# **Collaborative** Development & **Source Code** Versioning

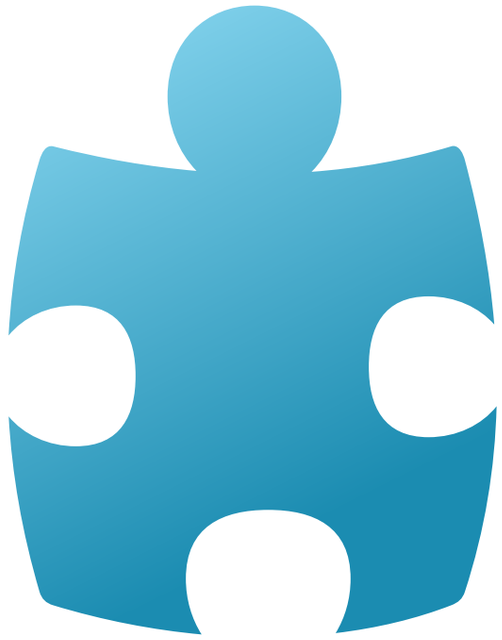
**Sébastien Mosser**

Lecture #2, 20.09.2017

pictures: sxc.hu & C.Line

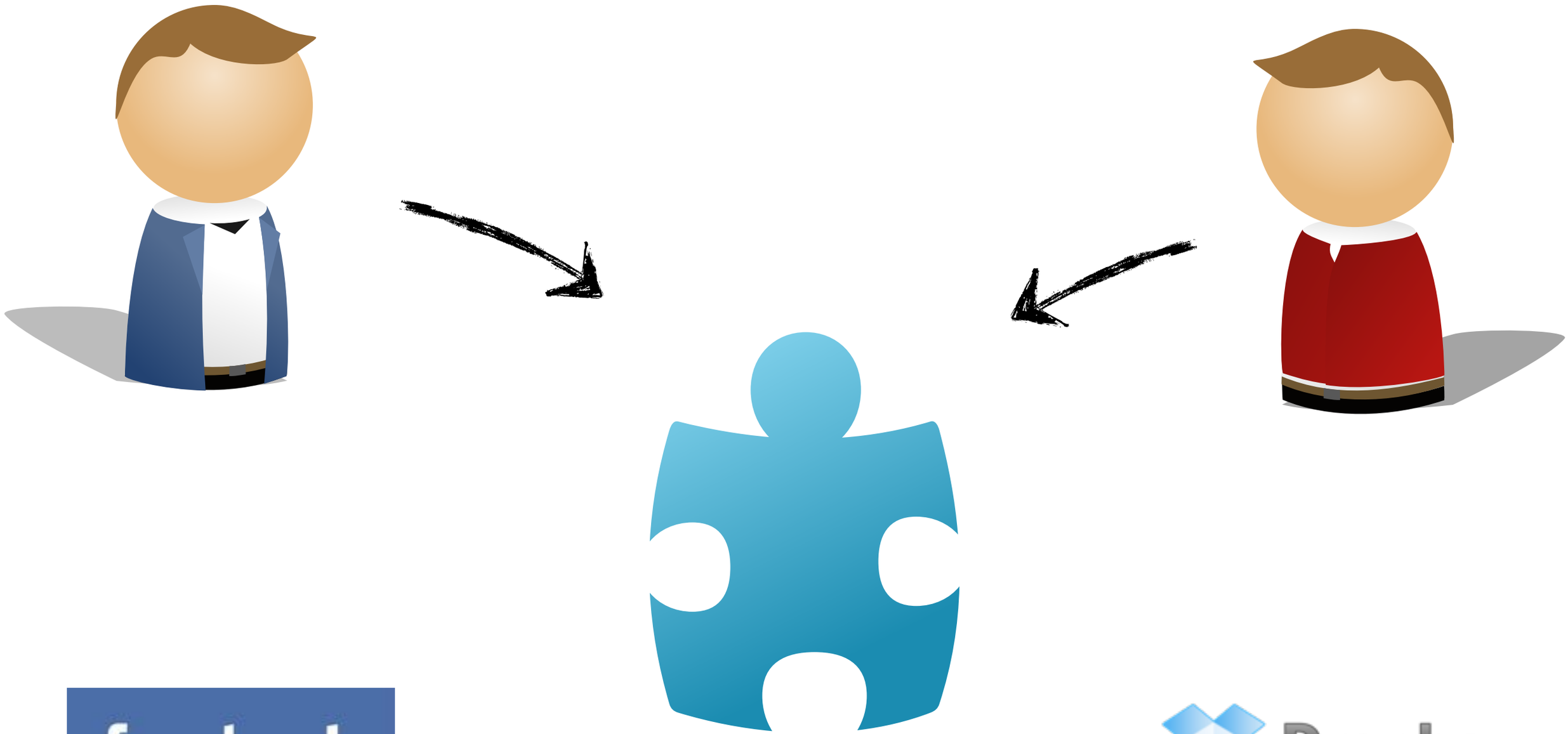
developer

works on



piece of software

# Collaborative Development



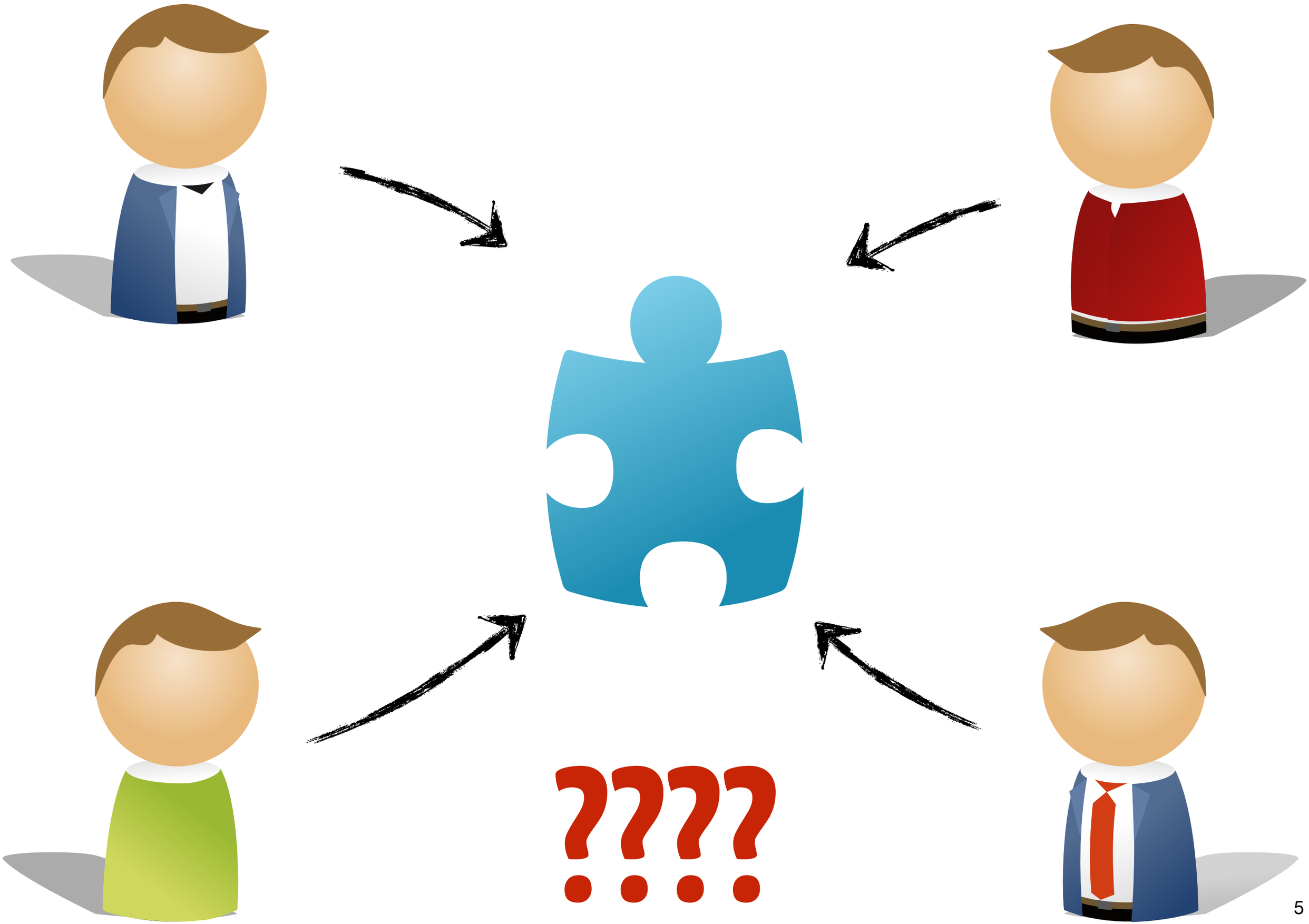
facebook

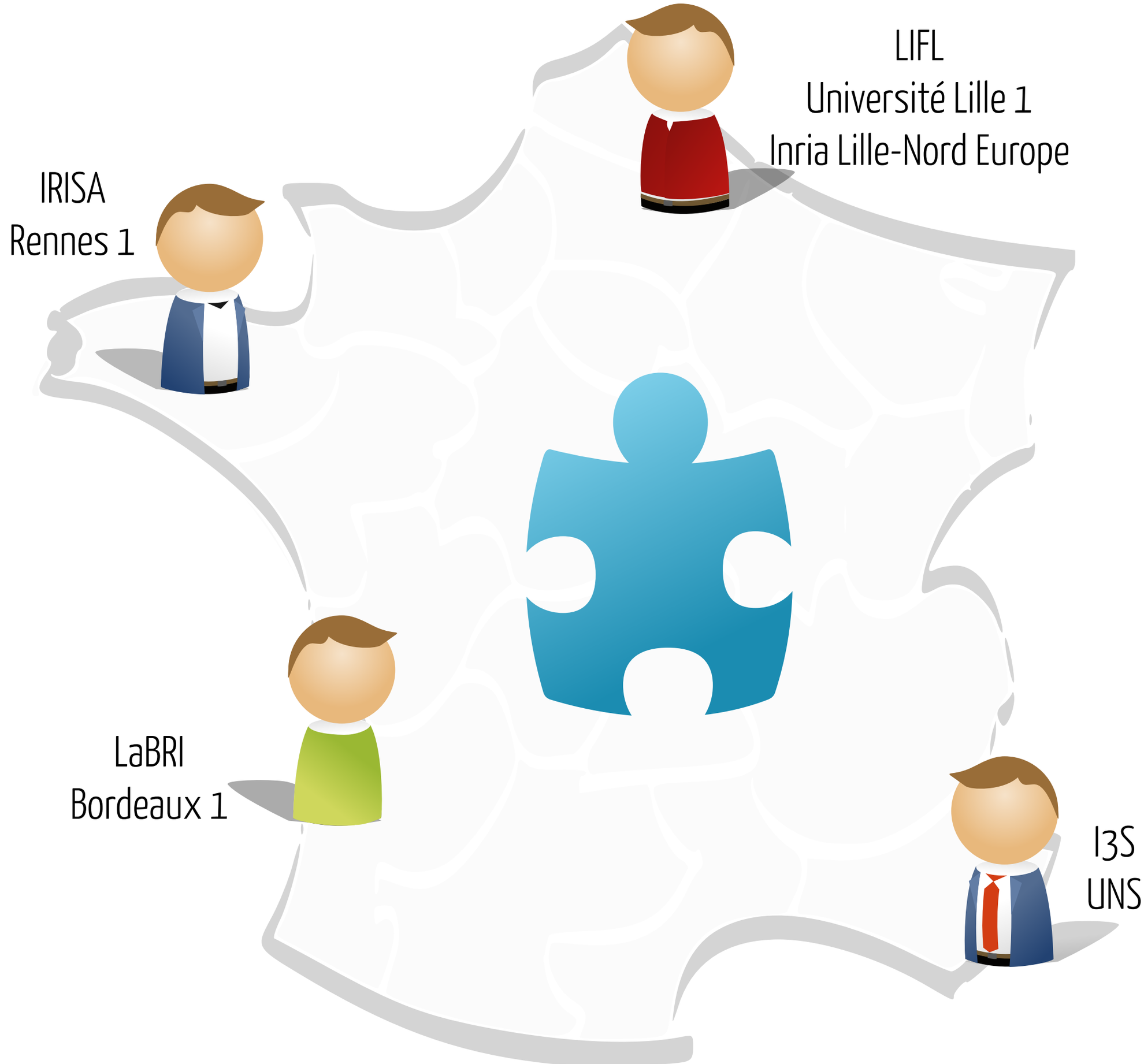
Dropbox

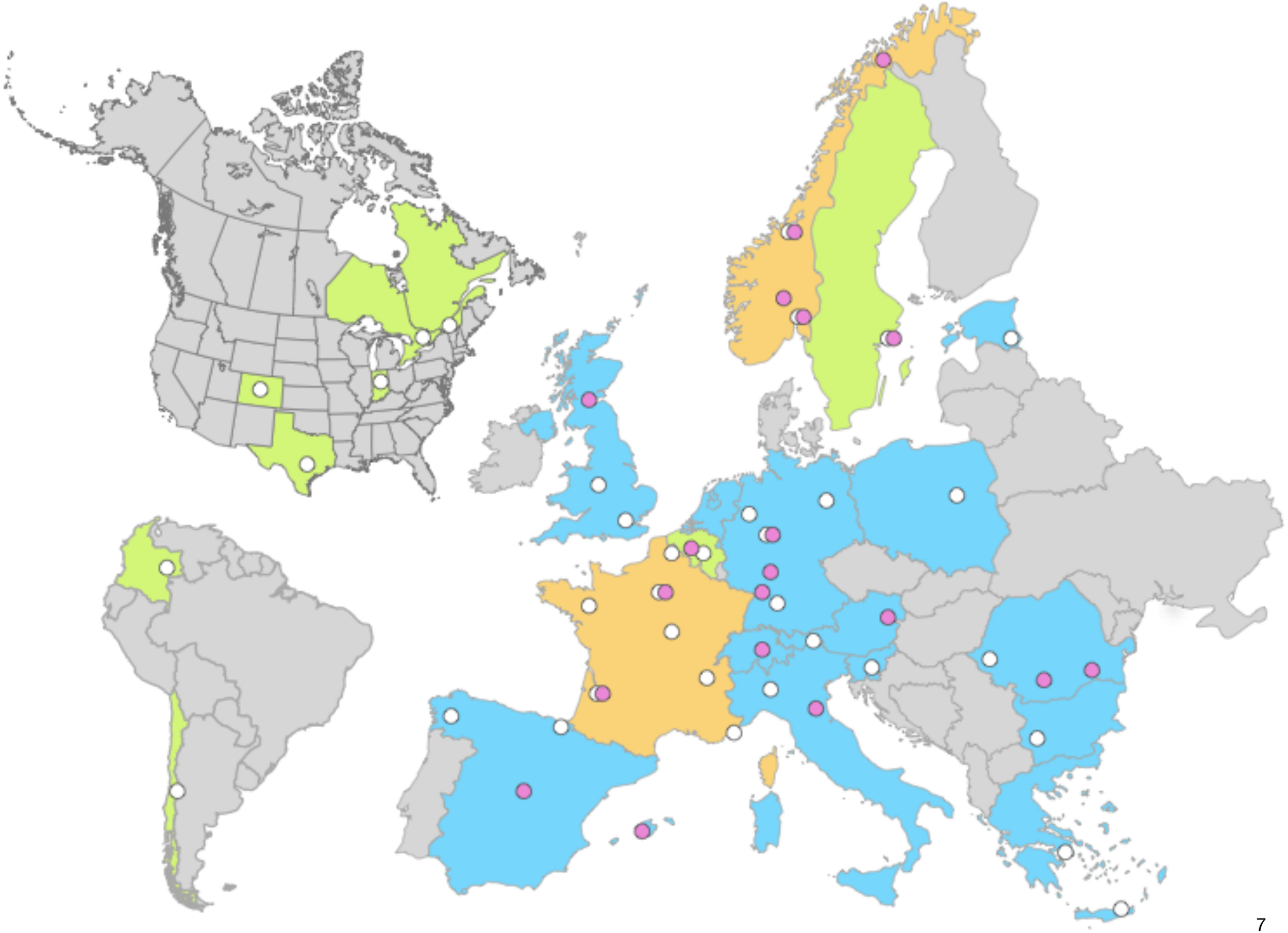
USB Key?

Email?

Shared directory?




LIFL
Université Lille 1
Inria Lille-Nord Europe
IRISA
Rennes 1
LaBRI
Bordeaux 1
I3S
UNS




# «**Why** do we version source code?»

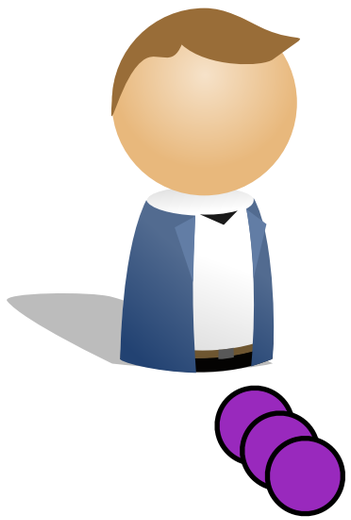
## Motivations (among others)

# Before

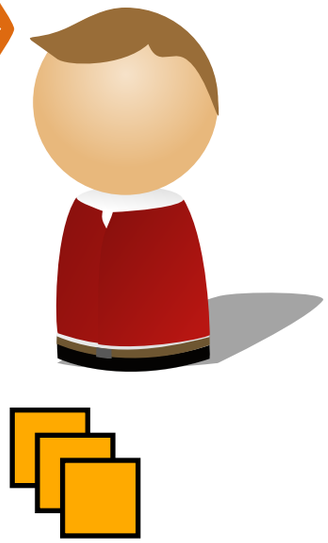
# During

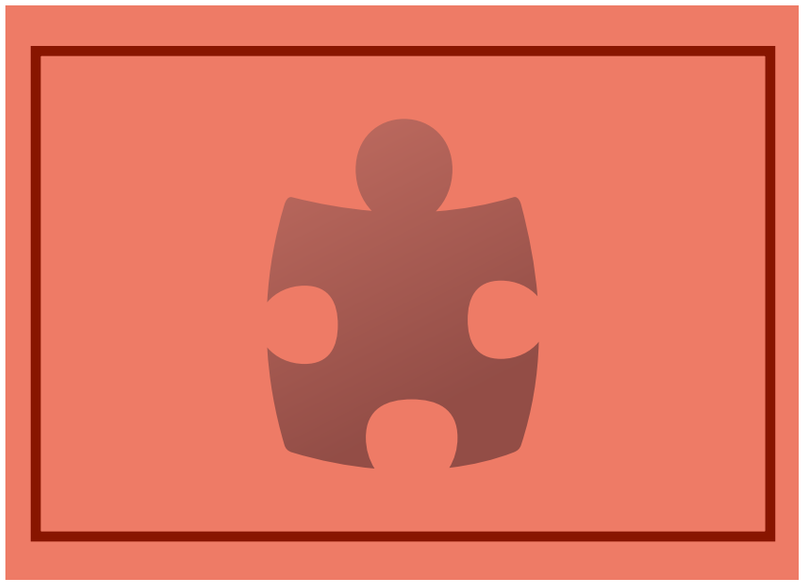
# After

«not me!»



«not me!»





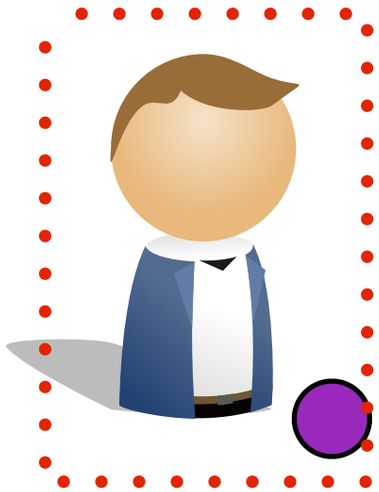
«not me!»

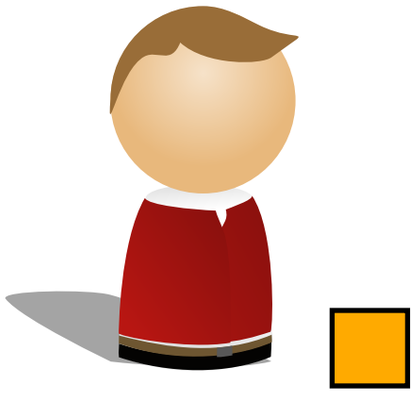
# BUG!

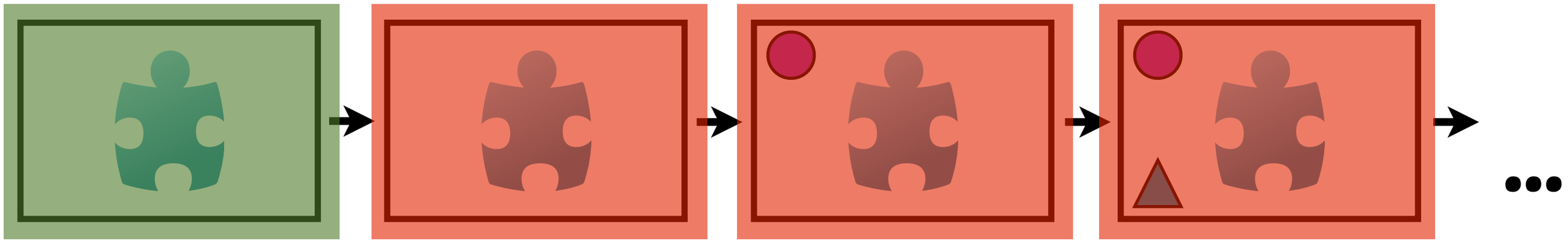
«not me!»

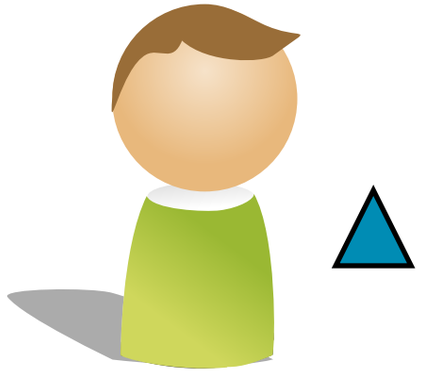
«**Why** do we version source code?»

To **trace** changes!





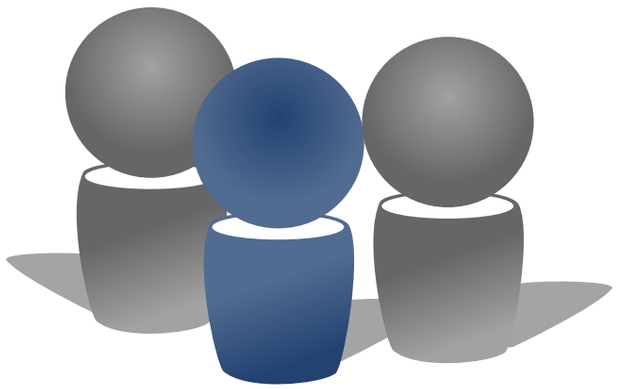


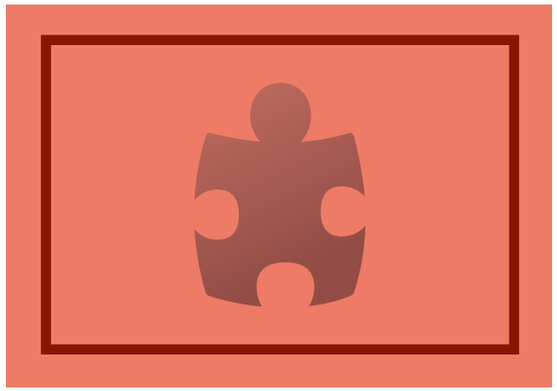


BUG!

Here is the new release!

???





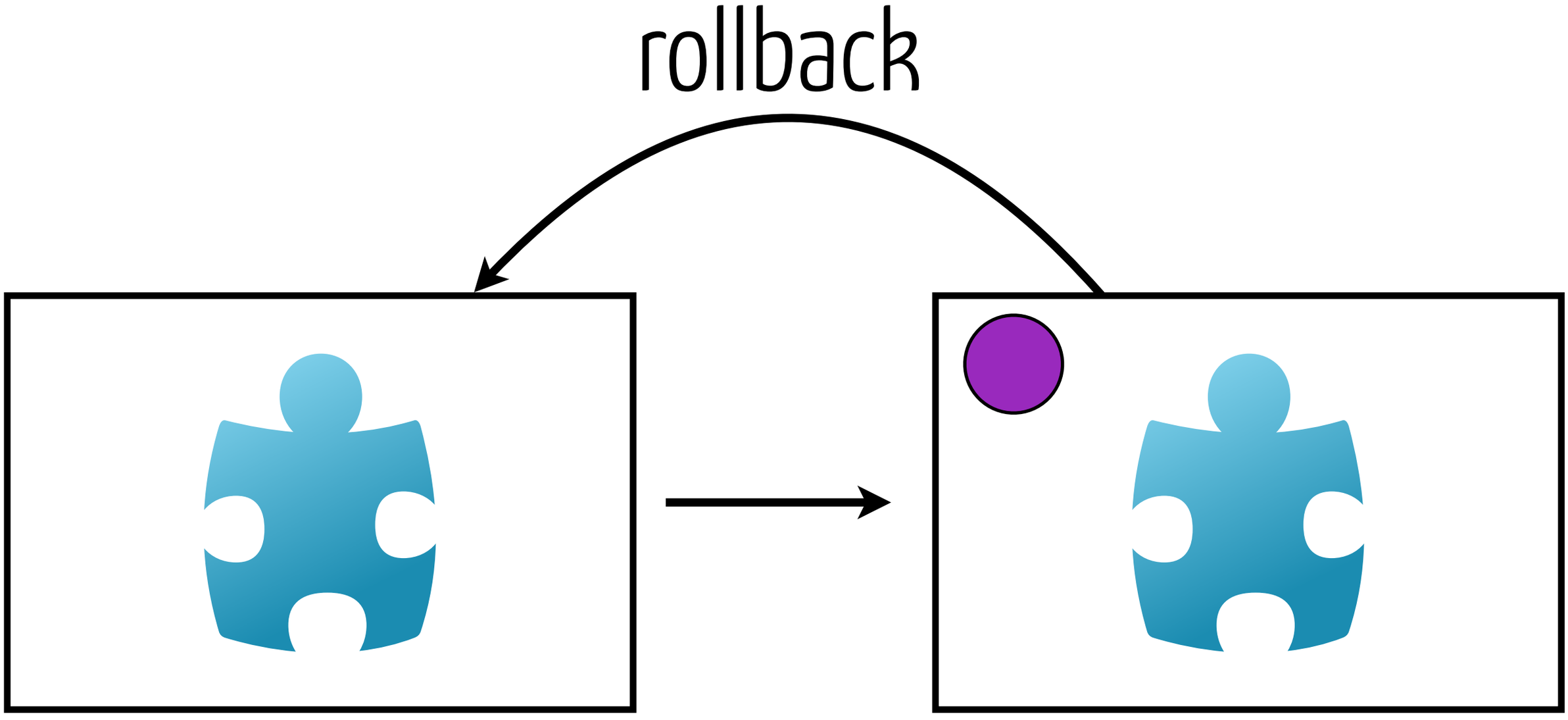
BUG!

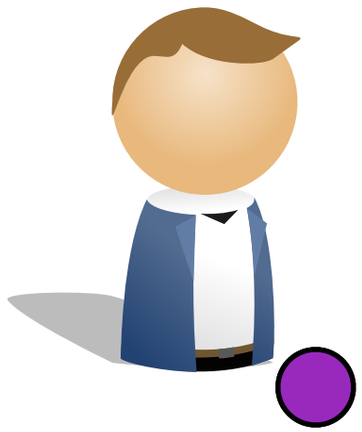
It was working
2 days ago...

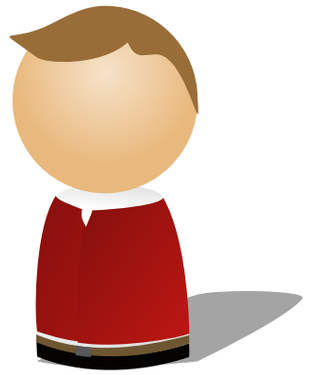
Can I see it?

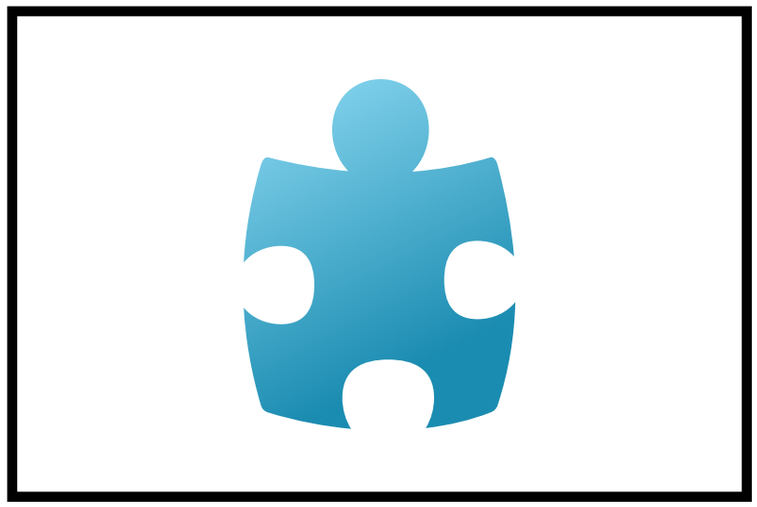
«**Why** do we version source code?»

To **rollback** changes!


rollback





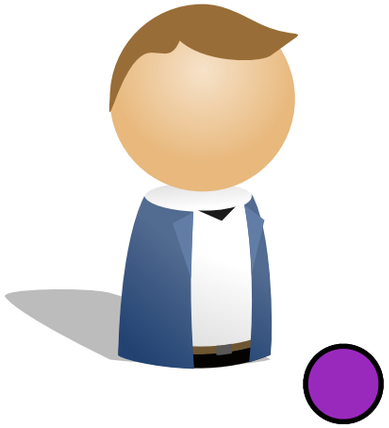


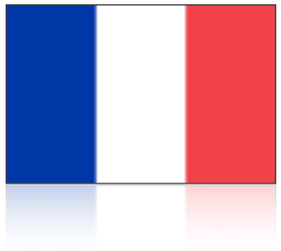
???

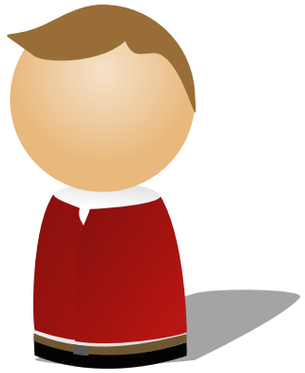
???

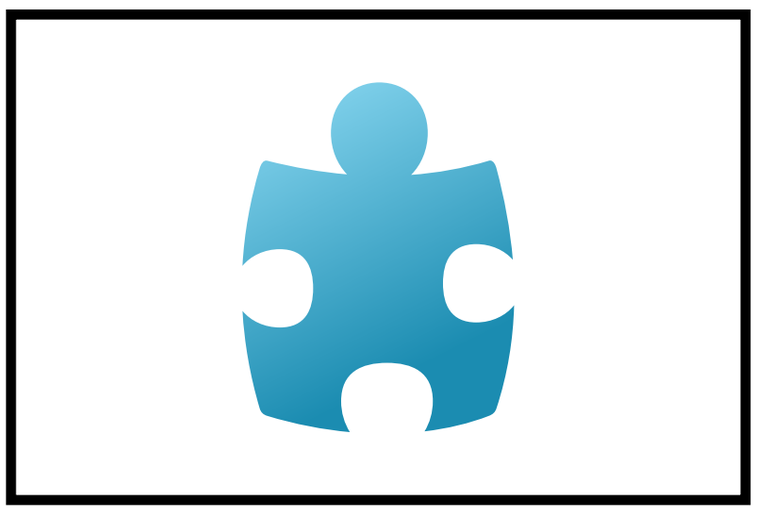
# «**Why** do we version source code?»

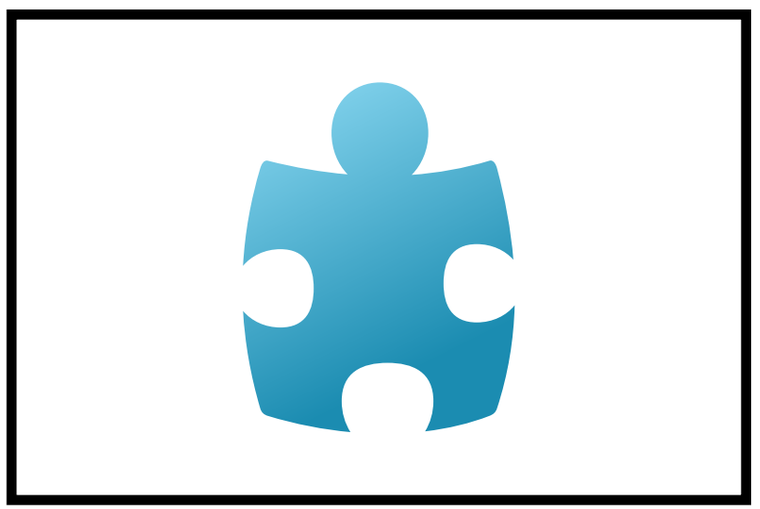
# To **share** changes!







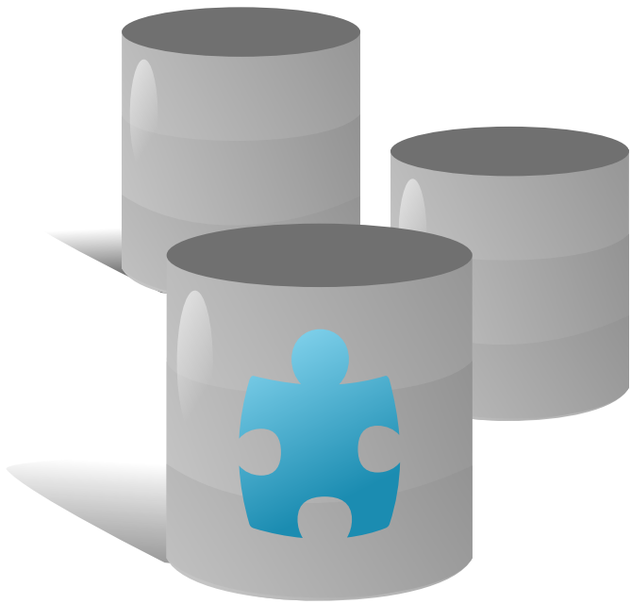


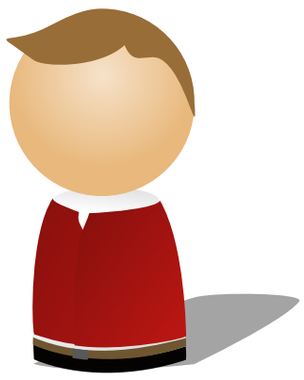


# Centralized Model

(e.g., CVS, Subversion)

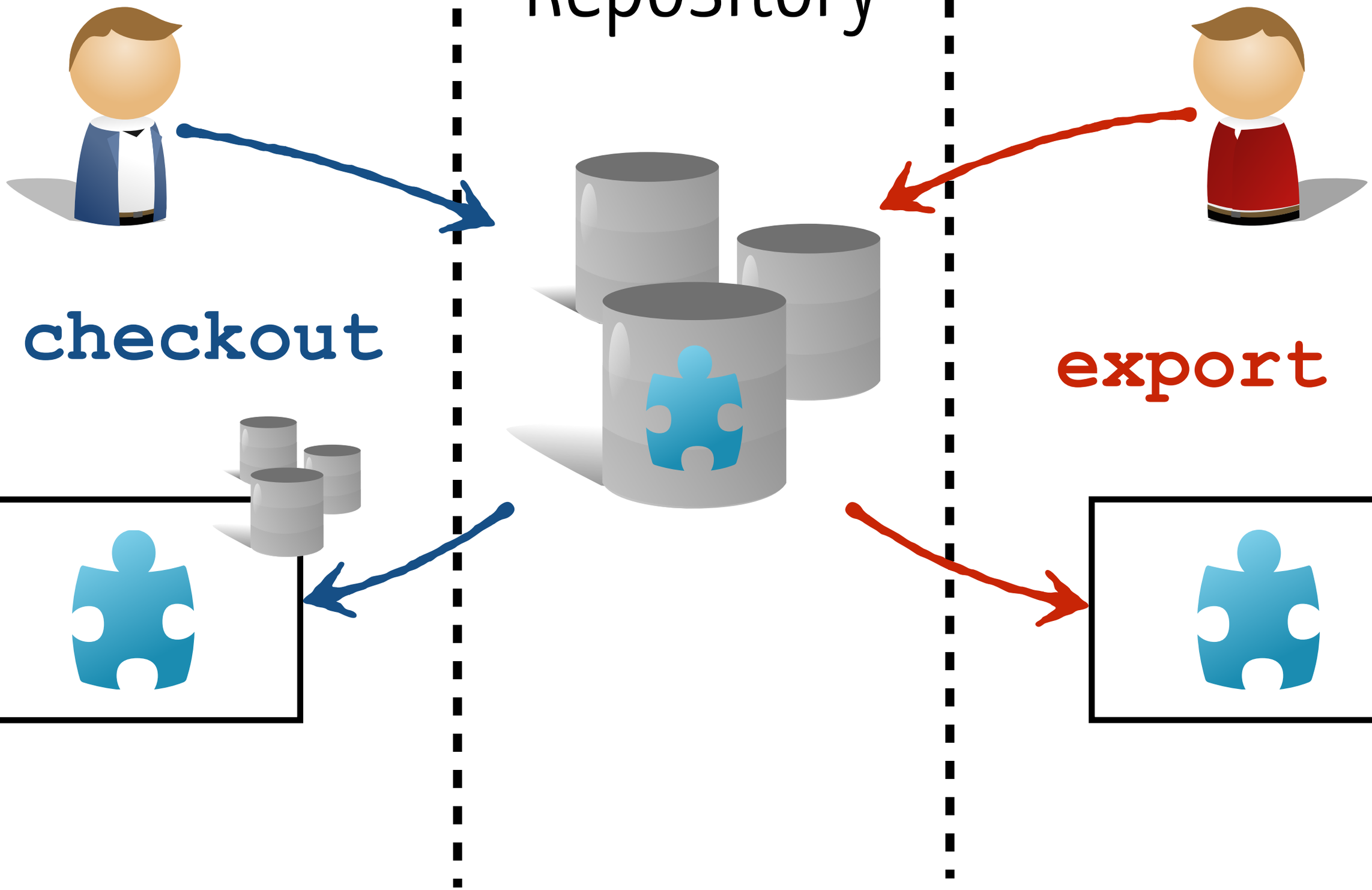
# Shared Repository



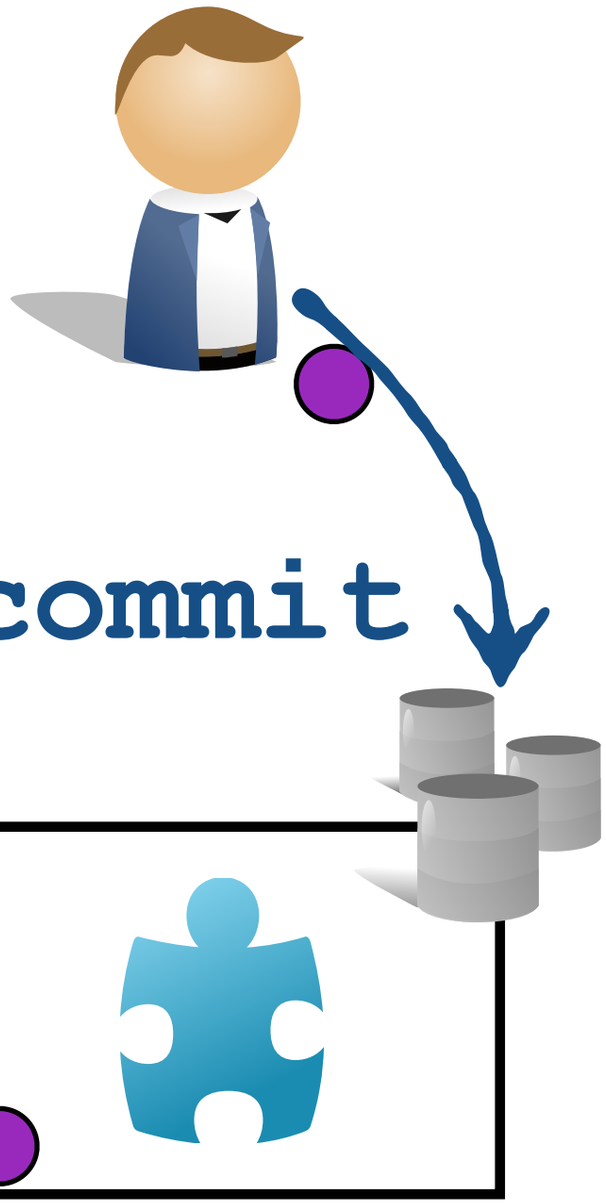


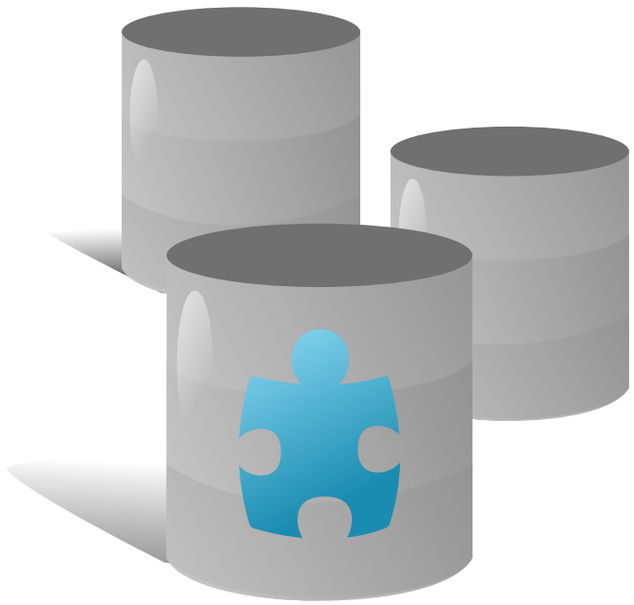
`create`

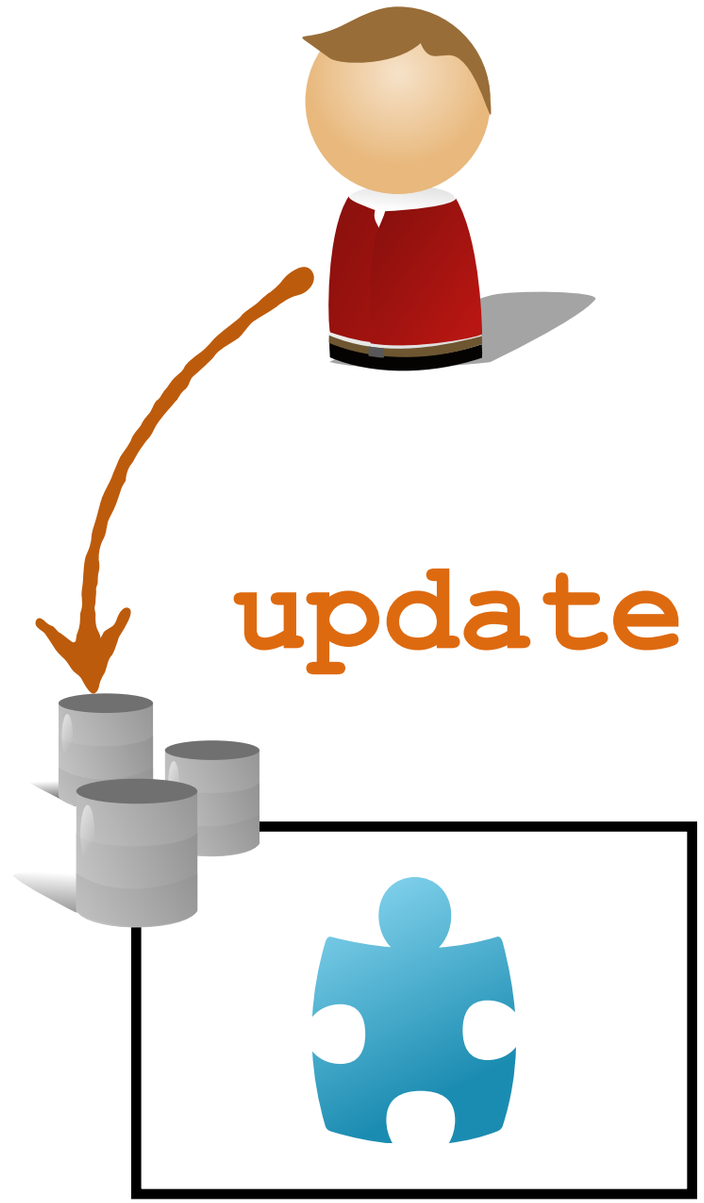
# Shared Repository



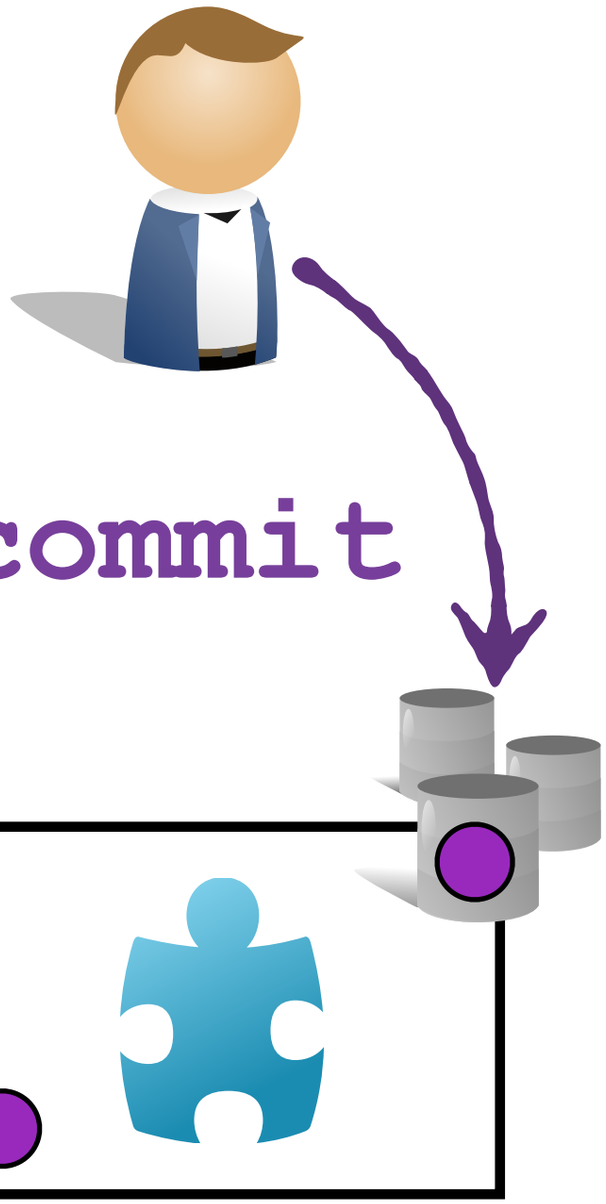
# Shared Repository

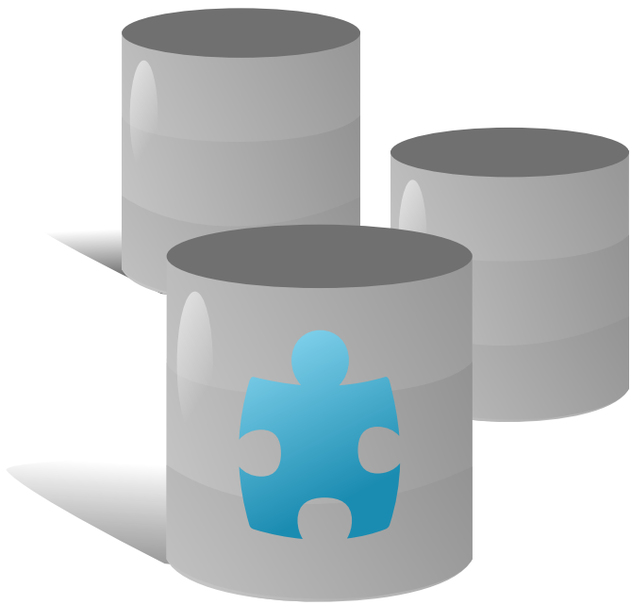


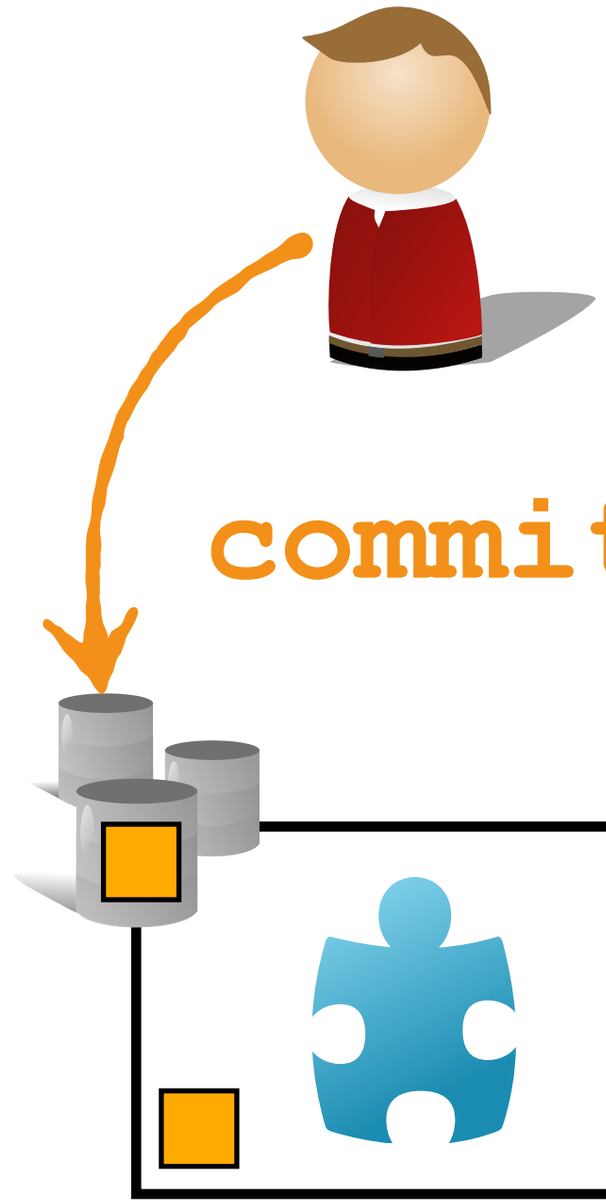


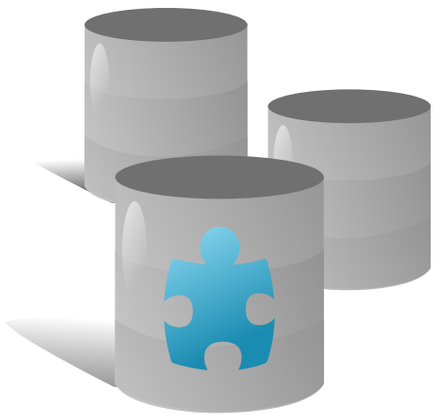
# Shared Repository



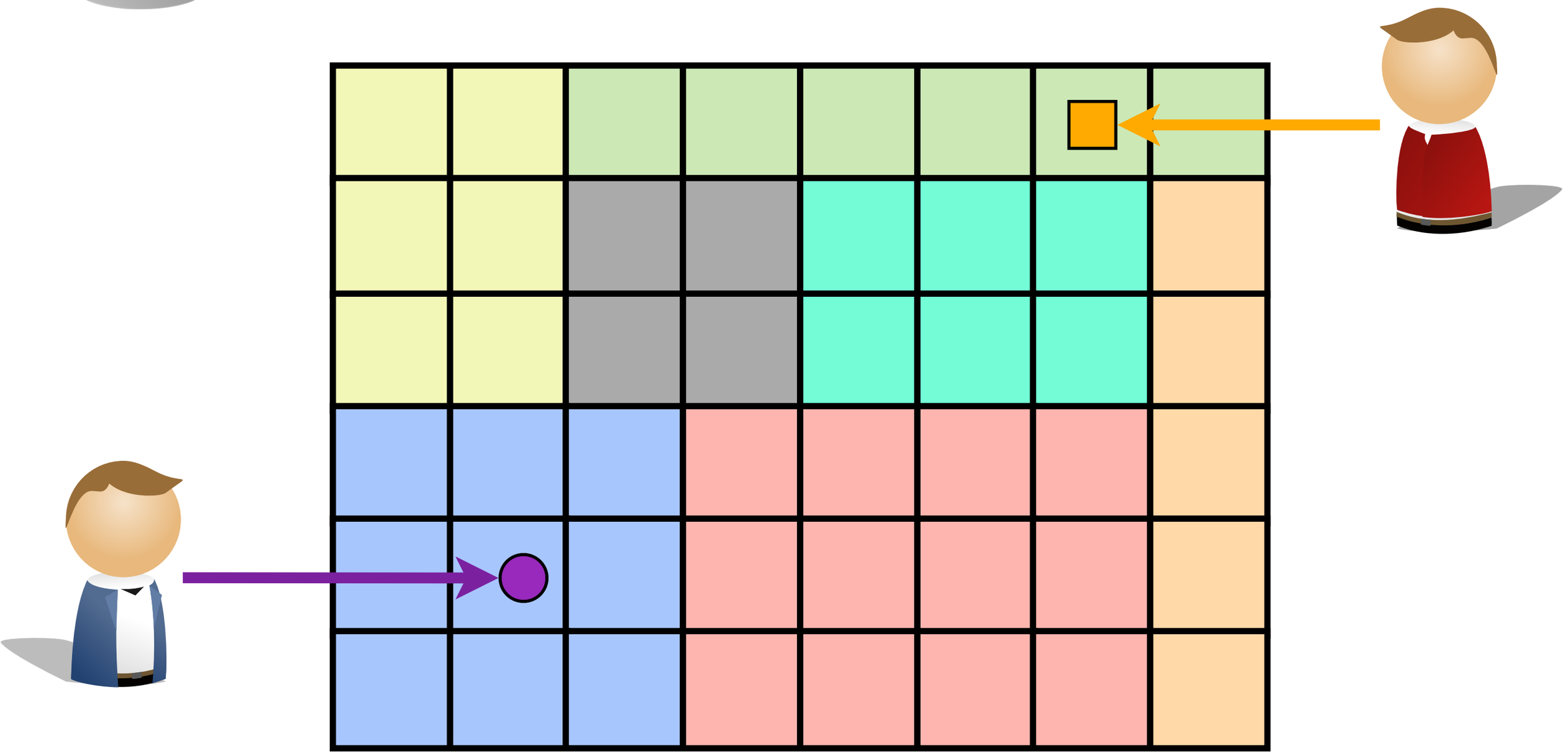



????

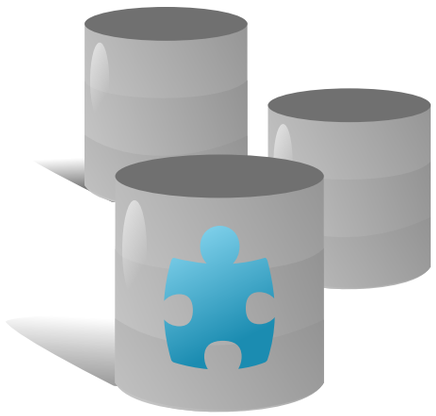




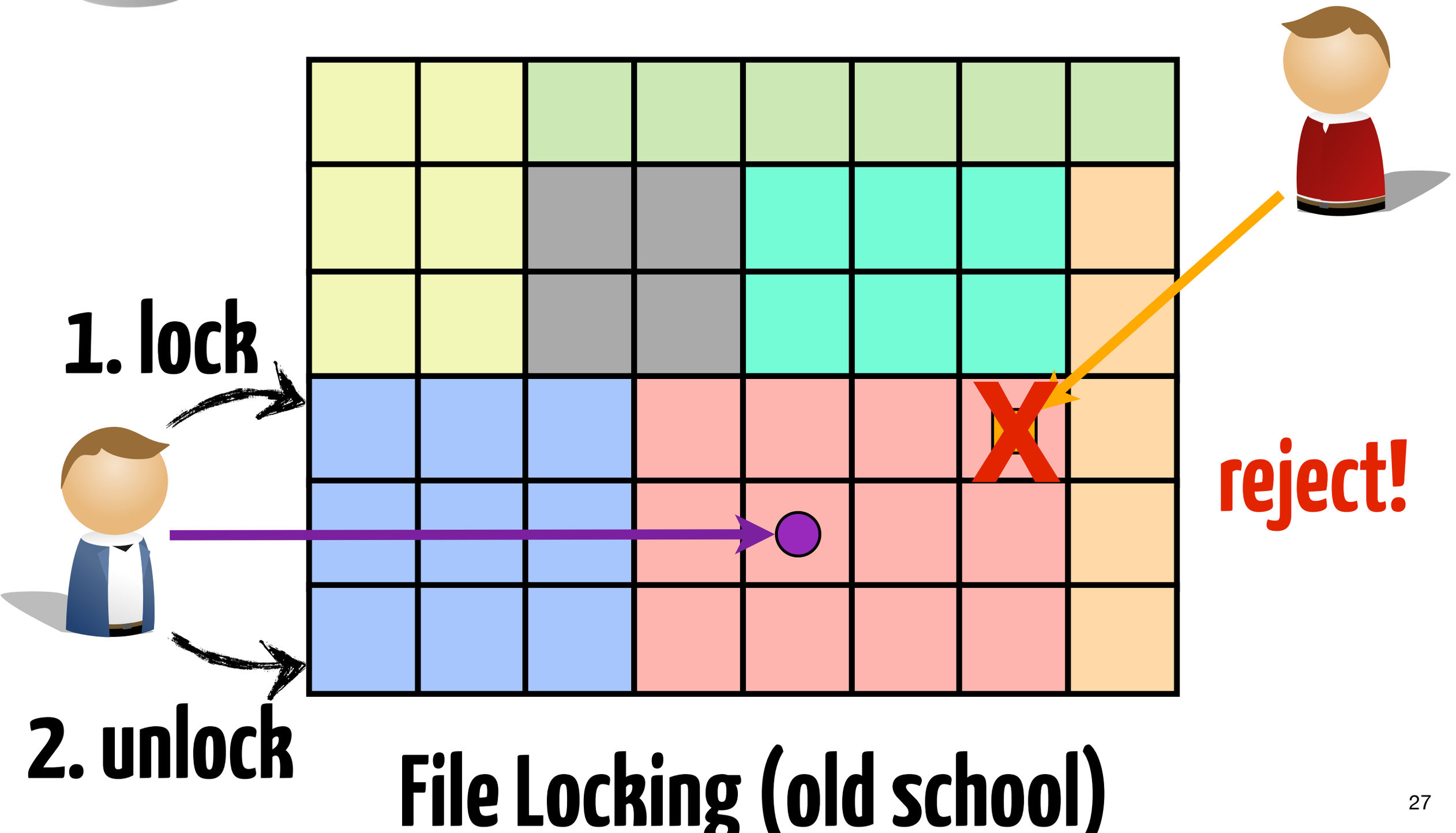
# #1: different files



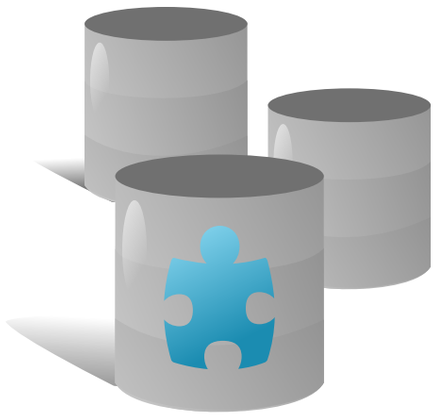
Atomic operations. No problem at all!



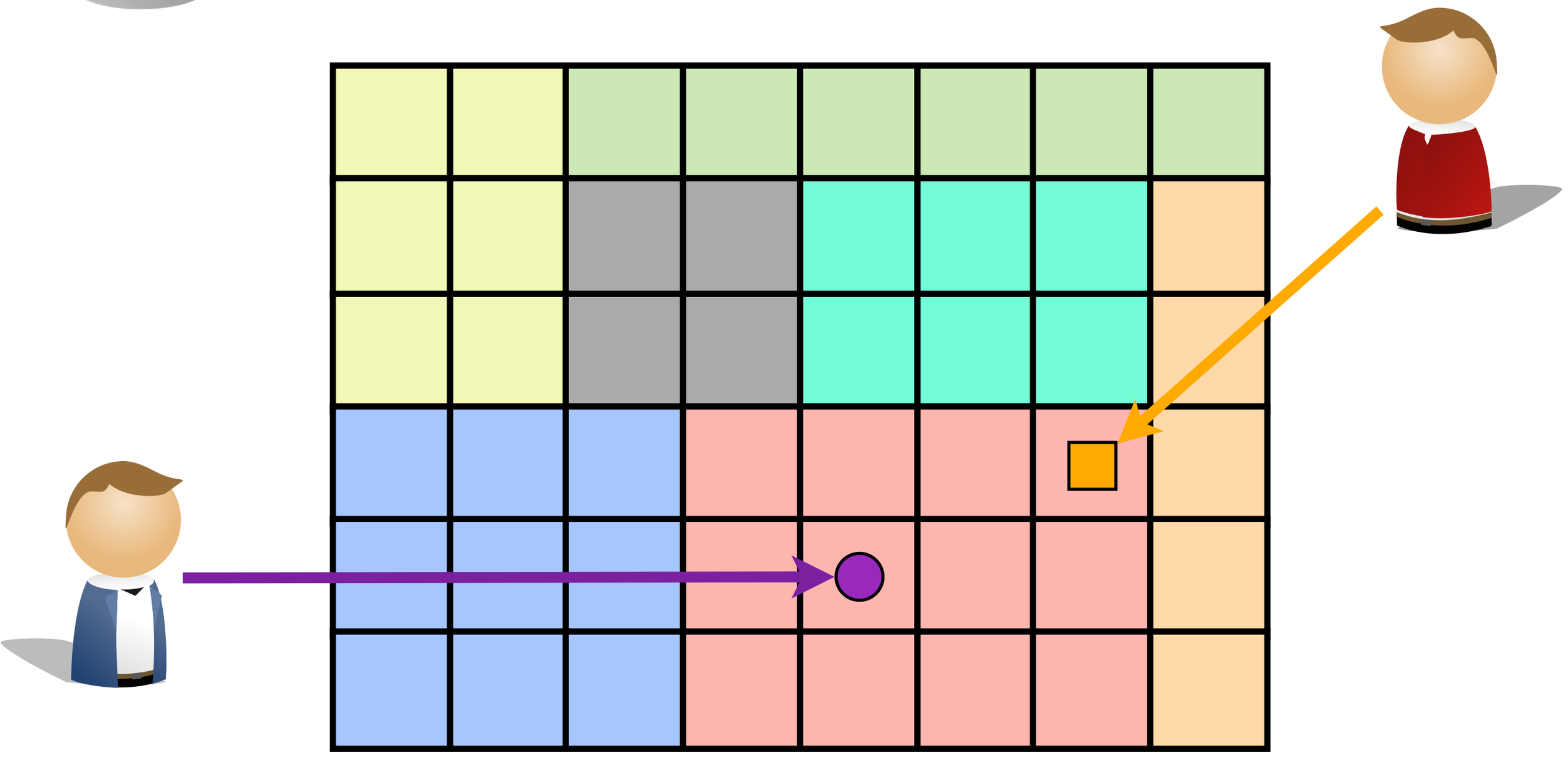
# #2: different part of the same file




File Locking (old school)



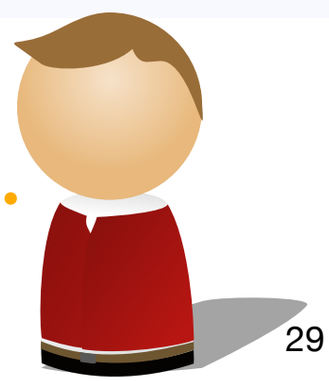
# #2: different part of the same file

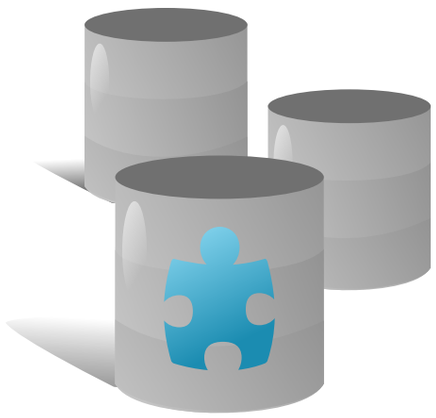


**Automatic merge**

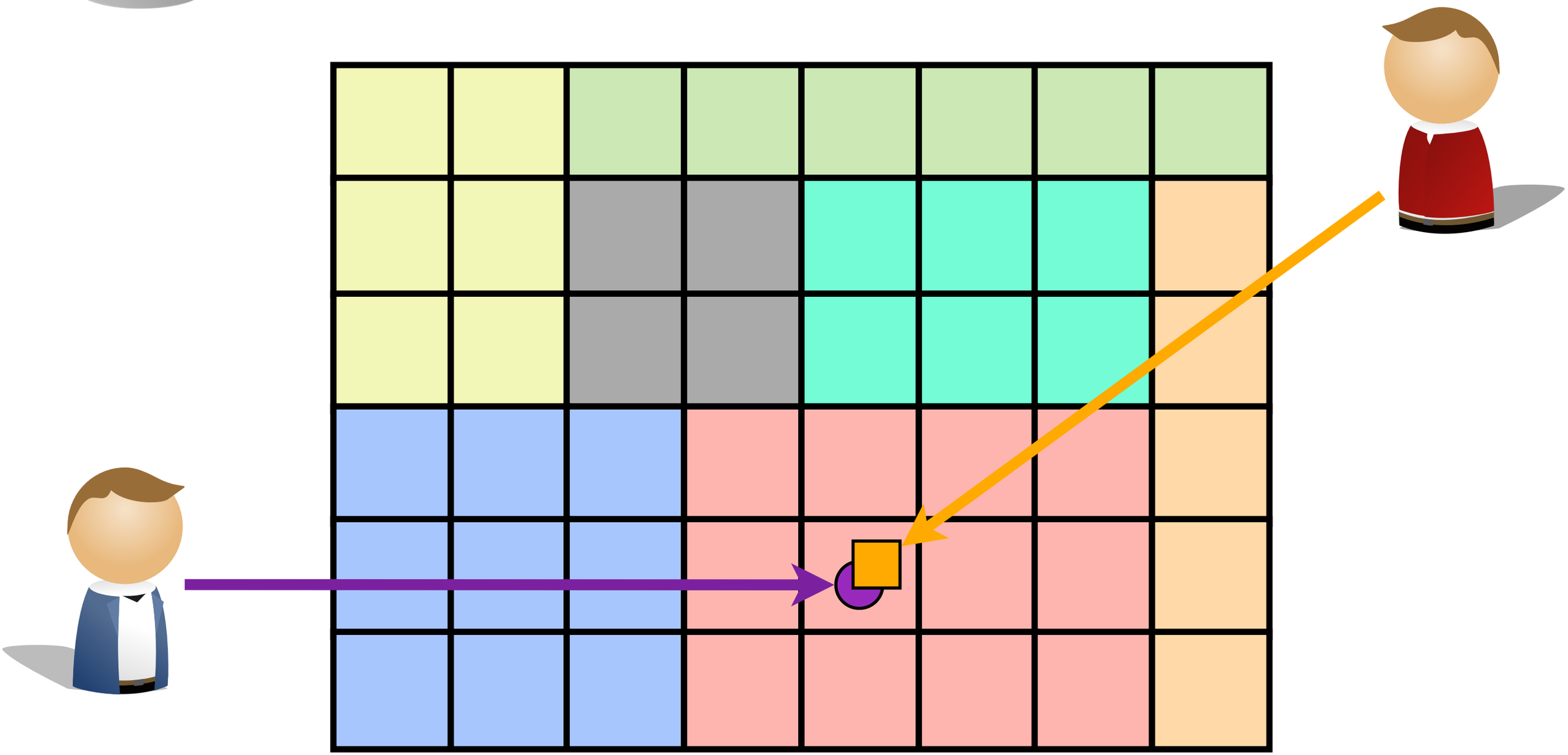
ScalaMOF

```
...  ...  @@ -45,8 +48,8 @@ class Operation  extends TypedElement with MultiplicityElement {
45   48       */
46   49      def `class`: Class = _class
47   50      def class_=(c: Class) {
48        -     require(c != null)
49        -     require(c.ownedOperations contains this)
     51   +     require(c != null, "`class` attribute cannot be null")
     52   +     require(c.ownedOperations contains this, "`class` must contain this operation")
50   53        _class = c
51   54      }
52   55      private[this] var _class: Class = _
...  ...  @@ -54,7 +57,7 @@ class Operation  extends TypedElement with MultiplicityElement {
54   57      /**
55   58       * <em>"The parameters to the operation."</em>
56   59       */
57        -   def ownedParameters: Seq[Parameter] = _ownedParameters
     60   +   def ownedParameters: Seq[Parameter] = _ownedParameters.reverse
58   61      private[this] var _ownedParameters = List[Parameter]()
```



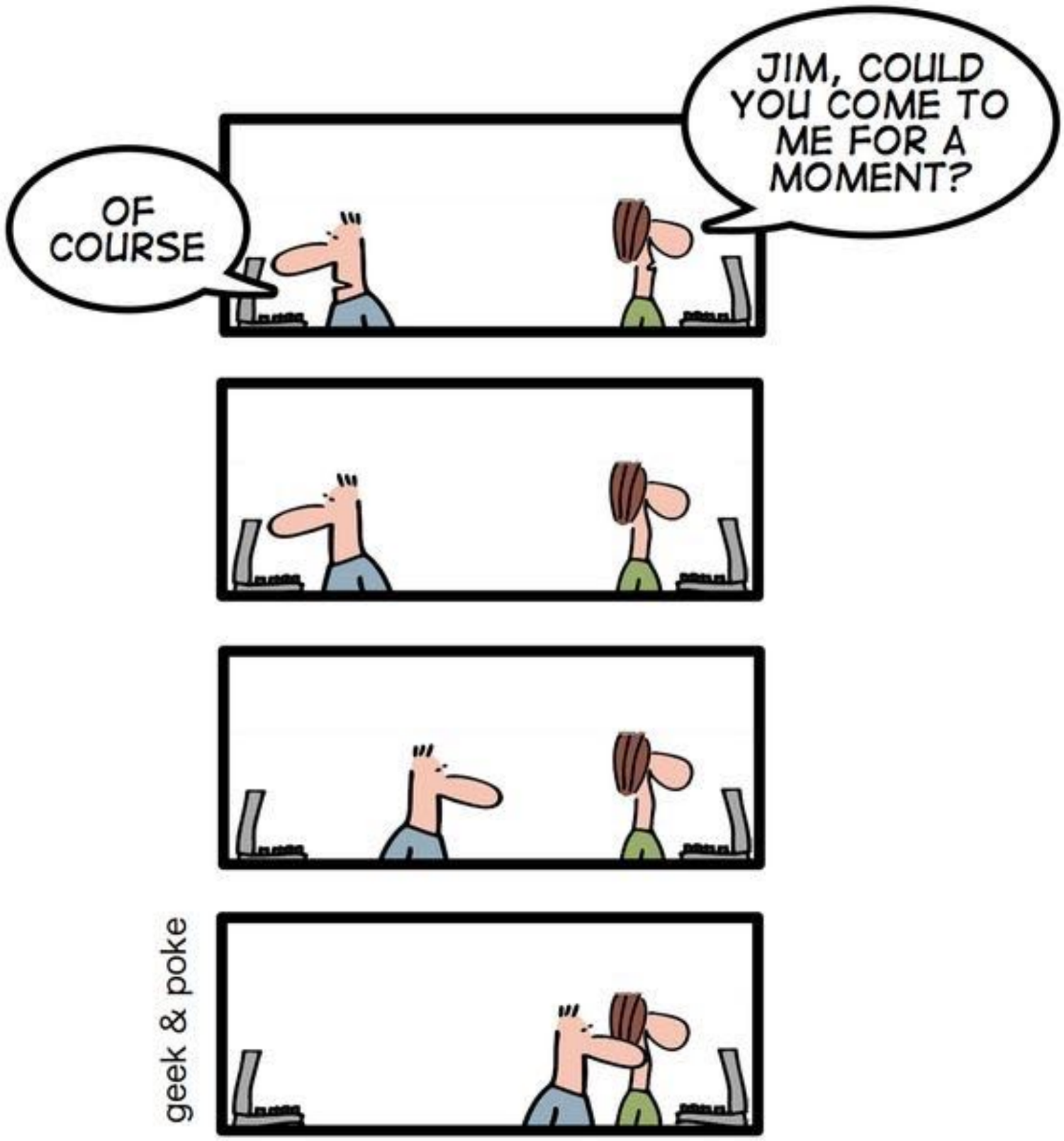


# #3: same part of the same file



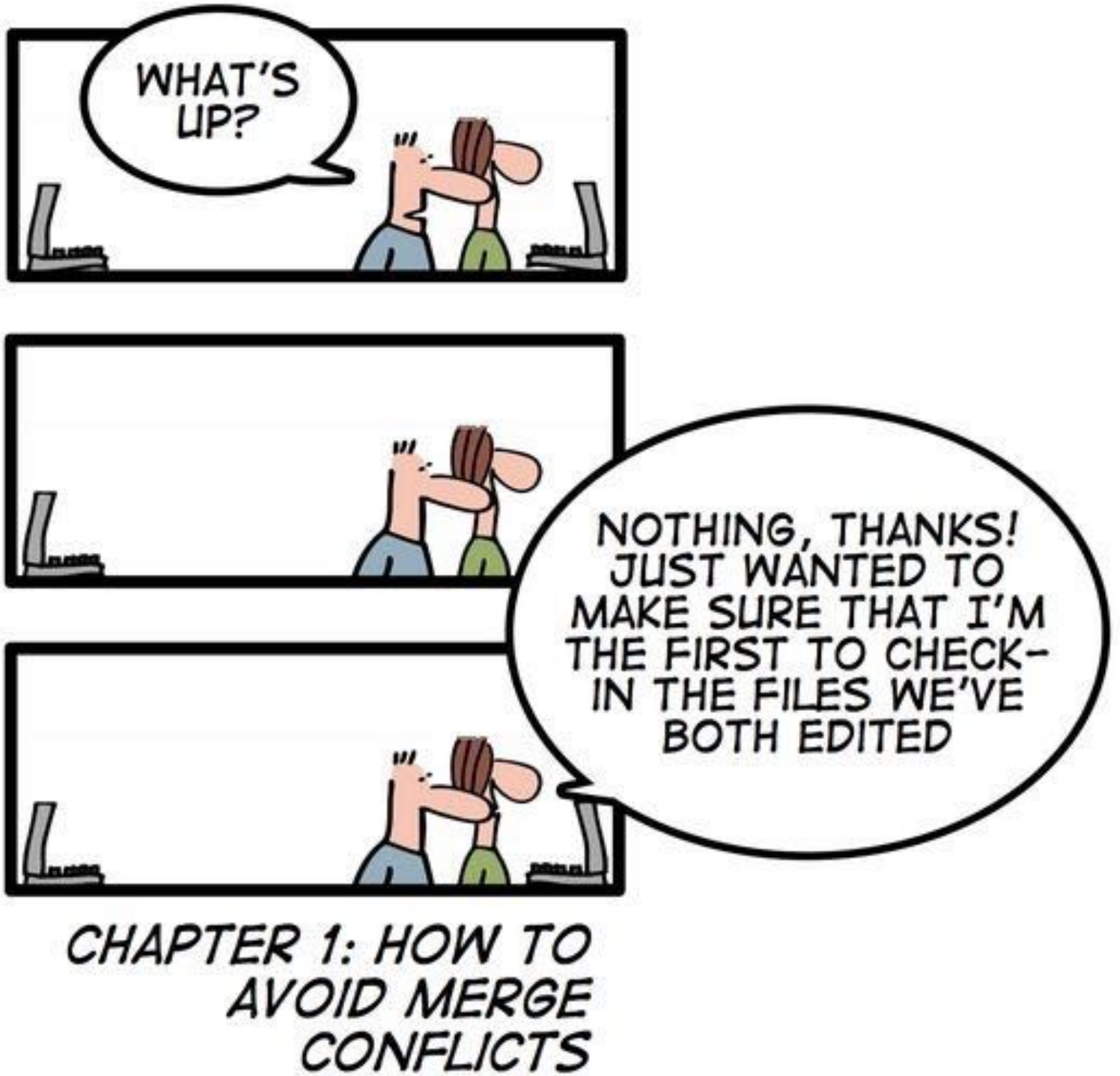
**Conflict!**

# BEING A CODER MADE EASY





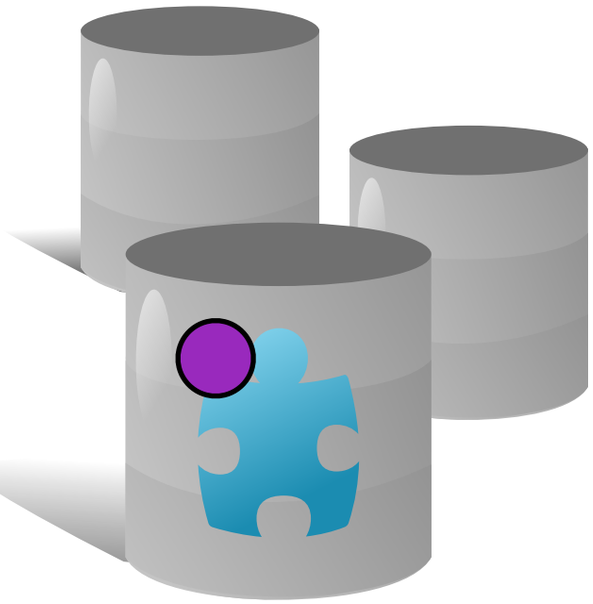

http://geekandpoke.typepad.com/geekandpoke/2010/10/being-a-code-made-easy-chapter-1.html

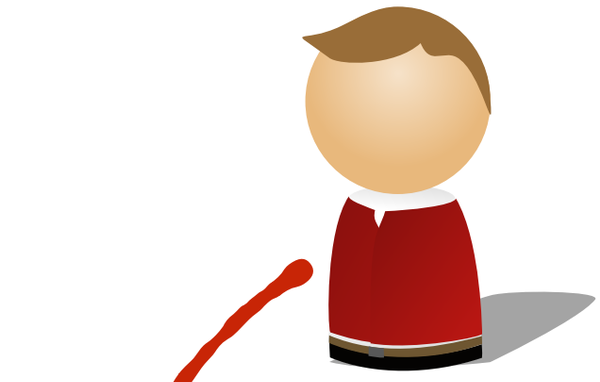




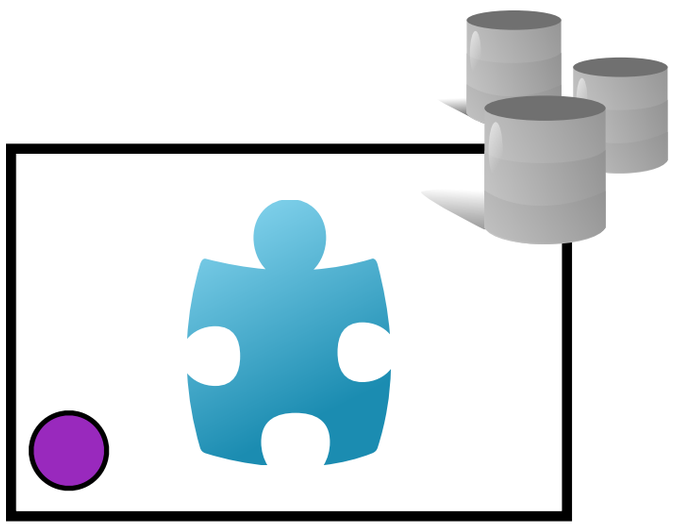

http://geekandpoke.typepad.com/geekandpoke/2010/10/being-a-code-made-easy-chapter-1.html

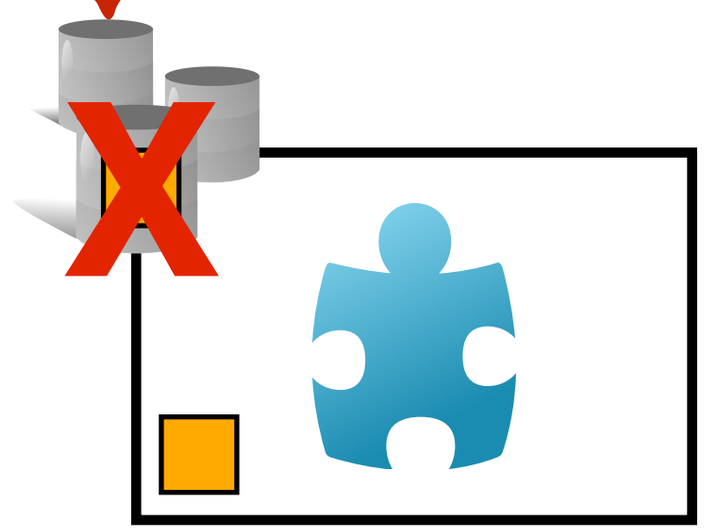
# Shared Repository



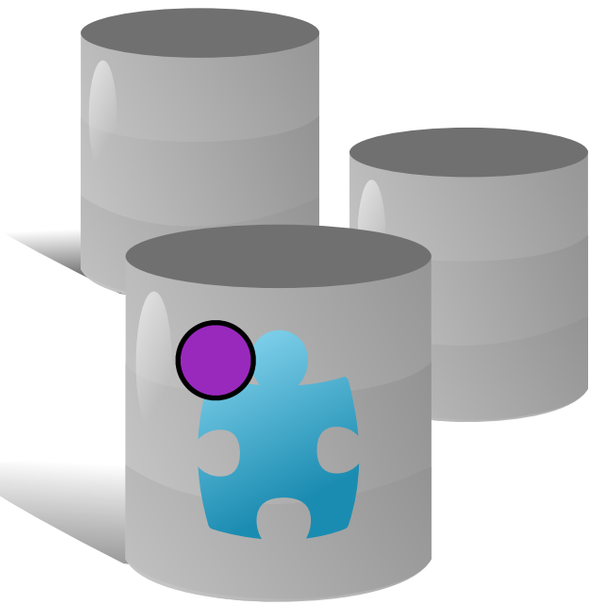


commit

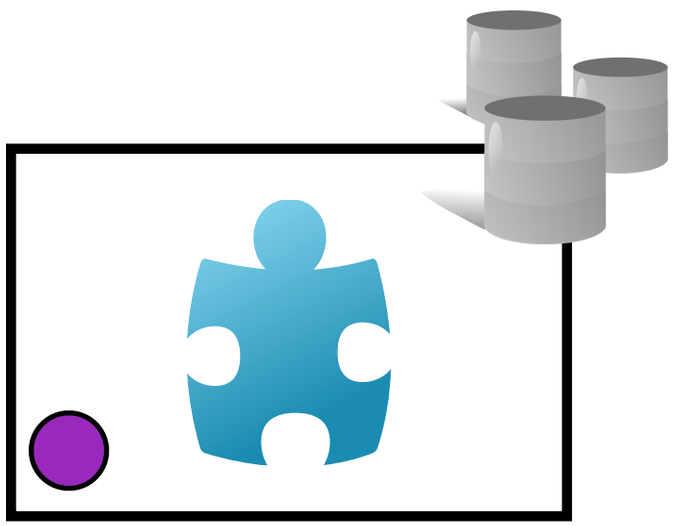


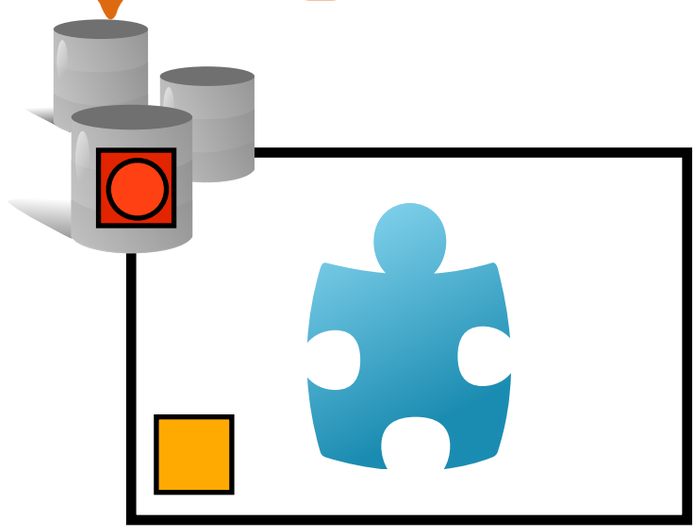


# Shared Repository



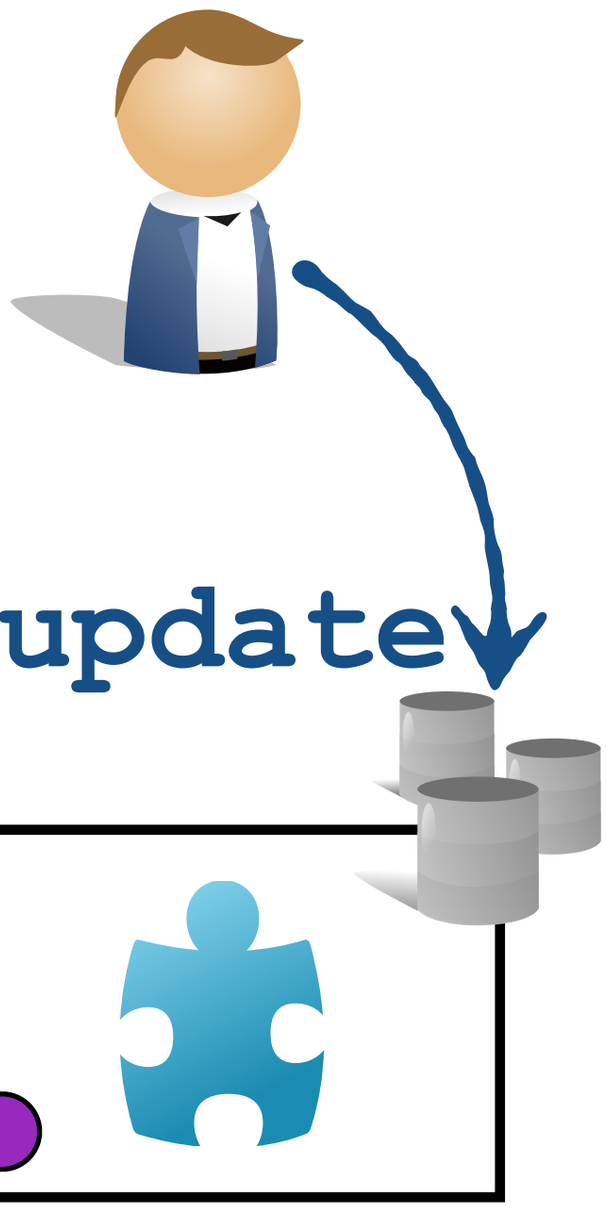
update

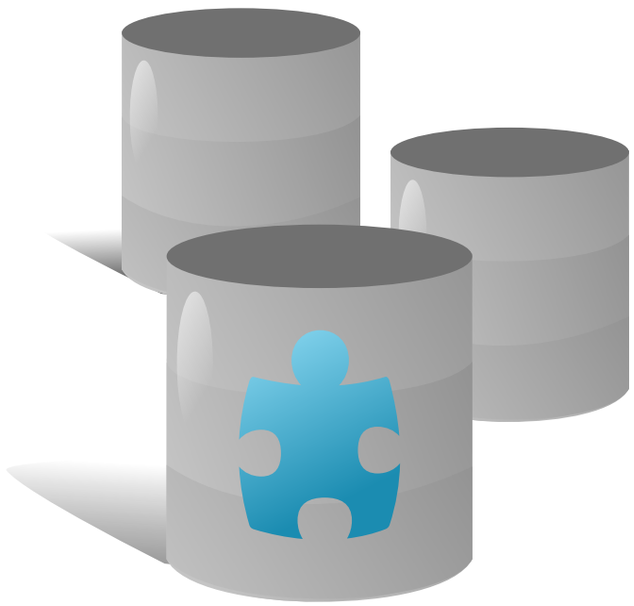


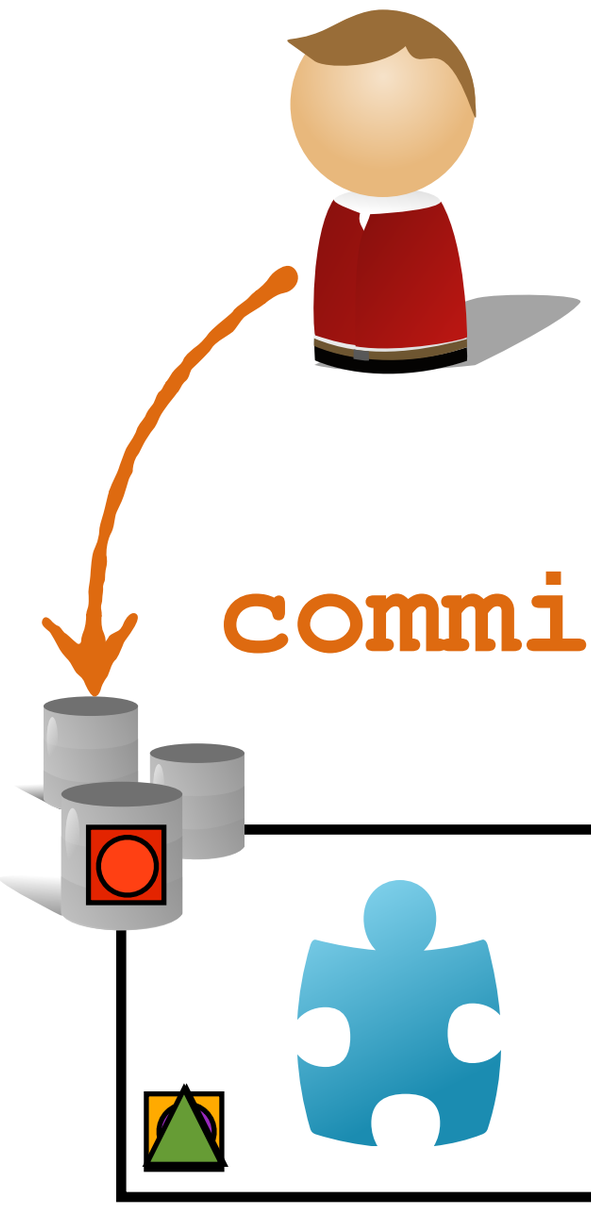


**Conflict!**

# Shared Repository






Resolved!

# **Distributed** Model

(e.g., Bazaar, Git)

Centralized = 1 repository

Distributed = N repository

when **N = 1**, Centralized = Distributed

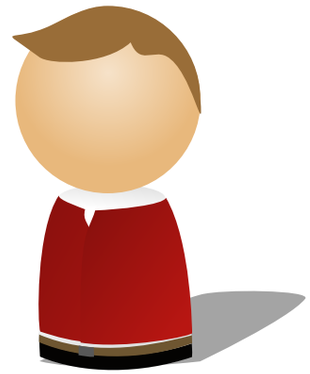
# He who can **do more**

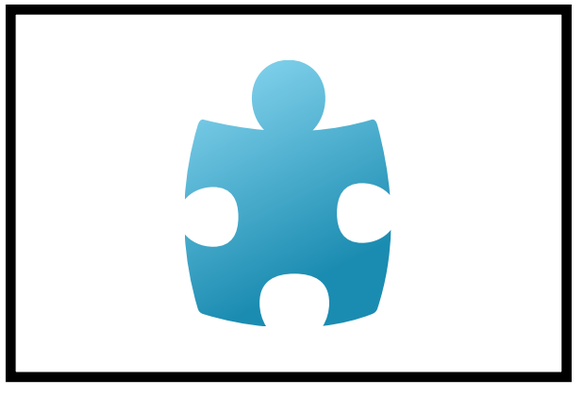
# can **do less**

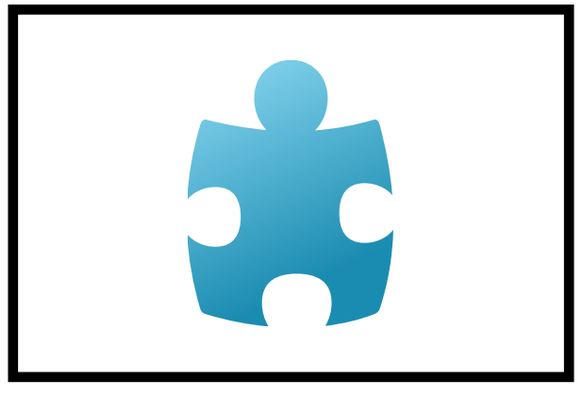
repository

clone

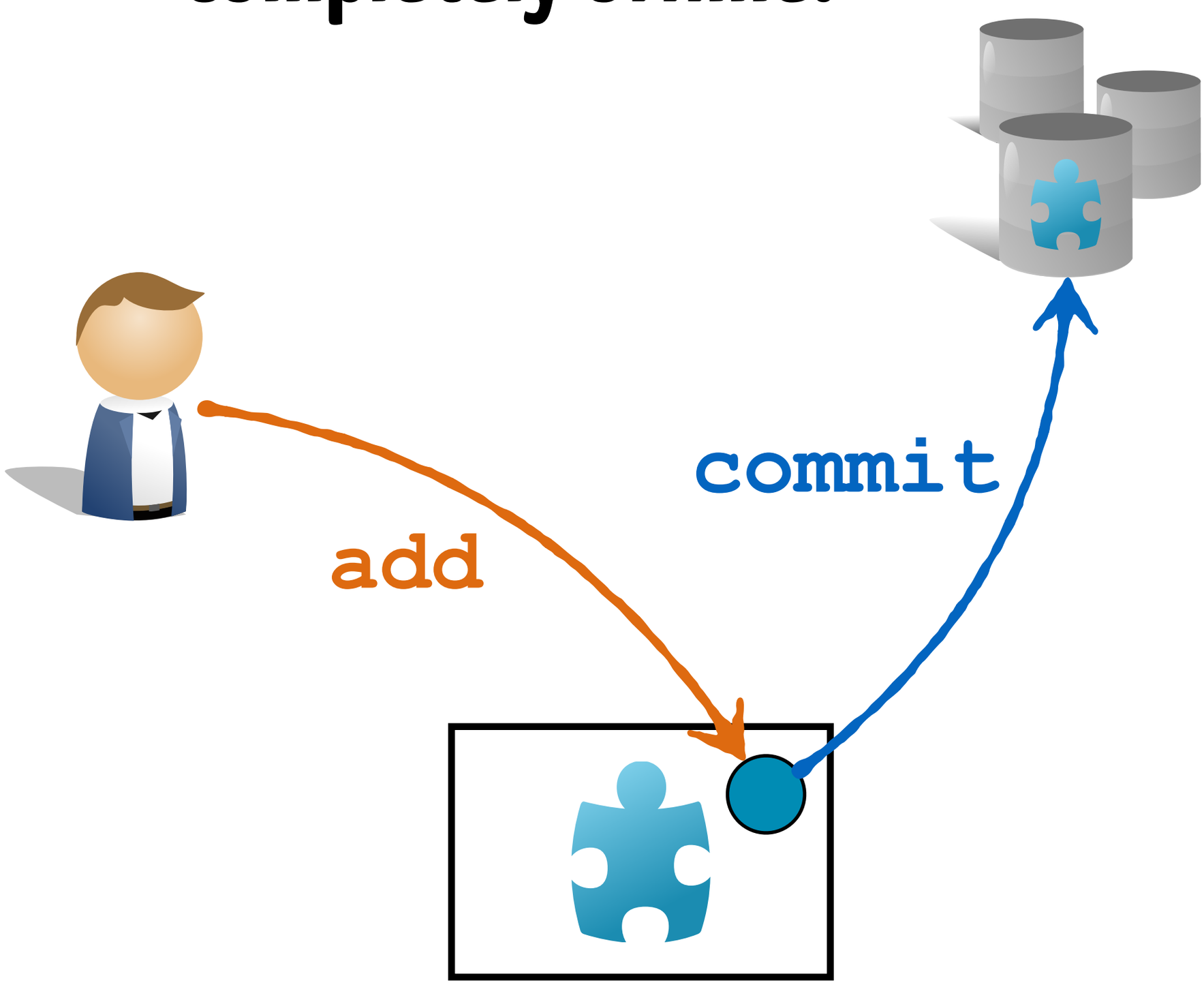
clone

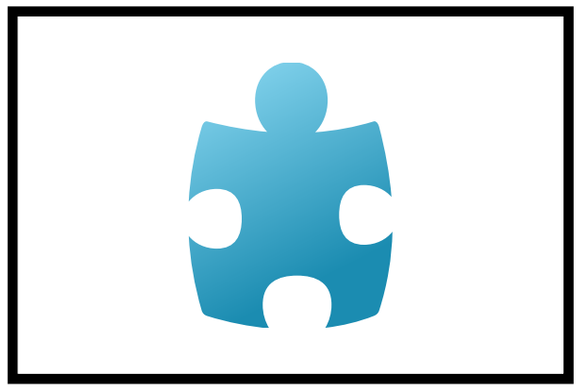




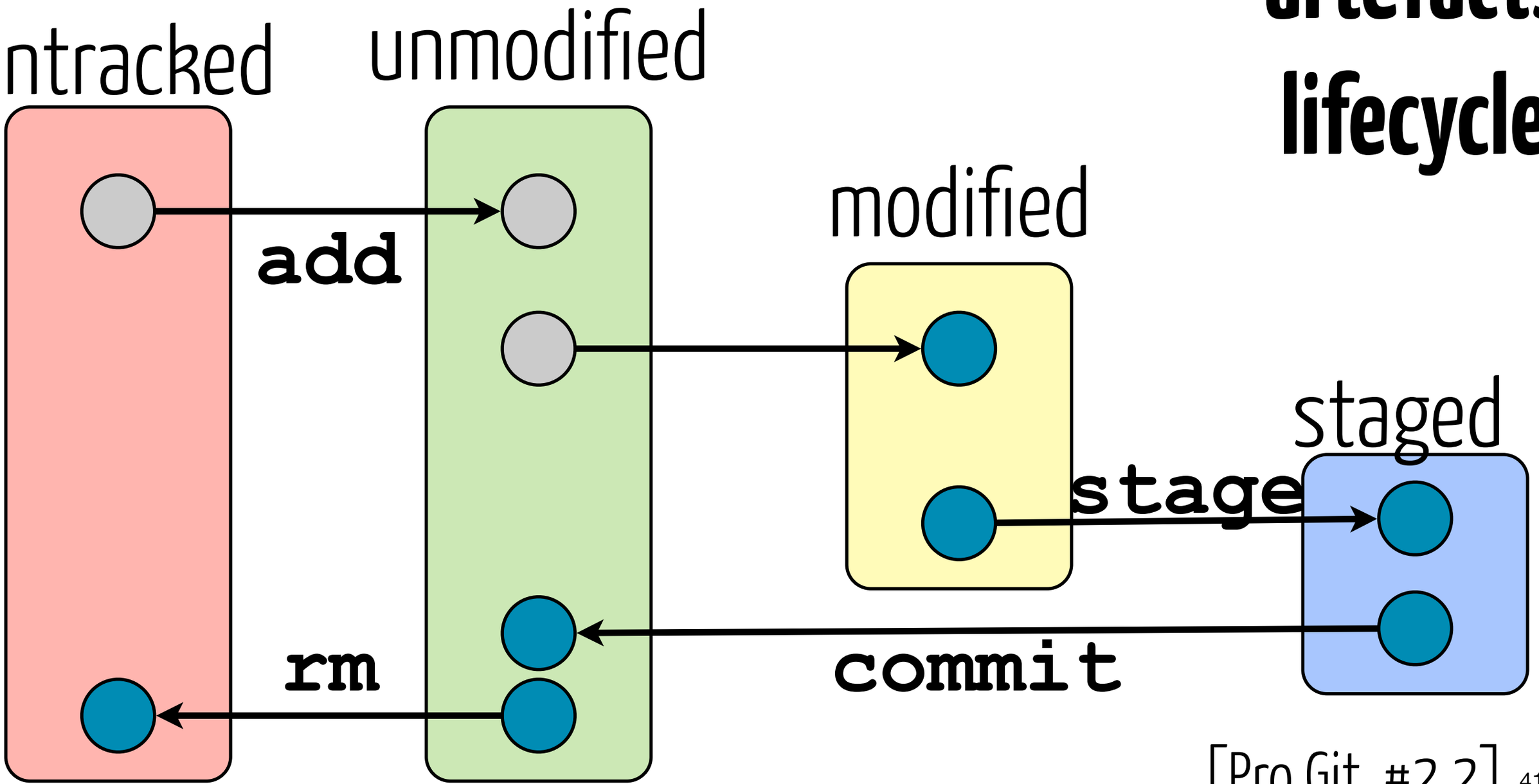


# completely offline!

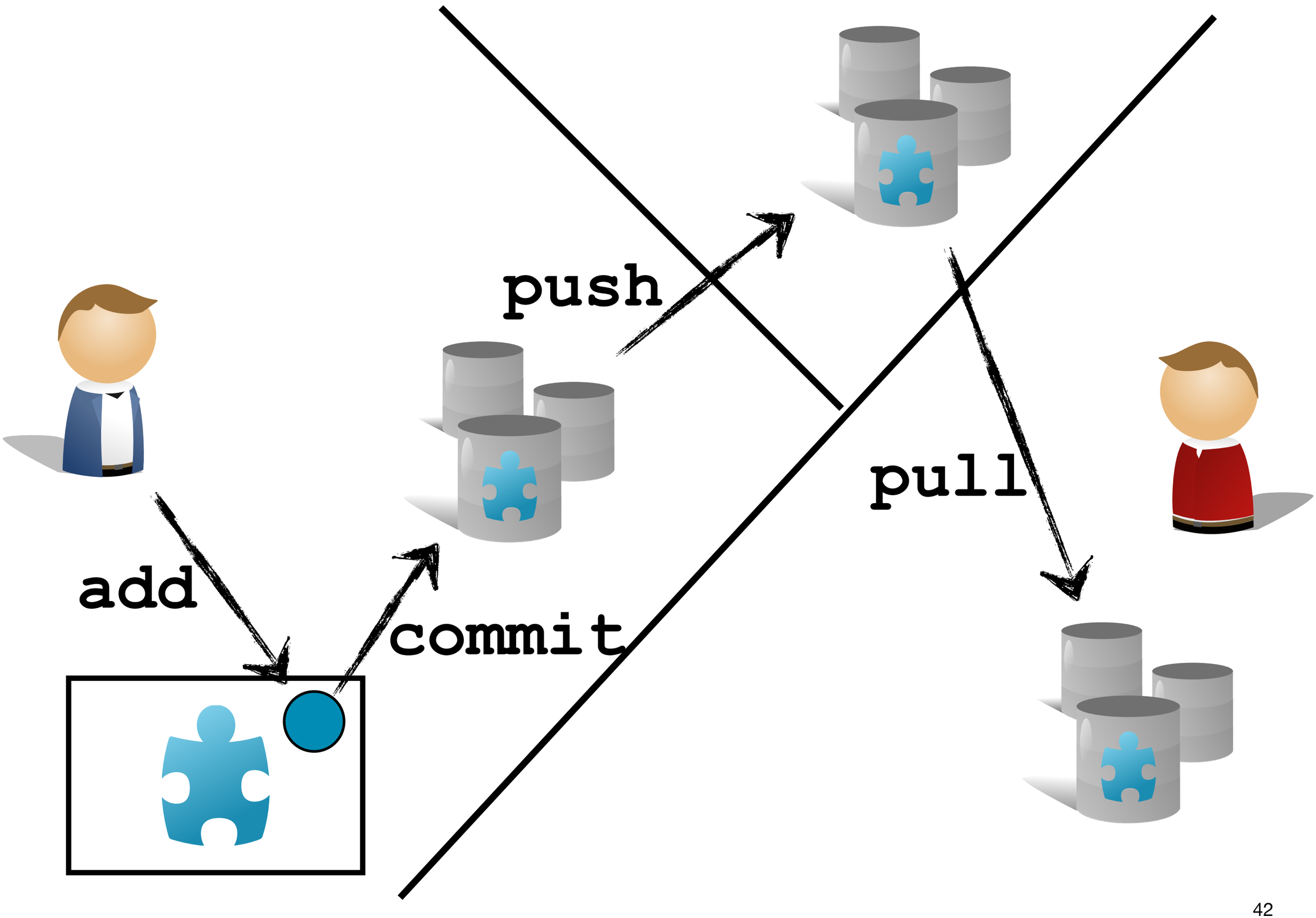


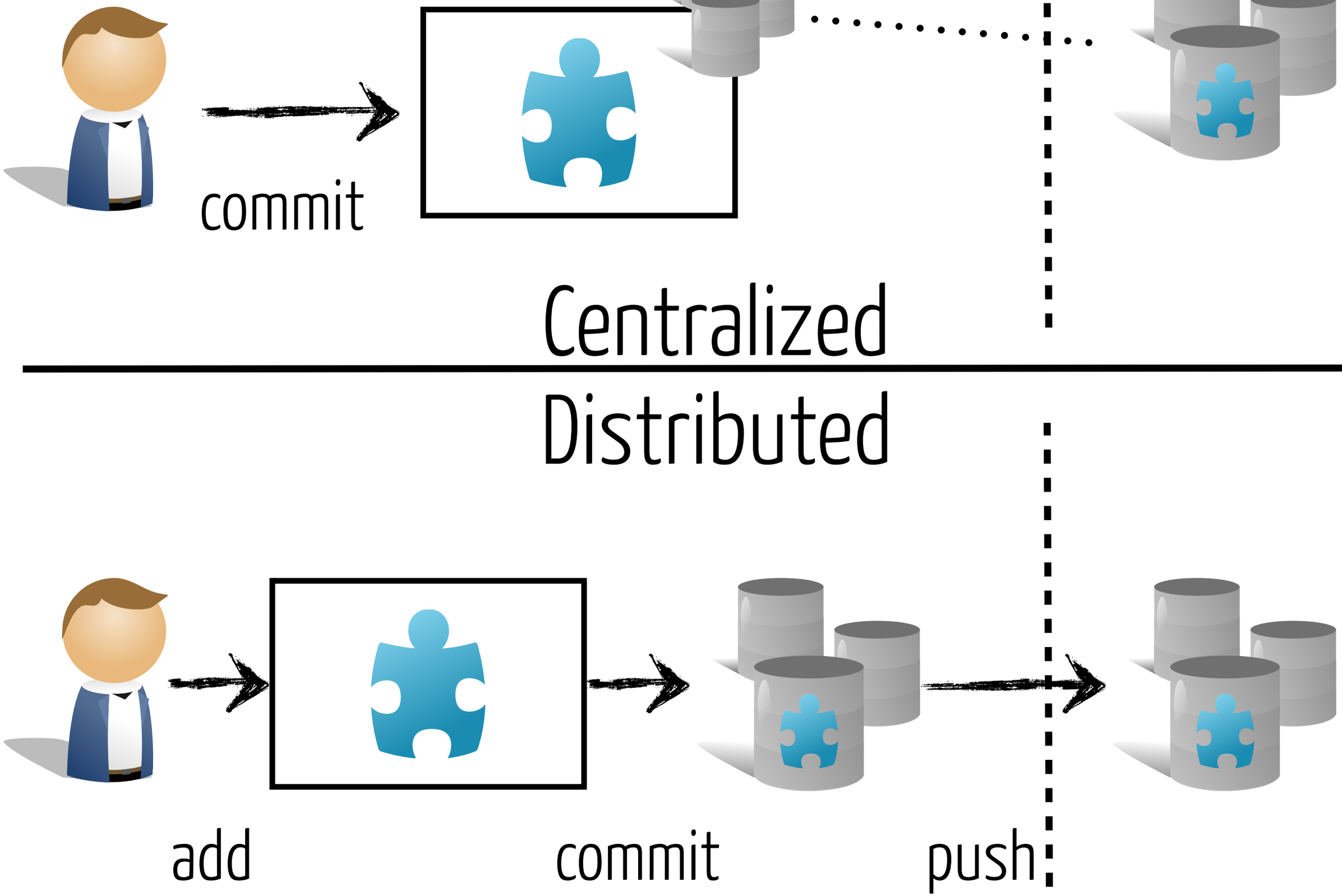


# artefacts lifecycle



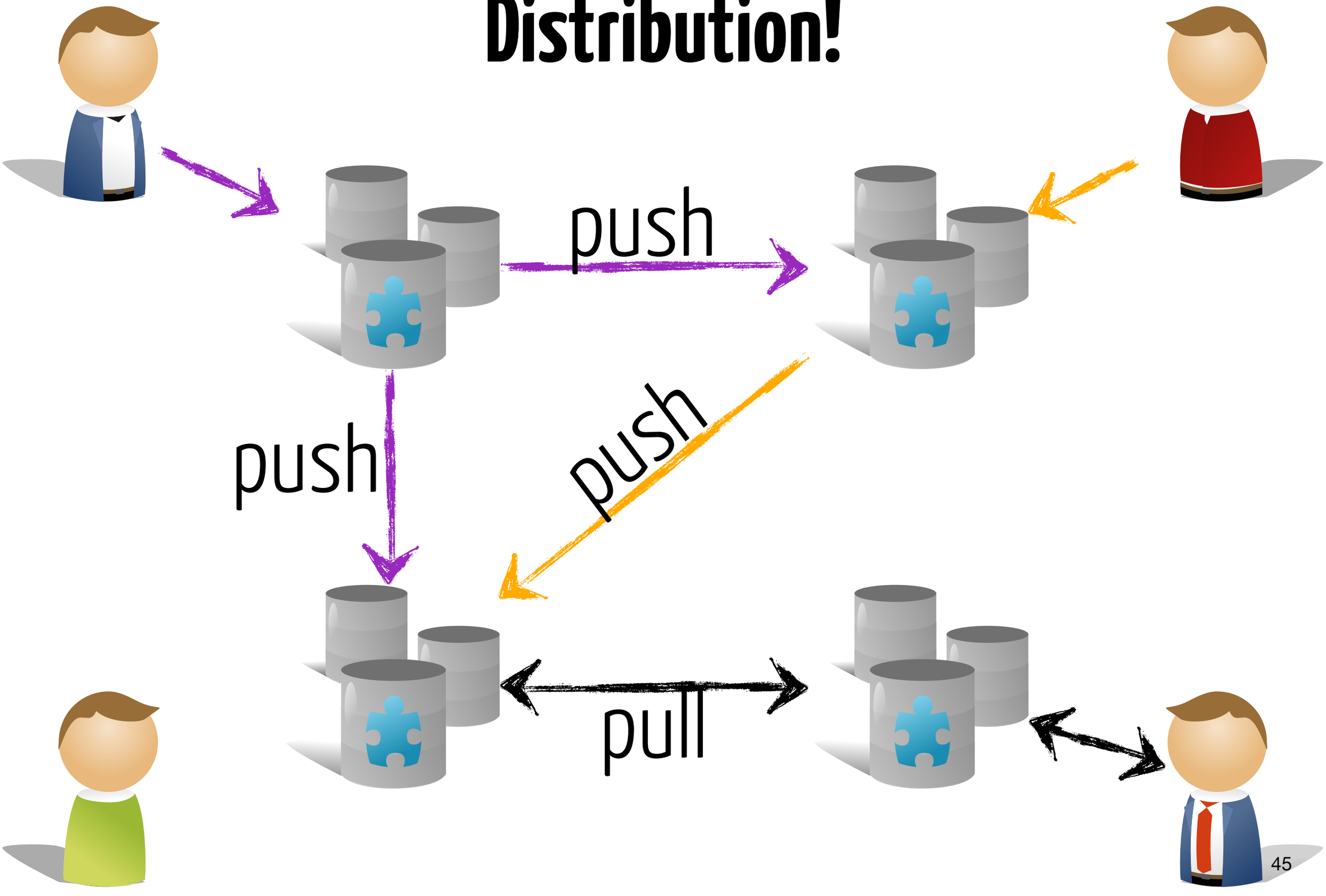

[Pro Git, #2.2]


push
pull
add
commit


commit
Centralized
Distributed
add
commit
push

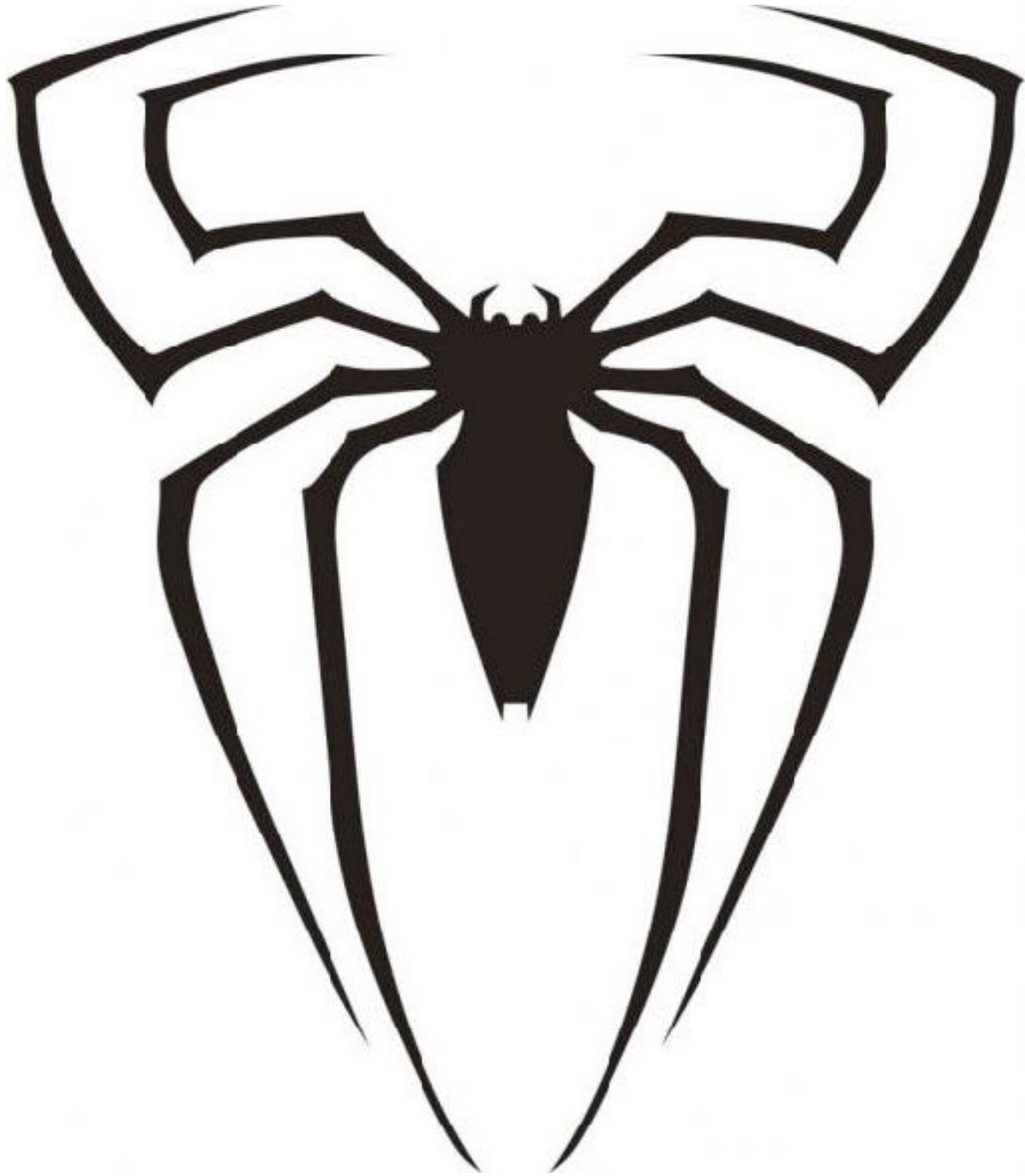
# Seriously?

# Distribution!



# Spiderman's Theorem

«With great power comes great responsibility»




# Conclusions

# **Why** do we version code?

---

To **trace** changes!

To **rollback** changes!

To **share** changes!

(among others)

# **Why** do we **version** code?

---

To **trace** **changes**!

To **rollback** **changes**!

To **share** **changes**!

(among others)

# Different models for code versioning

**Centralized**

versus

**Distributed**

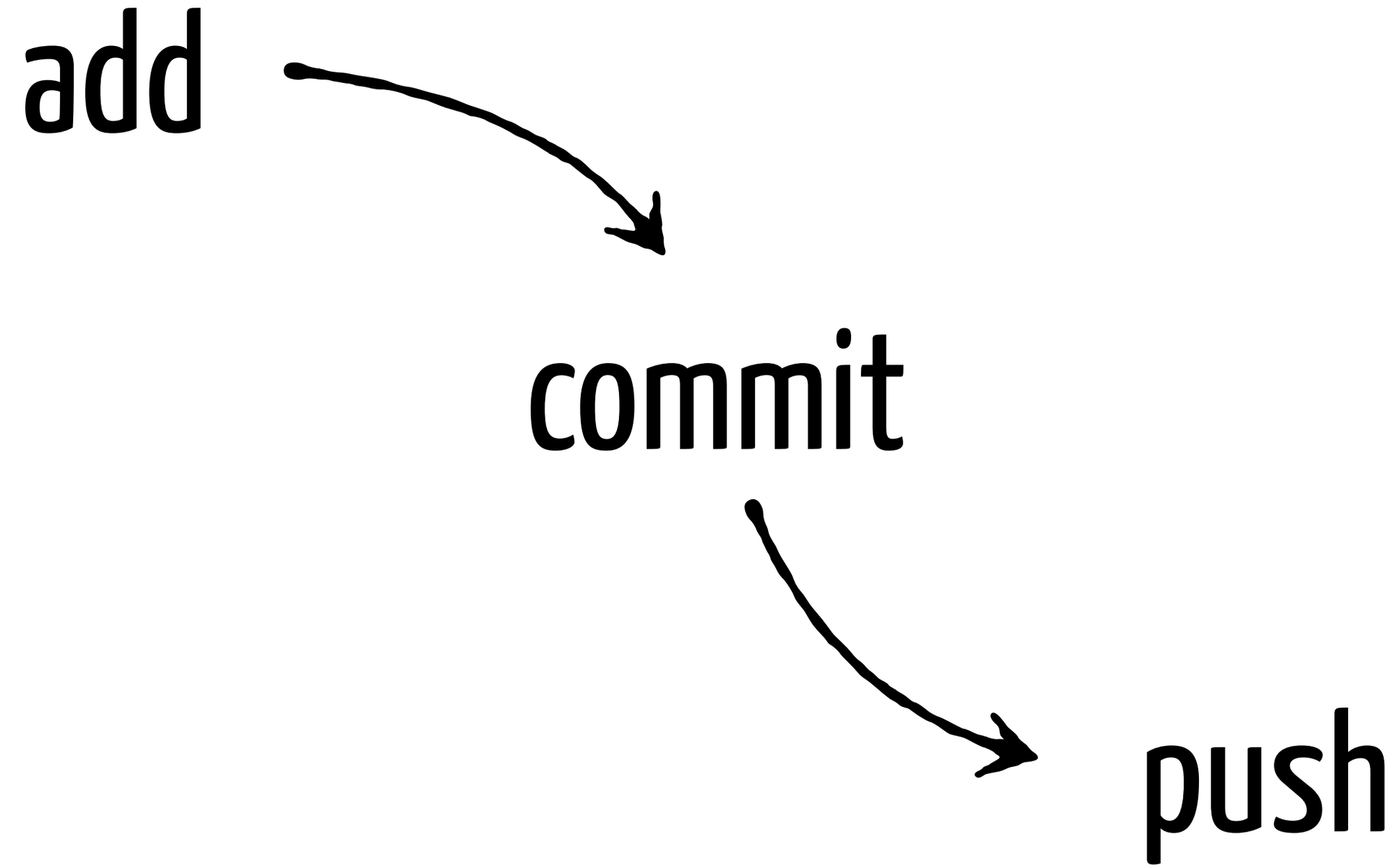
when **N = 1**, Centralized = Distributed

# He who can **do more**

# can **do less**

(also works for P = NP)

# Distributed model to be used during labs



# References

http://www.git-tower.com/files/cheatsheet/Git_Cheat_Sheet_grey.pdf

http://git-scm.com/book/fr

http://pcottle.github.io/learnGitBranching/

http://www.git-tower.com/learn/