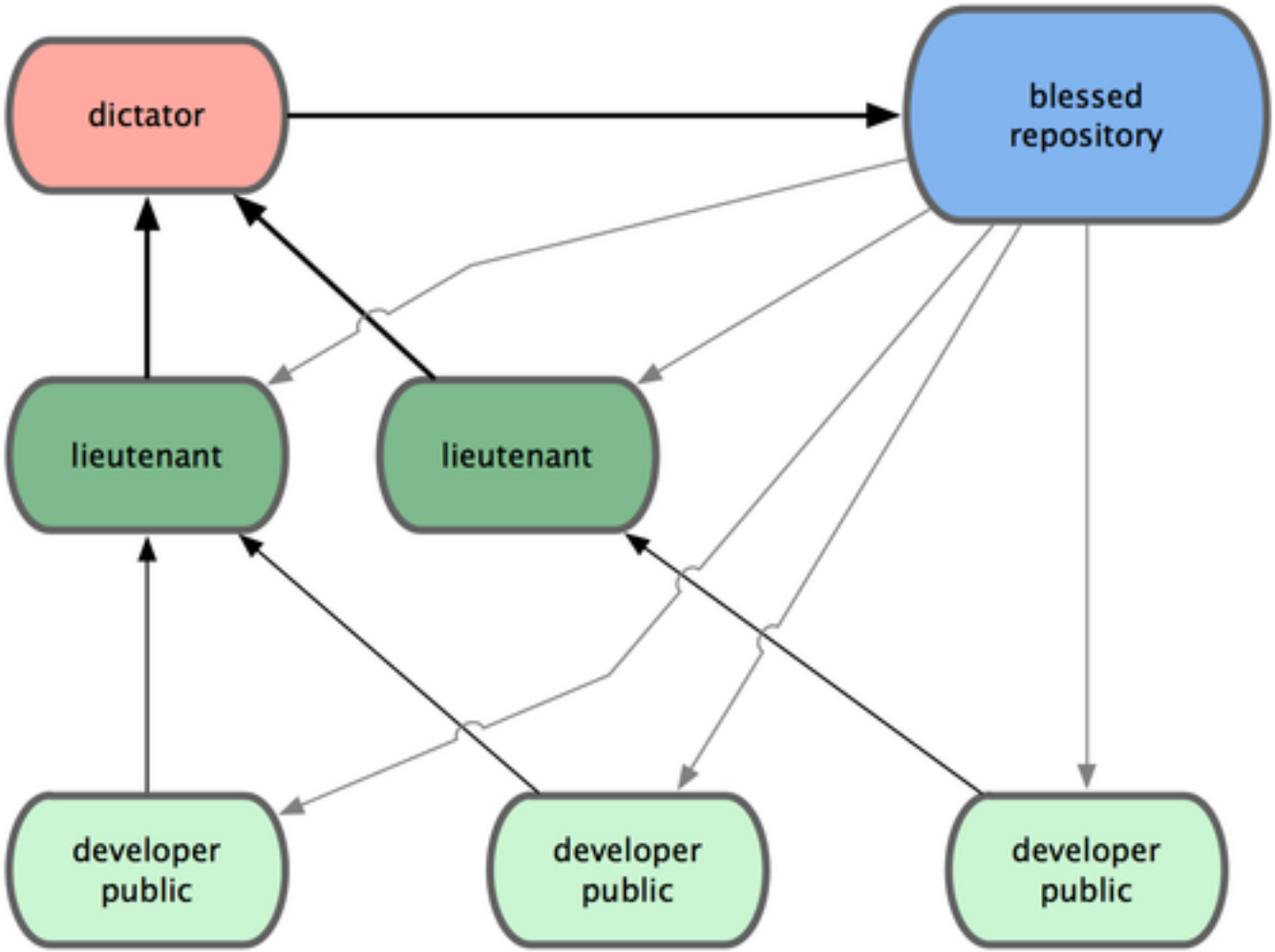


Distributied
Integration manager



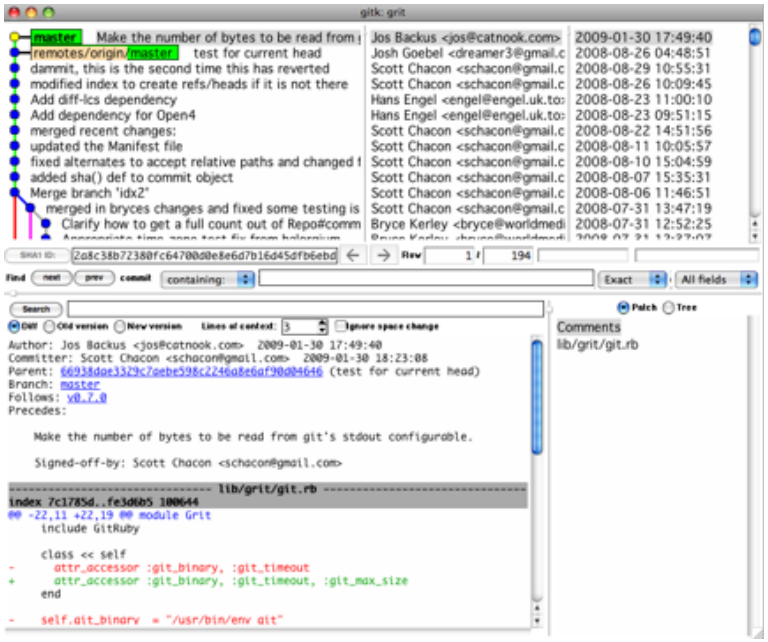
Distributied

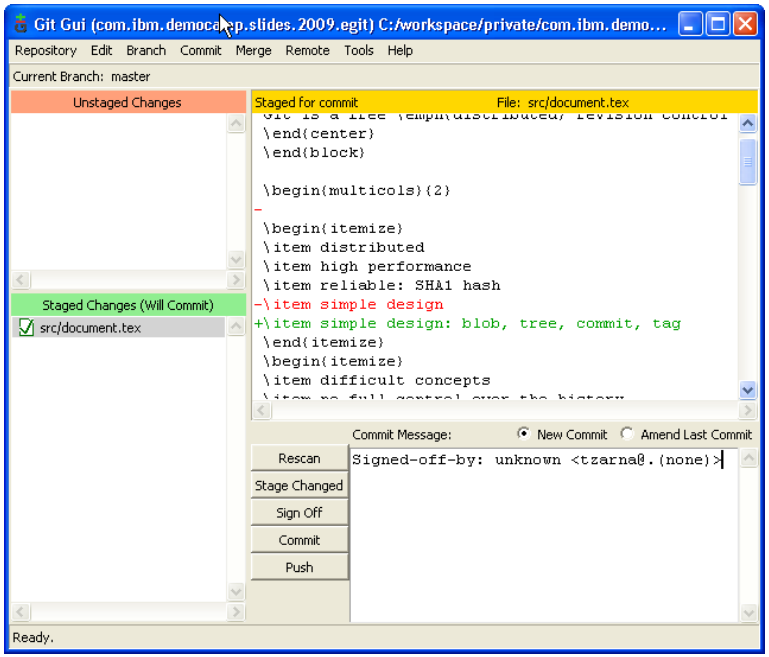
Dicator and lieutenants



gitk and git gui

Not only console





Adopters

- Amarok
- Android
- Digg
- Fedora
- GIMP
- GNOME
- GTK
- Linux kernel
- Perl
- Qt
- Ruby on Rails
- Samba
- VLC
- Wine
- ...

Outline

JGit and EGit

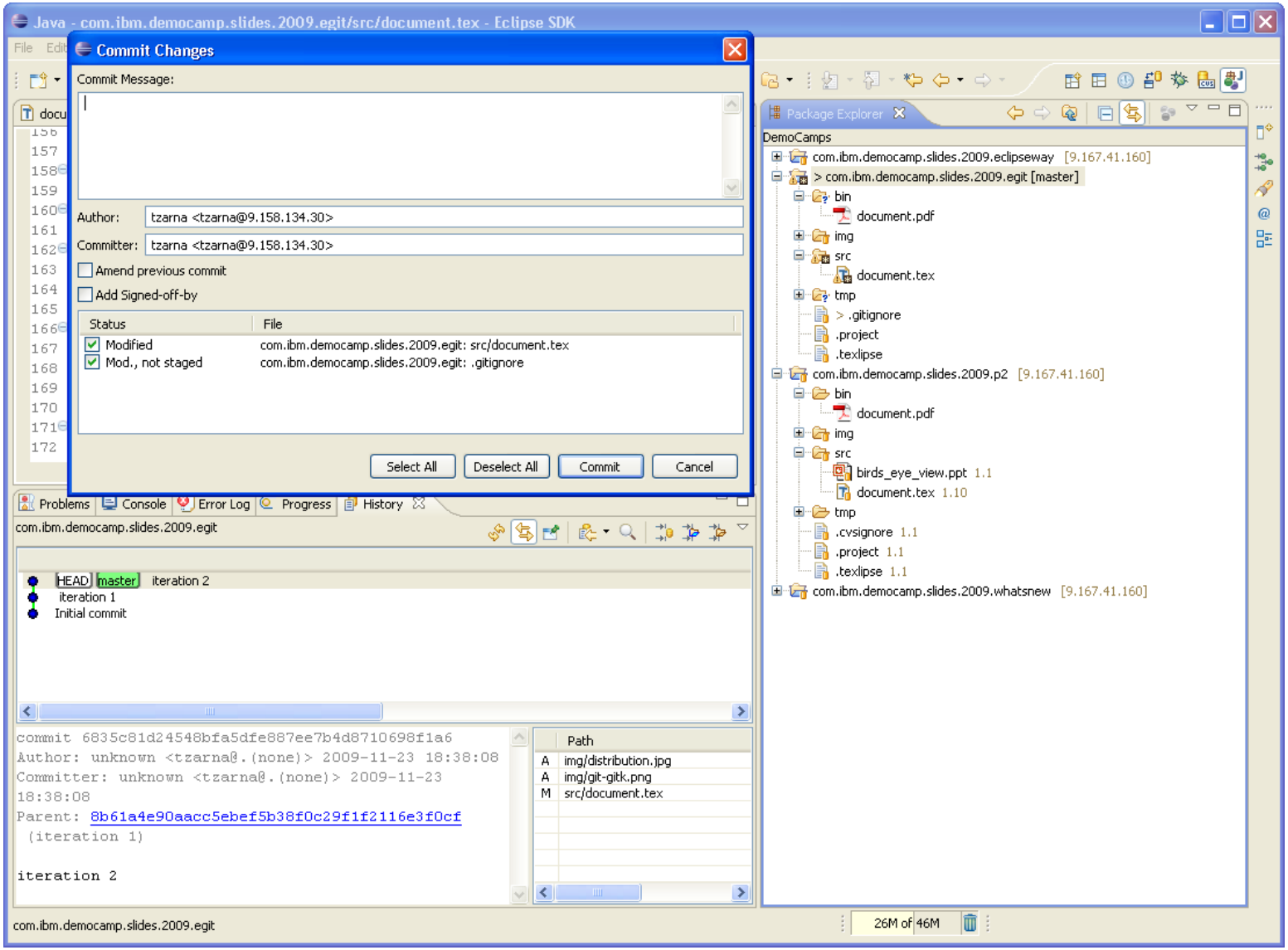
JGit

a pure Java implementation of Git's internals

EGit

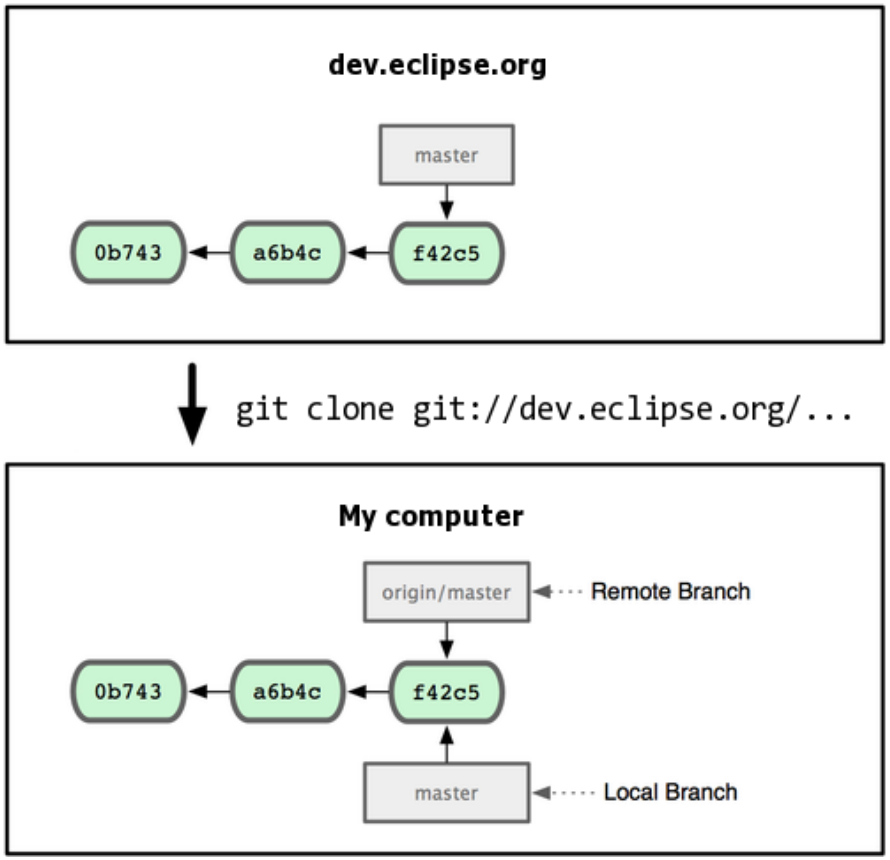
git plug-in for Eclipse

EGit at work



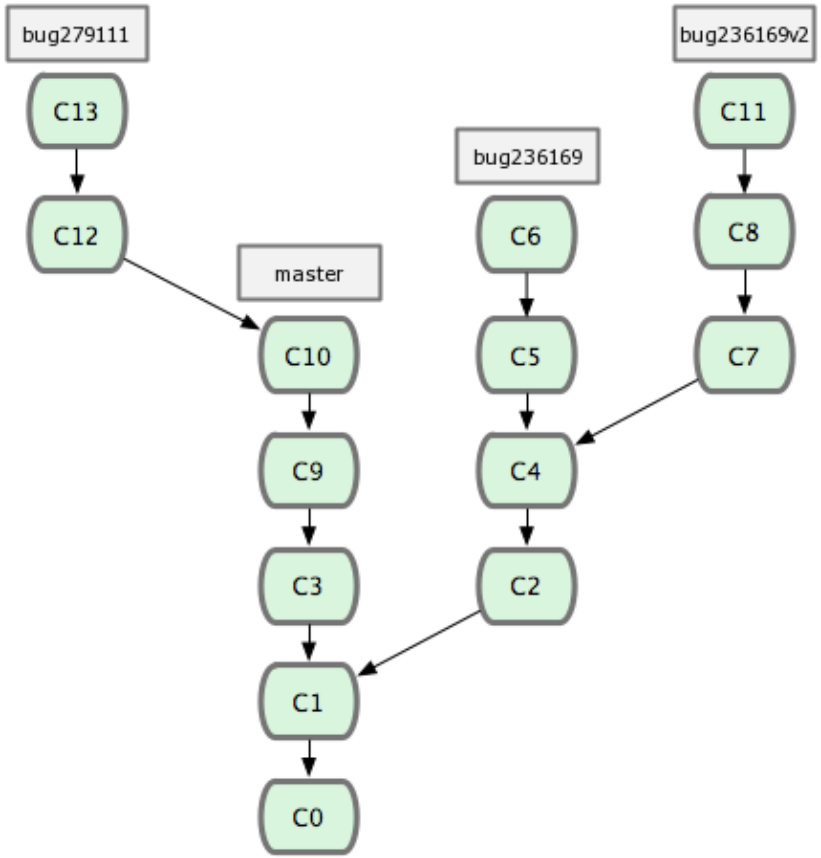
My scenario
Working with Eclipse git repos

1. clone an Eclipse git repo/project
2. branch
3. commit, commit, commit
4. fetch/pull origin
5. merge/rebase
6. create patch: ~~format-patch~~ diff
7. submit the patch
8. go to 2.



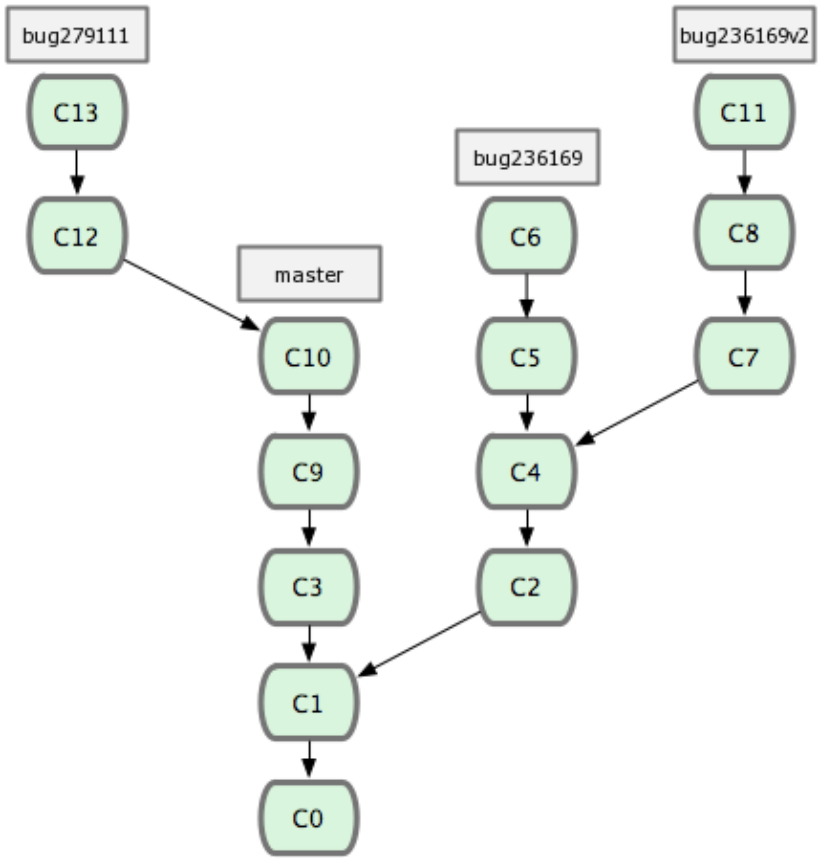
My scenario
Working with Eclipse git repos

1. clone an Eclipse git repo/project
2. branch
3. commit, commit, commit
4. fetch/pull origin
5. merge/rebase
6. create patch: ~~format-patch~~ diff
7. submit the patch
8. go to 2.



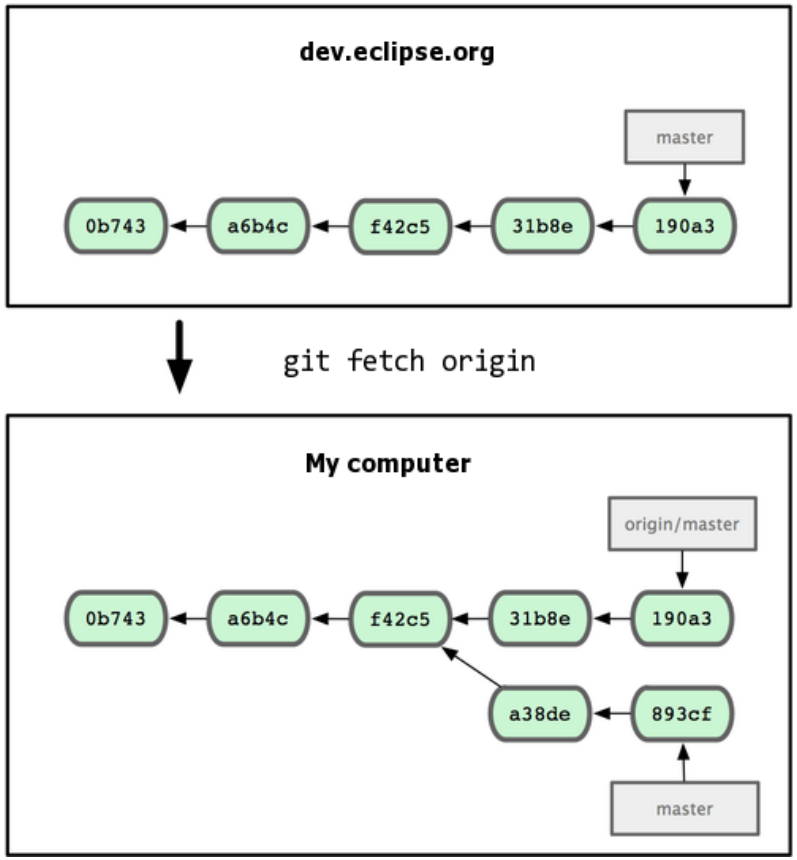
My scenario
Working with Eclipse git repos

1. clone an Eclipse git repo/project
2. branch
3. commit, commit, commit
4. fetch/pull origin
5. merge/rebase
6. create patch: ~~format-patch~~ diff
7. submit the patch
8. go to 2.



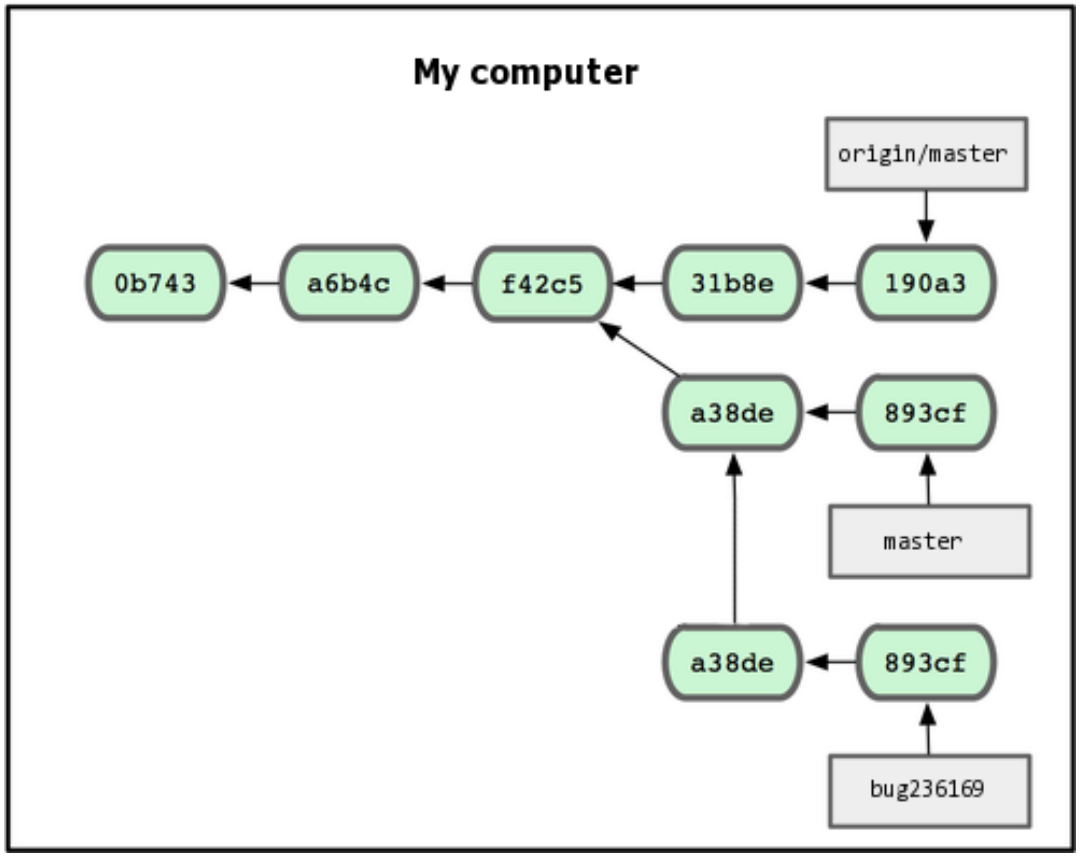
My scenario
Working with Eclipse git repos

1. clone an Eclipse git repo/project
2. branch
3. commit, commit, commit
4. fetch/pull origin
5. merge/rebase
6. create patch: ~~format-patch~~ diff
7. submit the patch
8. go to 2.



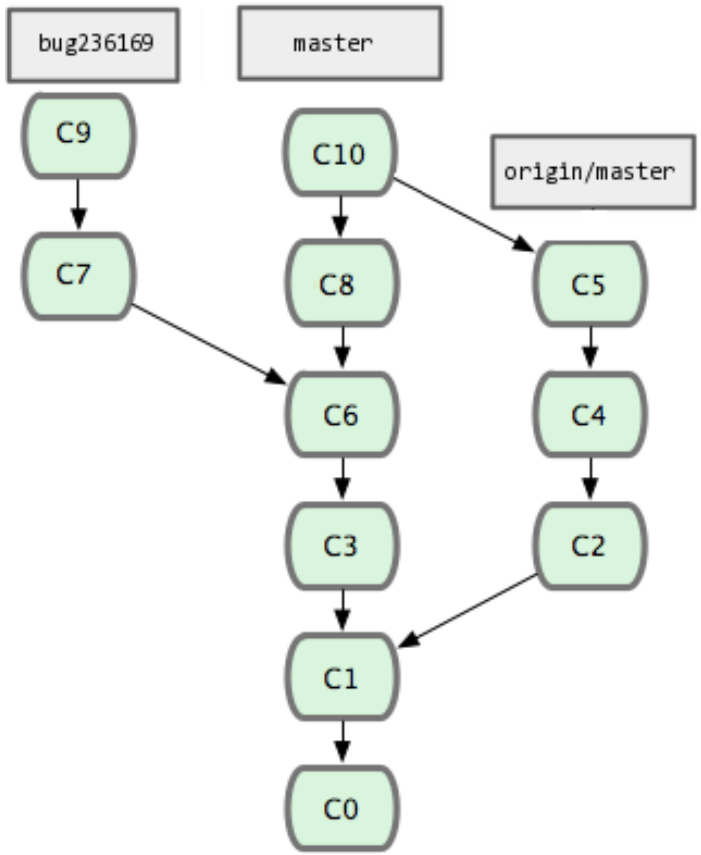
My scenario
Working with Eclipse git repos

1. clone an Eclipse git repo/project
2. branch
3. commit, commit, commit
4. fetch/pull origin
5. merge/rebase
6. create patch: ~~format-patch~~ diff
7. submit the patch
8. go to 2.



My scenario
Working with Eclipse git repos

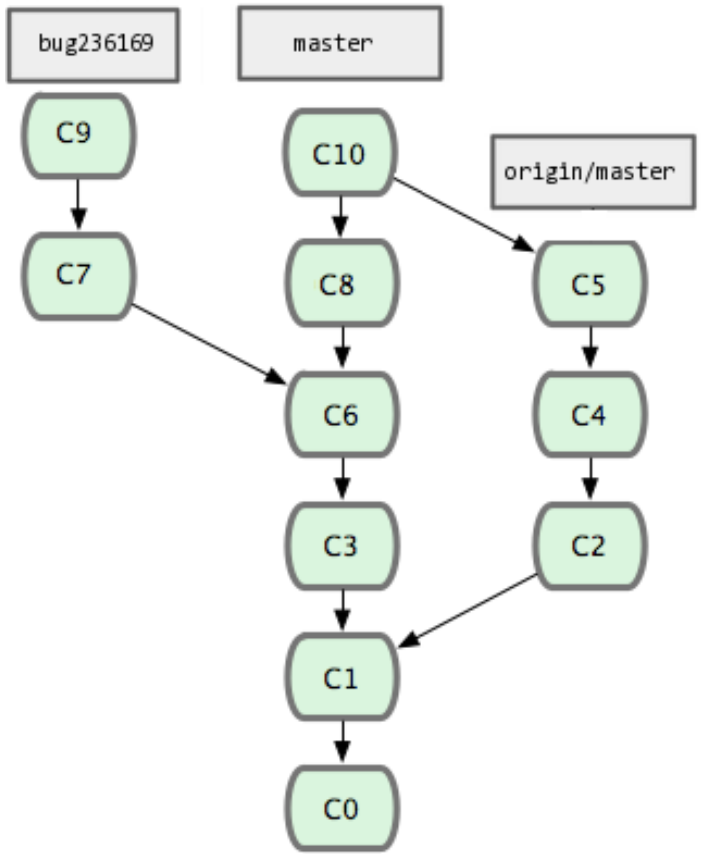
1. clone an Eclipse git repo/project
2. branch
3. commit, commit, commit
4. fetch/pull origin
5. merge/rebase
6. create patch: ~~format-patch~~ diff
7. submit the patch
8. go to 2.



My scenario

Working with Eclipse git repos

1. clone an Eclipse git repo/project
2. branch
3. commit, commit, commit
4. fetch/pull origin
5. merge/rebase
6. create patch: ~~format-patch~~ diff
7. submit the patch
8. go to 2.



My scenario
Working with Eclipse git repos

1. clone an Eclipse git repo/project
2. branch
3. commit, commit, commit
4. fetch/pull origin
5. merge/rebase
6. create patch: ~~format-patch~~ diff
7. submit the patch
8. go to 2.

Outline

Links

EGit links, git links

- Home page
- Update site
- Contribution guide
- Depracate old VCS tools bug
- Eclipse git repositories

Links

EGit links, git links

- Home page
- Update site
- Contribution guide
- Depracate old VCS tools bug
- Eclipse git repositories

- git home
- Pro Git
- msysgit, git on Windows
- GitHub
- git community book
- git ready

Thank you